Semi-Supervised Classification Techniques in Big Data Text Analytics

by

Geng Li

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
May 10, 2013

Keywords: Semi-supervised classification, Text mining, Naive Bayes classifier

Approved by

Mark Carpenter, Chair, Professor of Mathematics and Statistics
Peng Zeng, Associate Professor of Mathematics and Statistics
Peter M. Nylen, Professor of Mathematics and Statistics
Xiaoying Han, Associate Professor of Mathematics and Statistics

Abstract

Imagine that you are trying to read everything you got in 2011 as soon as possible. That will take the first three months of 2012. How can we get access to the vast unstructured literature, automatically process it, effectively predigest and make sense of it with less effort? Focusing on the entire preprocessing and classification steps, a hybrid semi-supervised text classification approach proposed in this dissertation will help you survive in a rising sea of information. The porter stemming, new adaptive TFIDF-LDA weighting, Zipf's law based dimension reduction, multinomial Naive Bayes classifier, and Expectation-maximization algorithm are harmoniously integrated together in the hybrid semi-supervised text classification model. From a small set of “known” labeled papers, you can use this mixture model to make predictions about newly “unknown” unclassified papers into the predefined categories. Extensive experimental results show that the proposed system dramatically reduces the feature dimension and improves the classification accuracy.

Acknowledgments

I would like to thank my advisor, Professor Mark Carpenter, for his guidance and endless support during my studies at Auburn University. I also thank my thesis committee members, Professor Peng Zeng, Professor Peter Nylen, Professor Xiaoying Han and Professor Roy Hartfield. Thank you for the countless hours spent in coaching me throughout my graduate study. Thank you so much for the time you invested in me personally and academically.

Life is a journey, I enjoy each moment of it. Thank you, my father, for supporting me, encouraging me, and for always being there helping me grow. Thank you, my husband Jun Yin, for your unconditional love, for incessantly caring me, for everything. Because of you, my life has been forever changed. Thank you for making me love and always happy.

I want to thank you again, every precious one, for giving me the best day of my life.

Table of Contents

Abstract ii
List of Tables vii
List of Figures viii
List of Abbreviations ix
Chapter 1 1
Introduction 1
1.1 Text Mining 1
1.2 Big Data 3
1.3 From documents to data 4
1.3.1 Term-document Matrix 4
1.4 Aspects of the Dissertation 5
1.4.1 Notations 6
Chapter 2 8
Model Background 8
2.1 Motivation of the Method 8
2.2 Machine Learning Methods: Unsupervised, Supervised, Semi-Supervised 9
2.2.1 Unsupervised Learning 10
2.2.2 Supervised Learning 11
2.2.3 Semi-Supervised Learning 12
2.3 Datasets and Protocol 12
2.3.1 Corpus 1: Five Classes Journal Papers 15
2.3.2 Corpus 2: Seven Classes Journal Papers 15
2.3.3 Corpus 3: 20 Newsgroups 16
2.4 Experiment Frame and Hybrid Model 17
Chapter 3 21
Text Preprocessing 21
3.1 Overview 21
3.2 Term Filtering 21

3.2.1 Stopwords & punctuations removal 21
3.2.2 Excluding too short & long strings 22
3.3 Porter stemming algorithm 22
3.4 Zipf's law based dimension reduction 23
Chapter 4 24
Feature Extraction-Applications of TFIDF-LDA Weighting 24
4.1 Overview 24
4.2 TF-IDF Weighting 24
4.2.1 Advanced TF-IDF Weighting 25
4.3 Probabilistic Topic Model-Latent Dirichlet Allocation (LDA) 26
4.3.1 LDA Model 27
4.4 Adaptive TFIDF-LDA Weighting 31
Chapter 5 37
Semi-supervised Naive Bayes Training & Classification 37
5.1 Mixture model frame 37
5.2 Naive Bayes Classifier 38
5.2.1 Introduction 38
5.2.2 Multinomial Naive Bayes Classifier 38
5.3 Naive Bayes+EM Algorithm on Labeled and Unlabeled Training Data 40
Chapter 6 43
Model Evaluation and Experimental Results 43
6.1 Confusion Matrix 43
6.2 Performance Measures: Precision, Recall, Accuracy & F-measure 45
6.2.1 Precision 45
6.2.2 Recall 46
6.2.3 Accuracy 46
6.2.4 F-measure 47
6.3 Experimental Results 49
6.3.1 Experiments on Corpus 1: Five Classes Journal Papers 49
6.3.2 Experiments on Corpus 2: Seven Classes Journal Papers 59
6.3.3 Experiments on Corpus 3: 20 Newsgroups 65
6.3.4 Comparative Experiments with Unsupervised, Supervised and Semi-Supervised Learning 68
Chapter 7 71
Conclusion and Future Work 71

7.1 Experiment Findings 71
7.2 Future Work 71
References 73

List of Tables

2.1 Data Descriptions. 14

6.1 The confusion matrix for a three class classifier 43

6.2 Confusion matrix for the 5-class text classification 44

6.3 Precision, recall and the F_1 score for the corpus 1 with 142 test sets 50

6.4 Comparison of three document classification methods on corpus 1 58

6.5 Precision, recall and the F_1 score for the corpus 2 with 226 test sets 14

6.6 Comparison on corpus 2 using number of used features, macro-F1 score and accuracy 64

6.7 Precision, recall and the F1 score for the corpus 3 with 300 test sets 66

6.8 Proposed method on corpus 3 using macro-F1 score and accuracy 67

6.9 Comparative experimental results with three machine learning methods 69

List of Figures

Figure 2.1: Interactivity of three types of machine learning..... 9

Figure 2.2: Experiment frame and hybrid model..... 19

Figure 4.1: Comparison of TFIDF-LDA, Hybrid TFIDF and TFIDF Standard methods 35

Figure 5.1: Mixture probability model frame 37

Figure 6.1: Comparison of F_1 score for class-1 mathematical education on corpus 1..... 52

Figure 6.2: Comparison of F_1 score for class-2 aerosol research on corpus 1..... 53

Figure 6.3: Comparison of F_1 score for class-3 text mining documents on corpus 1..... 54

Figure 6.4: Comparison of F_1 score for class-4 nuclear research documents on corpus 1 54

Figure 6.5: Comparison of F_1 score for class-5 image processing documents on corpus 1 55

Figure 6.6: Comparison of F_1 score for all 5-classes document classification on corpus 1 56

Figure 6.7: Comparison of F_1 scores using modified hybrid TFIDF algorithm 57

Figure 6.8: Comparison of F_1 scores using standard TFIDF algorithm. 57

Figure 6.9: F_1 score for each class on corpus 2..... 61

Figure 6.10: F_1 score for all seven classes on corpus 2 using proposed hybrid model..... 63

Figure 6.11: Comparison of macro-F_1 and accuracy curves on corpus 2..... 64

Figure 6.12: Comparison of macro- F_1 and accuracy curves on corpus 3 67

List of Abbreviations

EM	Expectation-Maximization Algorithm
LDA	Latent Dirichlet Allocation
LSI	Latent Semantic Indexing
PCA	Principal Component Analysis
pLSI	Probabilistic Latent Semantic Indexing
TFIDF	Term Frequency Inverse Document Frequency
VSM	Vector Space Model

Chapter 1

Introduction

1.1 Text Mining

According to Bloomberg Business Week, the average person receives about 63,000 words of new information a day [1]. That is about the length of a novel. The average computer user checks 40 websites a day and switches programs 36 times an hour [2]. That is about once every two minutes to change tasks. If you had a crazy idea to read everything you got in 2011. That is about the first three months of 2012 to spend. So, does it make us superhuman to acclimate to extracting information from the rapidly expanding literature? Tens of thousands of journals exist, and the deluge of new articles is leading to information overload. 30 percent of people under 45 said the use of devices such as smart phones and personal computers has made it harder for them to concentrate. In this era of information explosion, computers double in speed every year. The use of machine learning dramatically enhances the human ability to parse and understand the complex text. Hence, the common existence of numerous unstructured text documents and the ongoing importance of the textual information undoubtedly make text mining technology become a persistently interesting hotspot.

What is text mining? As a variation on a field of data mining which searches for interesting patterns from large databases, text mining is data mining using text documents as data. Text mining, also known as text data mining or knowledge discovery from textual databases, refers

generally to the process of extracting interesting and non-trivial patterns or knowledge from unstructured text documents [3] [4].

What is the purpose of text mining? In a nutshell, the purpose of text mining is to turn text into numbers and then automatically extract information contained in the text using various statistical and machine learning algorithms. The extracted information develops new observations or new hypotheses to be analyzed further in order to derive summaries for the words used in the documents or to find summaries and similarities of the documents based on these words. The goal of text mining is to discover heretofore unknown information from different written resources.

What are the text mining applications in this paper? For researchers, librarians and publishers, text mining is already being used and deployed in many ways across scholarly communications. To find the most useful scientific information hidden or buried in the literature, one can save a lot of time and money with ease by letting a computer repeatedly filter the text from tens of thousands of documents for you. Text mining techniques explored in this paper can easily help busy real-life researchers to efficiently exploit a large number of papers, quickly navigate the latest researches, more effectively predigest and go digging deeper into unstructured textual information of the literature, and ultimately pulling out the most relevant ones to further their research.

Text mining definitely can be an imperative alternative to the labor-intensive and manual intervention scholarly researches. Text mining already plays an important role in some researches. Unfortunately, in most cases, although copyright law and some other barriers are limiting the use of text mining at this time, text mining still overwhelmingly promises huge economic and research benefits. Some biomedical researchers have already pushed for the end of

publishers' default ban on computer scanning of medical papers to find links between genes and diseases. The impact of text mining will continue increasing in the following years and certainly its application will finally move away from the experts' desks to become a daily used tool by every researcher. The next big thing is already here-Text mining.

1.2 Big Data

Unstructured and massive information from YouTube, Facebook, Instagram, and Twitter flooded into our daily life. Have you still suffered from "Facebook addiction syndrome"? A recent claim from IBM showed that humans create around 2.5 quintillion bytes of data every day and almost 90% of the data in the world today has been created in the last two years alone [5]. Is this claim true or just a marketing hype? Although there are no rigorous research resources to verify this statement, we have to admit that a new era of big data has already happened. The tsunami of the large volumes of data makes the notion of "big data" a big trending term.

According to the definition from Wikipedia online dictionary, big data is defined as a collection of large and complex data sets which are very difficult to process using available database management tools [6]. Big data do not only mean big size, but also the big complexity. Actually, there are four major characteristics of big data which are volume, velocity, variety, and veracity [5]. Currently, many types of data accumulating gigabytes, terabytes even petabytes of information have already tremendously changed the way we view, learn, and analyze the world [7]. As the blood of a company, the volume of business transactional data has being dramatically increased from storing in relational databases to ever-growing cloud-based electronic database. Velocity shows the streaming data in motion feature and the large volume data movement [5]. There is also great variety in big data. We have to manage many different forms of data such as

structured, unstructured, text, and even multimedia. Veracity refers to the data quality. The decision based on the big data may have some uncertainty and deflection because of data ambiguities, inconsistency, incompleteness, or model approximations.

Big data analysis gives us intensive potential opportunities and meanwhile big technical challenges. Big data is more than just a large quantity of numbers. Not only the voluminous scale, but also heterogeneity, format variability and error-handling [8], timely and effective exploitation of big data urges the deep human insights and better analysis solutions.

1.3 From documents to data

How are the textual documents converted to data? The idea of the vector space model (VSM) [9] [10] [11] is widely used in text mining area for extracting knowledge automatically from given documents. Given a huge document collection, a set of terms or features can be defined as a point in a space or a vector in a vector space [12] to represent each document in a collection.

There are three stages, the document indexing, term weighting, and the document ranking stage, to build the vector space model [13]. In the document indexing step, the useful terms are extracted from the textual documents. Then the frequency of occurrence of the indexed terms will be calculated by a weighting scheme, and finally the documents will be ranked according to the similarity measure [14].

1.3.1 Term-document Matrix

In the term weighting stage, how can one consider the number of occurrences of a term in a document before applying the common weighting scheme? The most important basic step is to

construct a term-document matrix. Term-document matrix is a mathematical matrix which describes the frequency of terms that occur in a collection of documents [15] [16]. The columns of a term-document matrix correspond to the documents in the collection and rows correspond to the terms within the documents. Each document is a count vector and term-document Matrix tabulates the terms in all documents and how many times they appear.

1.4 Aspects of the Dissertation

In Chapter 2, the motivation and the experimental frame of the new hybrid semi-supervised learning model will be introduced. The comparison of three different machine learning methods of unsupervised, supervised, semi-supervised, both pro and con, will be discussed in details. In Section 2.3, the data sets used to analyze the experiments will be presented. A flow chart Figure 2.1 will also illustrate the whole experiment frame and proposed hybrid model.

In Chapter 3, the procedures used for text preprocessing will be outlined step by step. Term filtering, porter stemming algorithm, and Zipf's law based dimension reduction processes will be effectively implemented on the unstructured textual data in order to extract meaningful numeric indices from the text.

In Chapter 4, the feature extraction steps will be considered. A new adaptive TFIDF-LDA weighting approach will integrate the TF-IDF weighting theory and the application of latent dirichlet allocation probabilistic topic model to improve the recall and precision of text classification.

Chapter 5 will portray a whole picture of semi-supervised multinomial naive Bayes training and classification. Based on the labeled and unlabeled mixture training data, in this dissertation, naive Bayes classifier and Expectation-Maximization (EM) algorithm will be applied together in

a semi-supervised setting. In Chapter 6, the model evaluation and results will measure the accuracy of the proposed classification model. At last, Chapter 7 will summarize the main points of the dissertation and present the major experimental findings.

1.4.1 Notations

Before describing the hybrid model, several symbols and mathematical notations used throughout the dissertation will be presented as follows.

1. Word, the basic unit of literary text data, is defined as an item from a vocabulary list. In this dissertation, word is indexed by *{1, 2, ... ,V}*. *V* is the total number of vocabulary terms. The total number of words is noted as *N* and the *v*th word in the vocabulary list is represented by a *V*-vector *w* such that $w^v = 1$ and $w^u = 0$ for $u \neq v$.

2. A document is denoted by a sequence of *N* words $\mathbf{w} = (w_1, w_2, w_n)$, where w_i is the *i*th word in the sequence. $|D|$ is the total number of all documents in the collection. The collection of $|D|$ documents noted by $O = \{\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_D\}$ is called a corpus in text mining. $|V_j|$ is the total number of words in a document *j*.

3. The raw frequency of a term *i* in a document *j*, or the number of times that term *i* occurs in document *j*, is denoted as f_{ij}. The term or word type is also the unique word as a dictionary entry. df_i represents the amount of documents in the corpus that contain term *i*.

$\sum_{j=1}^{|D|} f_{ij}$ is the total number of times that a term *i* occurs in all documents, *i* is the term index and *j* is the document index.

4. The conditional probability of a word w_i given topic $z_i = k$ is $P(w_i|z_i = k, z_{-i}, w_{-i})$. z_{-i} is all other topic assignment $z_{-i} \neq z_i$. w_{-i} is also all other word types $w_{-i} \neq w_i$. $\phi^{(k)}$ is the

multinomial distribution with topic $z_i = k$ over words. $|T|$ is the total number of topics as your predefined class number of the corpus.

5. R_{ij} is the *adaptive TFIDF-LDA weights* showing the weight of term *i* in a document *j*.

6. In Chapter 5, d_j is the documents notation, c_k denotes the predefined categories, w_i represents every term, $|V|$ is the total number of terms in all documents or the vocabulary size, $|C|$ is the total number of all classes.

W_{ij} is the count of term w_i in document d_j, where i is still the term index , j is the document index and k is the category index.

Chapter 2

Model Background

2.1 Motivation of the Method

Almost anybody can propose a method for analyzing data which is already in a good shape. How can we get access to the unstructured data, process it and make sense of it through statistics? Although many existing studies have reported promising performance of text mining method, only a few works are focused on the entire preprocessing training and classification methods for analyzing the vast unstructured scientific literature data. The text mining model in this paper will systematically put you on the map and pull your grades up.

Imagine that people are trying to make inference about 5,000 documents with the goal of retrieving all texts relevant to mathematics education. One can randomly pick up the representative small samples since normally we cannot manually process so many documents at once. But this sampling choosing process will ignore lots of paper and then loss major part of the story. If people have the opportunity to process 5,000 paper using computers to look for unseen patterns and associations across the millions of words in the articles, why they shouldn't process it using text mining application? Let the machine to learn and pull out data.

In this paper, an application can automatically read files, put the analyzable format then run the statistics. How can one take the stuff that people wouldn't be looking at? Too many synonyms, too many different format and sources of data, people can investigate it and make sense of it through this hybrid semi-supervised text classification approach. That is powerful and the newest direction since people usually does modeling on existing dataset but people can pick up their own unstructured data and make sense of it.

2.2 Machine Learning Methods: Unsupervised, Supervised, Semi-Supervised

As an important branch of artificial intelligence, machine learning is defined as the science of computer learning without being explicitly programmed by Arthur Samuel in 1959 [17]. Way back around 1998, Tom Mitchell presented a new formal definition of machine learning as a computer program which learns from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E [18].

This field of study can automatically let computer learn from data and is supposed to effectively save human intervention time. From spam detection, speech/face recognition, computer vision, gene discovery, medical diagnosis to self-driving cars [19], cutting edge machine learning applications are everywhere in real life and often without even being aware of it.

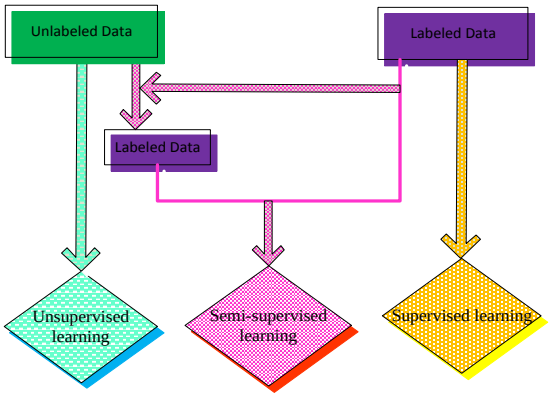


Figure 2.1: Interactivity of three types of machine learning.

In highly interdisciplinary machine learning field, there are three major techniques of supervised, unsupervised and reinforcement learning. The reinforcement learning, motivated by behaviorist psychology, is concerned with existence of optimal solutions to take actions in an environment in order to maximize the cumulative reward [20]. Supervised learning is to train completely labeled

data and unsupervised learning is to train unlabeled data. The semi-supervised learning used in this dissertation is a combination of supervised and unsupervised learning techniques that make use of both labeled and unlabeled data for training. The above Figure 2.1 illustrates the interactivity of these three different types of learning.

2.2.1 Unsupervised Learning

Unsupervised learning is a machine learning method to find hidden structures in the unlabeled data. Clustering algorithms is normally pointed as the part of unsupervised learning because there is actually no available information relating the membership of input unlabeled data to predefined classes [21]. The exploratory unsupervised algorithm has to experimentally exploit the statistical properties and structures in the distribution of the input unlabeled data without any guidance[19].

Unsupervised learning is widely used for many different problems since unlabeled data are easier to get and we can collect data that are known to belong to one of the classes without manually labeling it. There are a lot of real life applications of unsupervised learning such as web search, social network analysis and market segmentation [19]. Using unsupervised learning, a marketer can automatically divide the targeted marketing into different segments or separate diverse customers according to their hidden similarities.

However, unsupervised learning has both advantages and disadvantages. Although it is fairly easy to design a learning algorithm using cheap un-annotated training data, but it is very hard to tell whether the naturally discovered knowledge reports good patterns or trends. The generality, existence of many local maximums and lack of guided direction for the unsupervised learning algorithm limits its performance on some specific tasks. Thus, for achieving better performance

in some cases, the supervised learning turns out to be a popular alternative solution to unsupervised learning.

2.2.2 Supervised Learning

Supervised learning is a type of machine learning techniques that utilizes completely labeled data for training purpose. As students learn knowledge by examples from teachers and correct answers, supervised learning also extracts structures and regularities from labeled input data with pre-defined classes. Since more information is already contained by the annotated data examples, supervised learning can easily study a mapping between inputs and outputs based on a training data set to predict the outputs corresponding to test inputs [19].

Supervised learning is actually used for broad modern life applications such as spam detection, face detection, signature/character recognition, and medicine. For example, every time we send a mail via the post office, we are experiencing a supervised learning approach which can automatically recognize the handwritten zip code on the envelope [22].

The learning efficiency of supervised learning is relatively higher compare with unsupervised learning. However, obtaining fully labeled and sufficiently large training data needs a number of qualified experts for manually annotating the corpus and is usually expensive, time consuming, and error prone, which is the biggest bottleneck for supervised learning. The quality and amount of the labeled data undoubtedly limits the performance of supervised learning. So if you cannot afford the labeling costs required by supervised learning, semi-supervised learning approach, using both the unlabeled and labeled data, can substitute for unsupervised learning and supervised learning to generate considerable improvement in performance accuracy with less human effort.

2.2.3 Semi-Supervised Learning

As the size of online documents increases rapidly nowadays, to annotate a huge collection of corpus for supervised learning is almost infeasible. In reality, there is usually a huge amount of convenient unlabeled data but relatively limited labeled data. Semi-supervised learning, which is to learn with labeled in conjunction with unlabeled training data, takes the advantages of the strengths of both unsupervised learning method that discovers the underlying structure in the unlabeled data and supervised learning algorithm which employs sufficient labeled training data to learn classifiers [23] [24] [25] [26].

In the text processing practical application, semi-supervised learning reduces the high cost of obtaining labeled documents. In facial expression recognition, it is very tedious and difficult to manually label the video to the corresponding expressions. Semi-supervised learning provides useful assistance in the video indexing.

There are some commonly representative semi-supervised learning methods including Expectation-Maximization (EM) algorithm with generative mixture models, self-training, co-training, density-based transductive support vector machines (TSVM), boosting-based and graph-based methods [24]. As a compromise between the supervised learning and unsupervised learning, in this dissertation semi-supervised learning provides most cost effective solution in the document classification tasks.

2.3 Datasets and Protocol

A collection of large datasets will be used to conduct the simulation and experimental study for the document classification tasks in this dissertation. The proposed hybrid model and other benchmark methods will be applied to two real-life data sets primarily collected from the

publicly available online journal paper databases of the institute of electrical and electronics engineers (IEEE), which is the world's largest technical professional association. The popular benchmark text classification collection 20 Newsgroups data set, collected by Ken Lang in 1993, consisting of 18828 messages partitioned nearly evenly across 20 different UseNet discussion groups will be also utilized to compare the effectiveness of the proposed hybrid semi-supervised model [27] [28] [29].

Labeling data is normally expensive and time consuming in most text classification tasks especially the real world problems, unlabeled data are usually inexpensive, abundant and readily available. Therefore, with the aim of saving time and energy and simultaneously improving the classification performance, the semi-supervised machine learning method with both labeled and unlabeled data used for training will be applied in this dissertation.

For two real life journal paper datasets, the accuracy and F-measure of the proposed model will be estimated based on the average of results using the five-fold cross validation protocol. The five-fold cross validation will be also repeated four times on the same data set to assess the robustness of the algorithms as labeled training data instances are assigned differently at random to folds. All results will be reported as the averages over all trials of the experiments.

The following Table 2.1 tabulates a brief description of these three datasets. The total number of documents is indicated by “Total” and “Class” stands for the number of categories in each dataset. The amount of training data includes the count of both labeled and unlabeled training sets respectively.

Table 2.1: Data Descriptions.

Dataset	Source	Class	Total	Training		Test
				Labeled	Unlabeled	
5 class journal paper	Real life	5	710	(6%) 35	533	142
				(10%) 56	512	142
				(20%) 113	455	142
				(25%) 142	426	142
				(30%) 170	398	142
				(40%) 227	341	142
				(50%) 284	284	142
7 class journal paper	Real life	7	1133	(6%) 54	853	226
				(10%) 90	817	226
				(20%) 181	726	226
				(25%) 226	681	226
				(50%) 454	453	226
20 Newsgroups	UseNet	6	3000	(10%) 270	2430	300
				(20%) 540	2160	300
				(25%) 810	1890	300
				(50%) 1350	1350	300

To make the raw paper more valuable and easier for classification, all experiments of two real life journal paper datasets have considered only the short paragraph-sized abstracts of papers. Since the abstract sketches the detailed objective and summary of the main points of a paper. However, a complete long paper may contain extraneous items such as tables, images, mathematical equations and references, which add a lot of noise and may not be a good appropriate input for text classification tasks. Moreover, vector space model, which represents documents as a vector containing the frequency of how many times each term occurs in a document, is more effective for short documents [30].

2.3.1 Corpus 1: Five Classes Journal Papers

710 documents randomly accumulated mainly from the IEEE online journal papers databases will be used to analyze the experiments. These five categories of papers are respectively chosen from mathematics education (math), aerosol research (env), text mining (txt), nuclear research (nuc), and image processing (img) areas. This real life dataset is created with almost unequal numbers of documents per class (46 documents from mathematics education, 264 documents from aerosol research, 144 documents from text mining, 112 documents from nuclear research and 144 documents from image processing).

For this first journal paper datasets, different experiments will be conducted to see the impact of labeled and unlabeled training data in the semi-supervised learning environment. First of all, the training and testing data sets split is employed by randomly choosing approximately one-fifth of the original data sets into a test set, including a total of 142 manually labeled papers, and the remaining 568 papers as a training set of labeled and unlabeled documents. Then the 568 training documents are randomly split into two groups: labeled and unlabeled training sets. Seven different labeled data rates vary from 6%, 10%, 20%, 25%, 30%, and 40% to 50% of all training number are used to check the sensitivity and performance of the proposed model. For example, as shown in the Table 2.1, in the first experiment, 35 of the 568 training data or approximately 6% of the total training data are randomly chosen as labeled sets and the rest 533 documents are retained as unlabeled data sets.

2.3.2 Corpus 2: Seven Classes Journal Papers

The second real world dataset includes a total of 1133 documents randomly extracted from the IEEE online journal databases. There are seven categories of papers respectively chosen from

mathematics education (math), aerosol research (env), text mining (txt), nuclear research (nuc), image processing (img), transportation (tra) and robotics (rob) fields. To keep the comparison fair, this dataset also has the same unbalanced amount of documents per class (46 documents from mathematics education, 264 documents from aerosol research, 144 documents from text mining, and 112 documents from nuclear research, 144 documents from image processing, 216 documents from transportation, and 207 documents from robotics).

The training and testing data sets consist of 907 and 242 documents, respectively in the experiments. Five experiments will be conducted to investigate the performance of different algorithms as the labeling rate increases from 6%, 10%, 20%, 25% to 50%. For example, as listed in the Table 2.1, in the first experiment, 54 of the 907 training data or approximately 6% of the total training data are randomly selected using SAS surveyselect procedure as labeled sets and the remaining 853 documents as unlabeled data sets. The average results obtained from 20 runs of cross validation for each experiment will be reported in details.

2.3.3 Corpus 3: 20 Newsgroups

Collected from UseNet postings over several months in 1993 by Ken Lang for his paper, 20 Newsgroups data set has been a popular benchmark dataset in the large scale text classification fields [31] [27]. A total of 18828 messages collected from web are partitioned into 20 different Usenet newsgroups from computer, sports to politics related topics. For testing the performance of proposed hybrid model, multiple document classification experiments are conducted using 20 Newsgroups corpus in the Chapter 6.

In the experiments, 3000 newsgroup postings from 6 categories are drawn at random from the 20 Newsgroups data. This dataset has the same balanced amount of documents per class including 500 documents each randomly extracted from comp.graphics, comp.os.ms-windows.misc,

comp.windows.x, misc.forsale, rec.sport.hockey, and sci.crypt categories. We can find that many of the categories fall into the confusable cross classes such as three of them are computer related discussion groups.

A total of 2700 documents are used for training purpose, while 300 test sets are retained for evaluating predictive classification accuracy. A labeled training set is built by randomly selecting 10%, 20%, 30% and 50% documents respectively from those 2700 training documents. The remaining documents are formed as unlabeled training datasets. Preprocessing the 20 Newsgroups data in the proposed text filtering manner resulted in a vocabulary of unique terms. The accuracy and F-measure of classifying the new testing article into the predefined newsgroup to which it was posted are estimated and reported as the averages over all trials of the experiments.

2.4 Experiment Frame and Hybrid Model

What is the goal of text classification for this dissertation? To make a long story short, from a small set of “known” labeled papers, you can make predictions about new “unknown” unclassified papers into predefined categories. The hybrid semi-supervised text classification approach consisting of Porter stemming, TFIDF-LDA weighting, Zipf's law, Naive Bayes classifier, and Expectation–maximization algorithm is implemented for classifying given literature.

The first and foremost step I will take should be text pre-processing, which is a useful technique to represent the input text files of each document as a feature vector. A frequency of character string in the text pre-processing phase always gives us a lot of interesting information. For example, as one of the most artistic valuable Chinese ancient literatures, the writing of Song

poetry is always a big challenge since a lot of basic knowledge is needed. After the words frequency analysis for the Chinese Song poetry corpus, result shows high frequency content words “east wind, where, human world, ...” Therefore, using the statistical words counting information, we can easily write a new Song poetry in just three seconds, “In the human world, where is the east wind.”

How do we count the words information? The following process is used for text preprocessing and feature extraction in the hybrid model. The flow chart Figure 2.2 illustrates this experiment frame and hybrid model.

1. Convert “pdf” files to plain text: Use the SAS TGFILTER procedure to read and extract plain text from a collection of your own papers, which can be stored in various formats, such as Word or PDF.

2. Abstracts extraction: Since analyzing only the abstracts gives the best journal paper classification results. The abstracts will be then pulled out from the text of all journal papers in the document collection.

3. Term filtering: After transforming all characters to lowercase, the common English stopwords typically 400-600 words, punctuations and the words whose length less than three and larger than sixteen will be removed. The words are excluded for being too short, too long or too common.

4. Porter stemming (Lemmatization): In order to further reduce the word space dimensionality, word suffices are removed by the most popular porter stemming algorithm.

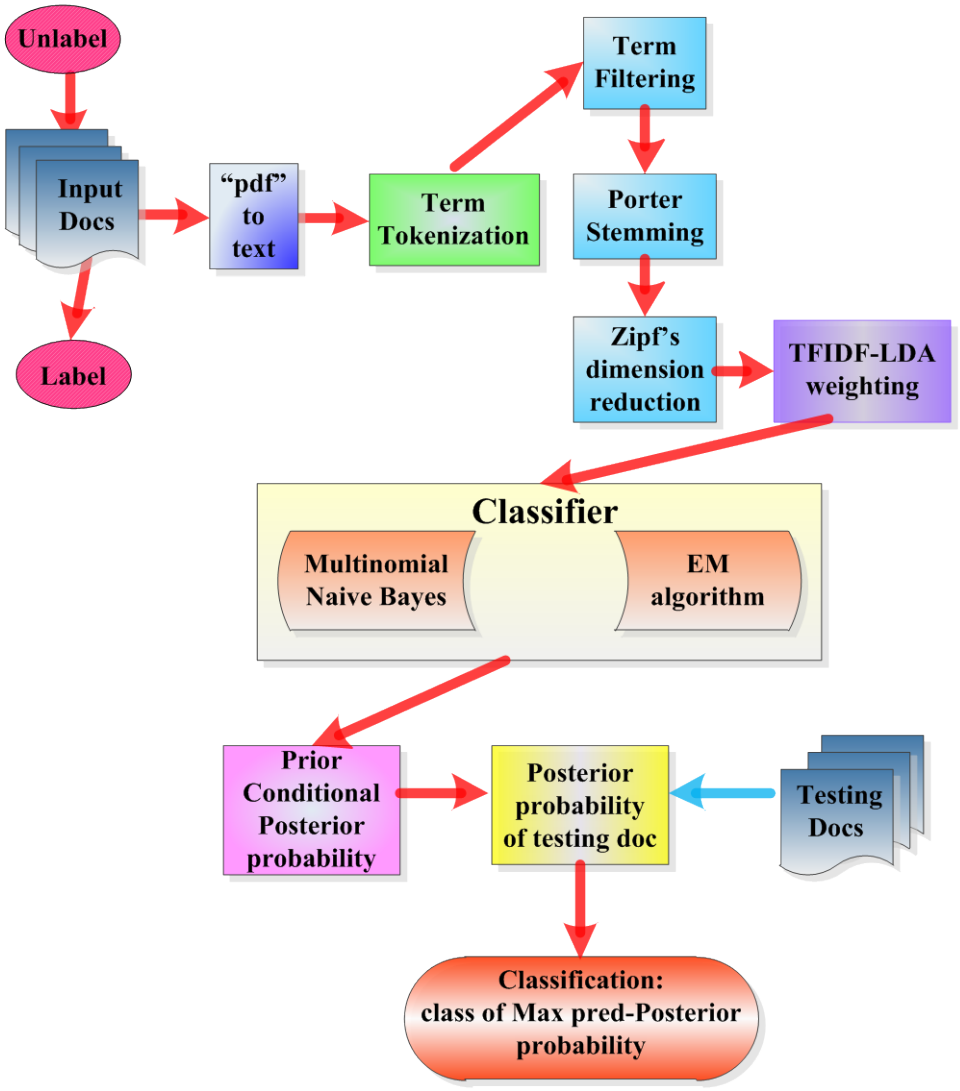


Figure 2.2: Experiment frame and hybrid model.

5. Tokenization: Word token is defined as the occurrences of a word. After the porter stemming matlab procedure, the papers are imported from excel format into SAS data set and then parse the text in these papers into separate words and count the frequency and statistics in order to perform the dimension reduction techniques.

6. Zipf's law based dimension reduction: According to the Zipf's law in linguistics, there is a very small subset of words that occur very frequently and almost all words are rare. If the

occurrence of words in all papers is relatively low, deletion of such rare words is therefore out of the question.

7. TFIDF-LDA weighting: Text classification is based on the accurate information of vector space model. The brand new TFIDF-LDA weighting method is proposed to describe documents in the vector space model.

Chapter 3

Text Preprocessing

3.1 Overview

In the text preprocessing, the key question is how to process unstructured textual information and extract meaningful numeric indices from the text. There are many special techniques for pre-processing text documents to make them suitable for mining. To evaluate the journal papers, this hybrid model will first parse the text of abstracts in these documents into separate words, perform following preprocessing dimension reduction techniques and use the resulting information from the dimension reduction to significantly improve the classification accuracy of the documents.

3.2 Term Filtering

3.2.1 Stopwords & punctuations removal

The stopword list used in this paper has about 575 most frequent words that often do not carry much meaning. Since including non-informative words will dilute our analysis, the data should be as clean and consistent as possible. After removing stopword list, a simple collection of low-information or extraneous words that you want to remove from the text such as a, an, the, be, with, by, etc., we can create a crucial start for obtaining valid and useful results. Moreover, synonym list can also be used to improve the quality of the text mining output, but creating the synonym list is a very labor intensive and time-consuming process. Normally, a change in the stopword list and synonym lists can dramatically alter the term weightings. Sometimes it is a bit hard to tune. The porter stemming algorithm treats words with the same stem as synonyms and you can use it as a substitute to synonym list. So by considering the costs and accuracy, I will

not devote a large amount of effort to create good stopword and synonym lists and only throw away all punctuations and 575 stopwords.

3.2.2 Excluding too short & long strings

The next factor I have to consider is a length of the character string. Short strings like “us” express a little useful information and meanwhile undesirably long and redundant strings are usually expected to have low frequencies. According to several experiments on given dataset, excluding too long and too short strings can highly reduce dimension and clean text. Thus, after removing all symbols but only letters and digits, the strings whose length are less than three and larger than sixteen will be excluded.

3.3 Porter stemming algorithm

English words like “work” can be inflected with a morphological suffix to produce “works, working, worked” which share the same stem “work”. The porter stemmer [32] has five steps to progressively strip the suffixes as “s, es, ed, ing, al, er, ic, able, ment, ive, etc.” for short and long stems. The porter stemming algorithm applied in this research is usually but not always perfectly useful to map all inflected forms into the right stem, since language is highly ambiguous and this complex stemming algorithm process may have some exceptional cases and mistakes such as that it replaces the word “department” to “depart”.

In addition, some special word like “kept” has “kept” as its lemma. However, using the porter stemmer, their link will be missed and the stemmed word for original word “kept” is still “kept”. Since the porter stemmer operates on a single word without knowledge of the context, and we cannot distinguish the words of different meanings solely depending on the part of speech. It requires lemmatization, a dictionary look-up process, which can essentially select the appropriate

lemma depending on the context to solve this issue. But the porter stemmer is normally run faster and easier to implement. Therefore, the reduced accuracy without lemmatization in this research may not matter too much.

3.4 Zipf's law based dimension reduction

Named after the linguist George Kingsley Zipf, Zipf's Law [33] [34] discovered the law about the distribution of words: the frequency of any word is inversely proportional to its rank in the frequency table. Frequencies of words used in everyday English follow the Zipf's distribution showing that a very small subset of words occurs very frequently as "common words" and a large subset of words are rare as "rare words". Zipf's law points out an important empirical observation that almost all words are rare. This heavy tail property motivated the following dimension reduction research.

Most terms occur only in a very limited number of documents, which makes them interesting candidates for our dimensionality reduction purpose [35] [9]. The elimination threshold occurrence of rare word was experimentally determined to be around the range from 1% to 2% of all documents in my experiments. The multiple experiments on the first journal paper corpus show an empirical fact that the rare terms appearing only three or fewer in the entire collection of documents after the porter stemming can be removed in order to reduce the term space dimensionality and improve classification accuracy.

Chapter 4

Feature Extraction-Applications of TFIDF-LDA Weighting

4.1 Overview

How can one extract the best terms that are able to discriminate certain individual documents from the remainder of the collection? Feature extraction is exactly a special form of dimensionality reduction. In test mining, the vector space model (VSM) is widely used for representing textual documents. Given a massive document collection, a set of terms or features can be defined. Then, by putting associated weighting value of each word in documents, a high dimensional and sparsely filled vector is generated from this feature set. Usually, the raw term frequencies can be weighted to find the best terms for document content identification, using one of the following available measures, such as TF-IDF, Entropy, Normal, Chi-Square, etc. The proper weighting of the set of features that represents the text can improve the performance of the model. In this dissertation, a new weighting method based on the term frequency-inverse document frequency and statistical estimation of the word importance for topic model LDA is proposed.

4.2 TF-IDF Weighting

Documents are represented as a vector in the vector space model. Each vector component of a document is related to a particular word in the term vocabulary lists. A numerical value is normally assigned to every component of a document vector to estimate the importance or weight of the word in the document. From old to present, there are many studies relate to this problem of quantifying the significance of terms in the documents. In 1957, Luhn [36] first discovered the measurement of term significance, Term Frequency (TF), the number of

occurrences of terms in a document. In 1972, Spark-Jones [37] found another useful measurement Inverse Document Frequency (IDF), the number of occurrences of terms over the documents. In 1983, Salton and McGill [38] represented the most famous term frequency-inverse document frequency (TFIDF) weighting, the production of TF and IDF, for detecting the importance of terms in the documents.

As a benchmark weighting method, TF-IDF weighting [39] [40], is a statistical measure used to evaluate how important a word is to a document in a corpus. The best terms normally have high term frequencies but low overall collection frequencies. Therefore, TF-IDF weighting assumes that the importance of words increases proportionally to the occurrence of a word in the document but is offset by the frequency of the word in the corpus [41]. The definition of the TF-IDF weighting function is the multiplication of two statistics, term frequency and inverse document frequency as follows,

$$TFIDF = f_{ij} \cdot log\left(\frac{|D|}{df_i}\right) \tag{4.2.1}$$

where f_{ij} is the raw frequency of a term i in a document j, i.e. the number of times that term i occurs in document j; The term or word type is defined as the unique word as a dictionary entry.

$|D|$ is the total number of documents in the collection;

df_i is the number of documents in the corpus that contain term i.

4.2.1 Advanced TF-IDF Weighting

The raw frequency sometimes may result in a bias towards longer documents. Therefore, normalized frequency, raw frequency divided by the total number of words in a document, is applied to prevent this issue. As a penalizing factor, the denominator of inverse document frequency can also be improved as the total number of times that a term occurs in all documents

to offset those terms too frequent. Thus the TF-IDF weighting in (4.2.1) can be modified as the following new advanced TF-IDF weighting (Adv_TFIDF) in my experiments:

$$Adv\_TFIDF = \frac{f_{ij}}{|V_j|} \cdot \left(1 + log\left(\frac{|D|}{\sum_{j=1}^{|D|} f_{ij}}\right)\right) \quad (4.2.2)$$

Where f_{ij} is the raw frequency of a term *i* in a document *j*;

$|V_j|$ is the total number of words in a document *j*;

$|D|$ is the total number of documents in the collection;

$\sum_{j=1}^{|D|} f_{ij}$ is the total number of times that a term *i* occurs in all documents, *i* is the term index and *j* is the document index.

4.3 Probabilistic Topic Model-Latent Dirichlet Allocation (LDA)

Probabilistic topic models have been successfully applied to many text mining tasks such as information retrieval, summarization, categorization, and clustering. As an updated most popular representative probabilistic topic model, Latent Dirichlet Allocation (LDA) has attracted intense research attention in recent years. In this dissertation, LDA model is used to mine large collections of documents and summarizes those using topics.

What is Latent Dirichlet Allocation?

LDA is a generative model that expresses the distributed probability of words and explains sets of observations by unobserved groups which show why some parts of the data are similar [42]. If observations are words collected into documents, LDA supposes that every document is a mixture of a small number of latent topics and these document's topics create every word [43]. Based on seminal work in latent semantic indexing (LSI) (Deerwester et al., 1990) [44] and Probabilistic Latent Semantic Indexing (pLSI) (Hofmann, 1999) [45], LDA model was first

proposed by David Blei, Andrew Ng, and Michael Jordan [42] in 2002 and then was reintroduced as a graphical model in the paper "Latent Dirichlet allocation" in 2003.

What are the benefits of LDA approach?

Based upon singular value matrix decomposition to deal with polysemy (words with multiple meanings) and synonymy (multiple words with the same meaning), LSI model does not have satisfactory statistical foundation. As a probabilistic topic approach to document modeling, the pLSI model is difficult to test the generalizability of the model to new documents, since there are not any assumptions about how the mixture weights θ are generated [45]. As an extended pLSI model, Latent Dirichlet Allocation (LDA) introduced a Dirichlet prior on θ, regularized and dealing directly with probabilities that works better for discrete data.

What are the LDA model applications in this dissertation?

LDA model is applied to several huge document collections which can get much easier to navigate using topics. Since there are far less topics than words and the topics usually retain a lot of the meaning of the documents. Like principal component analysis (PCA) and LSA, LDA improves the word features to topic features which highly reduce the dimensionality of the documents in the corpus.

4.3.1 LDA Model

In LDA model, every document is a random mixture over latent topics and every topic is an uncorrelated and discrete distribution over words from finite dictionary. A discrete latent model is assigned to words and each document maintains a random variable, indicating its probabilities of belonging to each topic.

The generative model LDA shows how every document obtains its words in a corpus *O* of a collection of $|D|$ documents. As the bag-of-words assumption in document classification, the order of the words in the documents is ignored. A topic is defined as a multinomial probability distribution over a collection of words. Each document includes terms from multiple topics and contains topics in distinct proportion comparing with other different documents. A *K* topic multinomial distribution with parameter vector ϕ of length $|V|$, which is the total number of terms in all documents, is assumed first. Then for each word in the document we draw one of the *K* topics from the assumed topic multinomial distribution and probabilistically choose a word from $|V|$ vocabulary according to parameter vector ϕ at random. For the purpose of automatically discovering the topics from given corpus, the *K* topic distributions can be learned through statistical inference.

Given the multinomial and Dirichlet probability distributions, LDA model assumes the following three generative process for each document containing *N* words in a corpus *O* [42] [46]:

1. Select the total number of words *N* for a document *w* as $N \sim$ Poisson (ξ).

2. Randomly draw a topic distribution $\theta \sim$ Dir(α). The proportions of the topic distribution for the document *w* are determined by a *k* dimensional Dirichlet random variable θ. α is the Dirichlet prior over the documents and is usually set as a fixed scalar hyperparameter in the LDA programming process although maximum likelihood fitting can find its optimal value.

The Dirichlet distribution, the exponential family distribution over the simplex of positive vectors that sum to one, is conjugate to the multinomial distribution and has the following probability density [41]:

$$P(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k} \alpha_i)}{\prod_{i=1}^{k} \Gamma(\alpha_i)} \prod\nolimits_{i=1}^{k} \theta_i^{\alpha_i - 1}$$

where the scaling parameter α is a positive k-vector with components $\alpha_i > 0$, and $\Gamma(x)$ denotes the Gamma function.

3. Next, a topic z_i belonged to multinomial (θ) distribution is selected for every N words w_i. Moreover, for every word in the document, a word w_i from $P(w_i|z_i)$, a multinomial probability conditioned on the topic vector $z_{i,}$, is then chosen at random. β is a $k \times V$ word-probability matrix for each topic as row and every term as column, which records the generated w_i word probabilities given the certain topic z_i.

The joint distribution of a topic mixture θ, N topics z, and N words w given the predefined constant parameters α and β, is defined as follows [41]:

$$P(\theta, z, w|\alpha, \beta) = P(\theta|\alpha) \prod_{i=1}^{N} P(z_i = k|\theta) P(w_i|z_i = k, \beta) \quad (4.3.1)$$

The probability of word w_i equals to the following summation of the multiplication of multinomial distribution over T topics and the conditional probability of a word w_i given a topic $z_i = k$ [47]:

$$P(w_i) = \sum_{k=1} P(z_i = k) P(w_i|z_i = k)$$

In this predictive multinomial Dirichlet distribution model, according to the paper *"Gibbs sampling in the generative model of Latent Dirichlet Allocation"* written by Tom Griffiths in 2002, the conditional probability of a word w_i given topic z_i is derived as follows [47] :

$$
\begin{aligned}
&P(w_i|z_i = k, z_{-i}, w_{-i}) \\
&= \int P\left(w_i|z_i = k, \phi^{(k)}\right) P\left(\phi^{(k)}|z_{-i}, w_{-i}\right) d\phi^{(k)} \\
&= \frac{\beta + n_{-i,k}^{(w_i)}}{|V|\beta + n_{-i,k}^{(\cdot)}}
\end{aligned}
\tag{4.3.2}
$$

where z_{-i} is all other topic assignment $z_{-i} \neq z_i$. w_{-i} is also all other word types $w_{-i} \neq w_i$. $\phi^{(k)}$ is the multinomial distribution with topic $z_i = k$ over words. $n_{-i,k}^{(w_i)}$ is the number of words assigned to topic *k* without word w_i. $n_{-i,k}^{(\cdot)}$ is the sum of all words assigned to topic *k* except the current word w_i [47]. β is a smooth hyperparameter. $|V|$ is the size of vocabulary.

In LDA model, the computation of the posterior distribution of the latent variables given a document is the most crucial and intractable inferential issue since the integral in the denominator $P(w|\alpha, \beta)$ which is difficult because of the coupling between θ and β parameters in the summation over hidden topics [41]. Thus, the approximate inference algorithms should be considered for LDA to approximate the posterior. By trading off speed, complexity, accuracy, and conceptual simplicity, the Gibbs sampling algorithm is used to estimate the parameters of topic-word distributions ϕ and topic distributions θ. Gibbs sampling, a form of Markov chain Monte Carlo which refers to a set of approximate iterative techniques designed to sample values from complex high-dimensional distributions, simulates a high-dimensional distribution by sampling on lower-dimensional subsets of variables where each subset is conditioned on the value of all others. For extracting a set of topics from a large corpus, Gibbs sampling is iteratively done and proceeds until the sampled values approximate the target distribution.

4.4 Adaptive TFIDF-LDA Weighting

In the information retrieval field, TF-IDF weighting is very appropriate to represent the usefulness of terms in the retrieval process. It is commonly very useful to weigh each word in the text document according to how unique it is. But the performance of the conventional TF-IDF approach on the text classification problem is not very clear since TF-IDF weighting method captures the relevancy among words, text documents and does not leverage the information implicitly contained in the classification job to represent documents [39]. For enhancing the TF-IDF weighting on the text classification, a new adaptive TFIDF-LDA weighting approach, which combines the word significance within documents and the estimation of a word importance for a particular classification task based on the LDA model, is created to improve the recall and precision of text classification.

The generative LDA model assumes that each document is generated by a mixture of topical multinomial. A document is first generated by selecting a multinomial over topics z_n given the Dirichlet prior α. Moreover, a topic for each term in the document is created from the document-specific topic distribution. Then generate each word in the document by the discrete distribution for that topic. After repeating the Gibbs sampling machine learning steps, eventually the topic mixtures of each document is estimated by counting the proportion of words assigned to each topic within that document and the topic-word probabilities of generated word w_n probabilities given certain topic z_n are calculated by counting the proportion of words assigned to each topic. The topic-word probabilities from LDA model output inspired me to infer the LDA weighting, which shows the importance of each word in all topics or classes.

Running the LDA Gibbs sampler on given documents, a set of topics and the probability of each word per topic can be extracted. In this paper, I will introduce a new weighting method, adaptive

TFIDF-LDA weighting, based on advanced TF-IDF weighting and LDA Weighting which is associated with the statistical estimation of the probability of each word per topic by LDA Gibbs sampler.

How to derive the adaptive TFIDF-LDA weighting formula?

For the LDA model, the input is a bag of word representation containing the number of times each words occurs in a document. After converting text file into proper LDA input data format, the Matlab Topic Modeling Toolbox 1.4 released by Mark Steyvers and Tom Griffiths [48] will be implemented to extract topics with LDA model. The following steps illustrate the extraction of the probability of each word per topic.

First, we have to compile the term dictionary, which lists the unique words in your corpus, in order to the future writing of the resulting word-topic distributions.

Then, the word indices in the dictionary and the document index of each word should be created.

Moreover, set the number of topics as your predefined class number of the corpus, the hyper parameters β as 0.01 and α as 50 divided by the number of topics. After 1000 times LDA Gibbs sampler iterations, you can get the probability list of each term per topic.

Except the LDA model, the adaptive TFIDF-LDA weighting formula developed in this dissertation is also motivated by the basic theoretical formulae of information theory. As a branch of applied mathematics, electrical engineering, and computer science involving the quantification of information, information theory was first developed by Claude E. Shannon 1948 [49] to find fundamental limits on signal processing operations including compressing data and on reliably storing and communicating data. Information theory is already applied and broadened to many different research areas such as statistical inference and natural language

processing. In this paper, the concept of entropy, a key notion of information, is used to quantify a measure of information how important each word in all topics should be.

As the definition represented in the textbook of *Elements of information theory* by Thomas M. Cover 1991 [50], the concept of entropy is a measure of uncertainty of a random variable. Let x_i and y_j be random variables representing two distinct events from finite event spaces X and Y. Given $x_i \in X$ and $y_i \in Y$, the marginal distribution of a joint probability distribution $P(x_i, y_i)$ is defined as:

$$P(x_i) = \sum_{y_i \in Y} P(x_i, y_i)$$

The amount of information, a basic quantity in information theory, is expressed as the log of the inverse of the probability [51]:

$$log\left(\frac{1}{P(x_i)}\right) = -log\, P(x_i)$$

The entropy $H(X)$ of a discrete random variable X, which expresses the degree of uncertainty about which an event will occur in a future observation, is then defined by [48]:

$$H(X) = -\sum_{x_i \in X} P(x_i) log P(x_i) \qquad (4.4.1)$$

The base of logarithm is 2 and entropy is expressed in bits.

Based on the above conventional information theory definitions in (4.4.1) and the word-topic distributions from LDA modeling in (4.3.2), the importance of each word in all topics is computed by a new LDA weighting as follows:

$$LDA_{Weighting} = -\sum_{k=1}^{|T|}(P(w_i|z_i = k) + \mu)\log(P(w_i|z_i = k) + \mu) = -\sum_{k=1}^{|T|} p_{ik} log p_{ik} \quad (4.4.2)$$

where $p_{ik} = P(w_i|z_i = k) + \mu$ is the probability of term i in a topic k. The smoothing parameter μ $(0 < \mu < 0.1)$ will be added into the formula for avoiding the zero probability estimation and $p_{ik} > 0$;

$|T|$ is the total number of topics as your predefined class number of the corpus.

Therefore, by integrating the word significance within documents illustrated by TF-IDF and the estimation of a word importance within classes by LDA modeling, a new weighting method, adaptive TFIDF-LDA weighting can be formulated according to the equations (4.2.2) and (4.4.2) as follows:

$$R_{ij} = \frac{f_{ij}}{|V_j|} \cdot \left(1 + log\left(\frac{|D|}{\sum_{j=1}^{|D|} f_{ij}}\right)\right) - \frac{1}{|T|}\sum_{k=1}^{|T|} p_{ik} log p_{ik} \quad (4.4.3)$$

where R_{ij} is the *adaptive TFIDF-LDA weights* showing the weight of term *i* in a document *j*;

k is the topic index.

Then we can apply a weights cutoff to exclude the terms with relatively small adaptive TFIDF-LDA weights in each document. The cutoff is usually determined empirically by your corpus from 2% to 3% or more. The experiment results in this paper show that the best classification accuracy was achieved by using cutoff around 3%. This adaptive TFIDF-LDA weighting method has the benefits to greatly reduce the feature dimension and meanwhile make feature selection robust.

To evaluate the performance of the feature selection by TFIDF-LDA weighting approaches, the commonly used F-measure metric, which is equal to the harmonic mean of recall and precision, is used in this paper. F-measure defined in the following Chapter 5.4 is widely used to evaluate text classification system because of the trade-off between recall and precision measures.

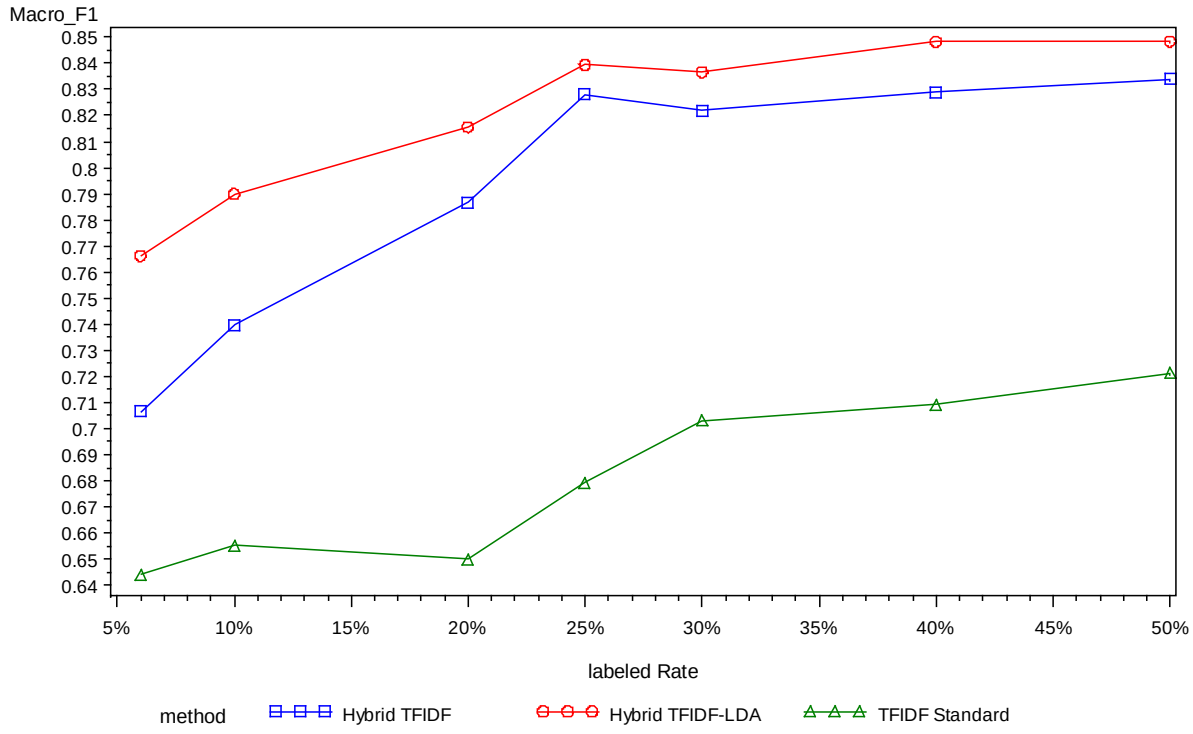


Figure 4.1: Comparison of TFIDF-LDA, Hybrid TFIDF and TFIDF Standard methods on first 710 journal paper dataset as the number of labeled documents increases.

Above Figure 4.1 shows the comparison of TFIDF-LDA, hybrid TFIDF and TFIDF Standard weighting methods on first five-class real life journal paper datasets, when we use the same semi-supervised training methods on these three solutions. The hybrid TFIDF method uses the exactly identical algorithms to preprocess the documents as the proposed TFIDF-LDA weights. As the number of labeled documents increases, the macro-F1 value of TFIDF-LDA weighing is higher than the hybrid TFIDF weighting and significantly greater than the conventional TFIDF

weighting. Moreover, there is much less features in the feature space of TFIDF-LDA weighting than other two methods associated with basic TFIDF weighting. More extensive experiments reported in Chapter 5 show that this new weighting method improves the classification accuracy and meanwhile reduce the feature dimension.

Chapter 5

Semi-supervised Naive Bayes Training & Classification

5.1 Mixture model frame

The training data set is already prepared through previous feature extraction process. Then, in the next step, how to classify new documents into the predefined categories? For computing the probabilities of these documents belonging to each category, in this paper, a naive Bayes classifier based semi-supervised learning [52] will be conducted on the labeled and unlabeled training text data. The following mixture model assumes that word distributions in documents are generated by a specific parametric model and the parameters can be estimated from the training text data [53].

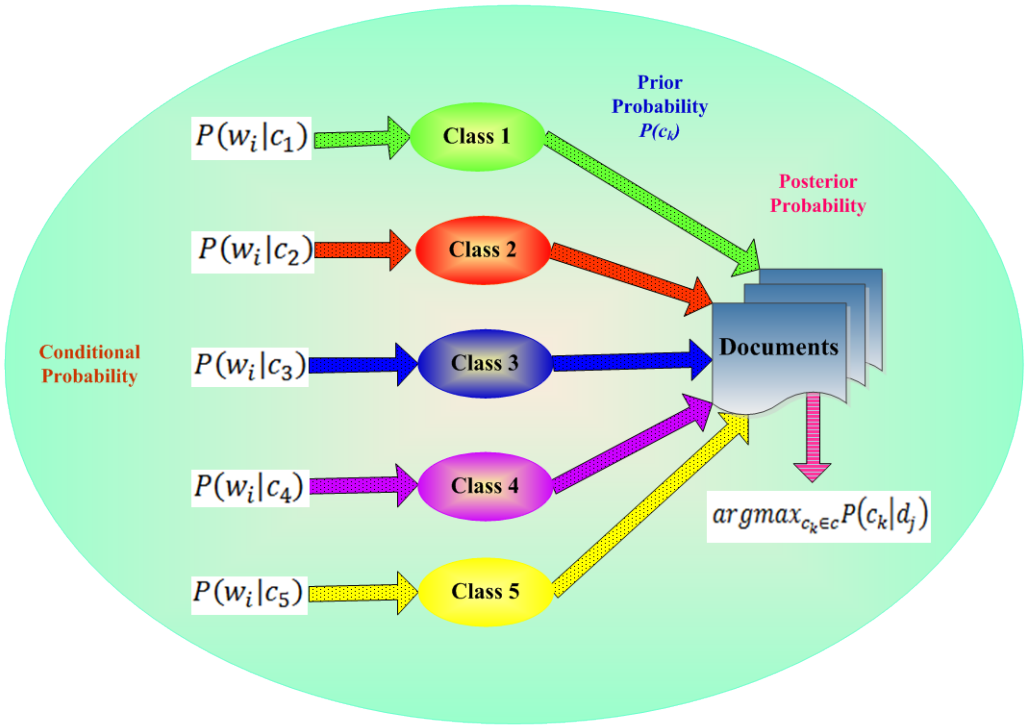


Figure 5.1: Mixture probability model frame.

5.2 Naive Bayes Classifier

5.2.1 Introduction

As a probabilistic classifier based on Bayesian theorem, naive Bayes classifier [54] is normally trained very efficiently in a supervised learning setting. The word naive in its name derives from the fact that the algorithm uses Bayesian techniques but does not take into account dependencies that may exist. With strong independence assumption, naive Bayes classifier assumes that given the class variable the presence of a particular feature of a class is unrelated to the presence of any other feature [29].

Naive Bayes classifier [55] can often outperform more sophisticated classification methods with less effort, since its general-purpose, simple to implement, suited when the dimensionality of the inputs is high, and particularly it only requires a small amount of training data to estimate the necessary parameters for classification.

5.2.2 Multinomial Naive Bayes Classifier

Suppose that each document is assumed to be generated by the multinomial distribution, d_j will be the documents notation, c_k denotes the predefined categories, w_i represents every term, $|V|$ is the total number of terms in all documents or the vocabulary size, $|C|$ is the total number of all classes and the count of term w_i in document d_j is W_{ij}. i is still the term index , j is the document index and k is the category index [56].

The conditional probability $P(w_i|c_k)$ of term w_i in given class c_k equals to the count of term w_i in class c_k divided by the total number of words in class c_k. If the training document d_j is in the class c_k, the probability $P(c_k|d_j) = 1$, otherwise $P(c_k|d_j) = 0$. The smoothing parameter λ will be added into the formula of for the following conditional probability $P(w_i|c_k)$ so as to avoid the zero probability estimation ($0 < \lambda < 1$).

$condi\_prob$:

$$P(w_i|c_k) = \frac{\lambda + condi\_term\_num}{\lambda|V| + term\_class\_num} = \frac{\lambda + \sum_{j=1}^{|D|} W_{ij}\, P(c_k|d_j)}{\lambda|V| + \sum_{i=1}^{|V|}\sum_{j=1}^{|D|} W_{ij}\, P(c_k|d_j)} \qquad (5.2.1)$$

condi_term_num is the amount of every term in each class;

term_class_num is the total number of terms in each class.

For example, the term "student", its *condi_term_num* is respectively 9 in class 1 and 4 in class 3. The total number of all terms $|V|$ is 661. *term_class_num*, the total number of terms in class 1 and 4 is separately 53 and 76. Thus, the conditional probability of the term "student" in class 1 equals to $\_prob = \frac{0.1+9}{0.1\times 661+53} = 0.0764$.

The prior probability of each class is equal to the number of documents in this class c_k divided by the total number of documents $|D|$.

$prior\_prob$:

$$P(c_k) = \frac{number\ of\ docs\ in\ \ class\ c_k}{total\ number\ of\ documents} = \frac{\sum_{j=1}^{|D|} P(c_k|d_j)}{|D|} \qquad (5.2.2)$$

According to the multinomial naive Bayes model, we can get the posterior probability of every document as follows:

$post\_prob$:

$$P(c_k|d_j) = \frac{P(d_j|c_k)P(c_k)}{P(d_j)} = \frac{P(c_k)\prod_{i=1}^{|d_j|} P\left(w_{d_j,i}\middle|c_k\right)}{\sum_{k=1}^{|C|} P(c_k)\prod_{i=1}^{|d_j|} P\left(w_{d_j,i}\middle|c_k\right)} \qquad (5.2.3)$$

where $w_{d_j,i}$ is the term w_i in document d_j;

$P\left(w_{d_j,i}\middle|c_k\right)$ represents the conditional probability that a term w_i will appear in document d_j given the class c_k;

$|d_j|$ is the total number of unique terms in the document d_i.

The posterior probability $P(c_k|d_j)$ tells us the probabilities of testing document d_j belonging to every category with the class label c_k. The category of the largest posterior probability $argmax_{c_k \in c} P(c_k|d_j)$ will be the predicted class of the testing documents.

5.3 Naive Bayes+EM Algorithm on Labeled and Unlabeled Training Data

Naive Bayes classifier usually works well in the standard supervised learning environment with the limited number of labeled training data. Due to the high cost of obtaining real world labeled documents, in this dissertation, naive Bayes classifier [57] [58] and Expectation-Maximization (EM) [59] [60] algorithm, an iterative method for finding maximum likelihood estimates of parameters in statistical models, are used together in a semi-supervised setting on both labeled and unlabeled training datasets [61].

The basic EM algorithm for semi-supervised learning of a text classifier has the following steps. Firstly, a classifier is initially built from the given labeled documents. Then, EM algorithm successively runs two major procedures, the E-Step (Expectation) and M-step (Maximization). After the looped modifications of parameters, the improved naive Bayes classifier is employed to estimate the probabilities of every unlabeled document associated with each class. This is the E-Step, first executed for each iteration, which estimates the posterior probability of unlabeled documents belong to each class according to the given posterior probability of labeled documents. The next following M-step re-estimate the parameter vector of the probability distribution of each class. A new naive Bayes classifier is iteratively reconstructed using all the mixtures of labeled and unlabeled training sets. The algorithm finishes when the distribution

parameters converge or reach the maximum number of iterations [62]. Then, the final tuned classifier can be used to predict the unlabeled documents in the test sets into a predefined class.

The SAS programming based on the naive Bayes classifier and EM algorithm [63] was created to train labeled and unlabeled training documents and classify the testing documents in my experiments. To formalize the semi-supervised classification with the naive Bayes classifier and EM algorithm, comparing with the previous description, the generative process restated in more detail as follows.

1. First of all, since the documents were pre-classified as different classes, the initialization of prior probability of each class $P(c_k)$ is calculated as the division of number of documents in current class c_k and the total number of documents as defined in equation (5.2.2).

2. Then, the initialized conditional probability of each term $P(w_i|c_k)$ is estimated using equation (5.2.1) with smoothing parameter λ =0.2.

3. Moreover, according to the initialized conditional probability and prior probability of each class, create and initialize posterior probability of every document as follows.

$$post\_prob\_initial = \frac{exp(log(prior\_prob) + \sum log(condi\_prob))}{\sum(exp(log(prior\_prob) + \sum log(condi\_prob)))}$$

4. Next, we will calculate the iterative prior probability which is changed iteratively at each convergence of the EM algorithm. The iterative prior probability is computed using the following equation

$$\begin{aligned} prior\_prob2 &= \frac{\sum post\_\text{prob}\_initial}{\sum\sum post\_\text{prob}\_initial} \\ &= \frac{sum\ of\ post\ probs\ of\ every\ class}{total\ sum\ of\ full\ post\ probs\ for\ all\ classes} \end{aligned} \tag{5.3.1}$$

5. Whereafter, we calculate the iterative conditional probability.

$$condi\_prob\_initial = \frac{\lambda + weight * condi\_term\_num * post\_prob\_initial}{\lambda|V| + \sum(weight * condi\_term\_num * post\_prob\_initial)}$$

Weight is a factor used for the leverage of the large unlabeled training documents since several experiments indicate that unlabeled data are quite often detrimental to the performance of classifiers.

When document $d_j \in$ unlabeled training documents, then weight=0.7;

Otherwise, document $d_j \in$ labeled training documents, then weight=1.

6. We can find the difference between two maximum number of prior probabilities *prior_prob* and the iterative prior probability *prior_prob2* estimated in (5.3.1). If this difference less than your default threshold such as$|(max(prior_prob) - max(prior_prob2)| < 0.001$, then the algorithms converges to a stable classifier. This process of classifying the unlabeled data and rebuilding the naive Bayes model finally stopped at this time.

7. After the EM iteration, we can use the final iterative conditional probability and prior probability tables of training documents to calculate the posterior probability of every testing document. And then according to the naive Bayesian classification algorithm, we can classify the testing datasets based on the class of the largest predicted posterior probability. $argmax_{c_k \in C} P(c_k|d_j)$.

Chapter 6

Model Evaluation and Experimental Results

6.1 Confusion Matrix

How good is the performance of predictive model? Confusion matrix [64] for each category is created to evaluate classifier accuracy of the classification model. The following Table 6.1 depicts a confusion matrix of a three class problem with the classes 1, 2, and 3. The rows correspond to the known true class of the data, i.e. the labels of the documents and the columns correspond to the predictions made by the model. The diagonal elements TP_i in the matrix illustrate the number of correct classifications made for each class, and the off-diagonal elements E_{ij} show the number of documents that have not been labeled by the classifier to the category that should have [65].

Table 6.1: The confusion matrix for a three class classifier.

		Predicted Class Label		
		1	**2**	**3**
Known Class Label	**1**	True Positive (TP_1)	Error(E_{12})	Error(E_{13})
	2	Error(E_{21})	True Positive (TP_2)	Error(E_{23})
	3	Error(E_{31})	Error(E_{32})	True Positive (TP_3)

For example, the following confusion matrix or actual-by-predicted Table 6.2 generated by base SAS programming gives us the whole picture of the errors made by the classification model. The

Table 6.2 from one of the experimental results for first 710 journal papers dataset is based on the condition that size of labeled data is 284 or 50% of the total 568 training data. The sum of diagonal elements 118 presents the total number of correct classifications. With 284 labeled training documents, proposed hybrid semi-supervised text classification reaches approximately 83.1% accuracy.

Table 6.2: Confusion matrix for the five-class text classification of 142 test journal papers with 284 labeled training documents.

Table of true_class by pred_class

true_class / pred_class Frequency Percent Row Pct Col Pct	1	2	3	4	5	Total
1	7 4.93 77.78 100.00	0 0.00 0.00 0.00	1 0.70 11.11 3.70	0 0.00 0.00 0.00	1 0.70 11.11 2.04	9 6.34
2	0 0.00 0.00 0.00	37 26.06 69.81 100.00	0 0.00 0.00 0.00	1 0.70 1.89 4.55	15 10.56 28.30 30.61	53 37.32
3	0 0.00 0.00 0.00	0 0.00 0.00 0.00	25 17.61 86.21 92.59	1 0.70 3.45 4.55	3 2.11 10.34 6.12	29 20.42
4	0 0.00 0.00 0.00	0 0.00 0.00 0.00	1 0.70 4.55 3.70	20 14.08 90.91 90.91	1 0.70 4.55 2.04	22 15.49
5	0 0.00 0.00 0.00	0 0.00 0.00 0.00	0 0.00 0.00 0.00	0 0.00 0.00 0.00	29 20.42 100.00 59.18	29 20.42
Total	7 4.93	37 26.06	27 19.01	22 15.49	49 34.51	142 100.00

There are a total of 24 misclassified documents. Two documents misclassified as third and fifth class respectively in the category one; one document misclassified as fourth and 15 misclassified as fifth class in category two; one document misclassified as fourth and 3 misclassified as fifth class in category three; and two documents misclassified as third and fifth class respectively in the category four.

6.2 Performance Measures: Precision, Recall, Accuracy & F-measure

In this dissertation, the performance of the proposed model is evaluated and compared with other methods using different measures. The precision, recall, accuracy and the F-measure including F_1 score and macro-F_1 average [66] are implemented to measure the accuracy of the classification model.

6.2.1 Precision

Precision, an important measure of the ability of a system to filter out irrelevant items and focus on potentially useful information, is defined as the proportion of the retrieved documents that are relevant. As a measure of the predicted classifier accuracy, precision in the following equation (6.2.1) is calculated by the column percent in the confusion matrix of Table 6.1 [67]. For example, $Precision_1$ formula is the first column percent for Table 6.1 confusion matrix and estimates the accuracy provided that class-1 has been predicted.

$$Precision = \frac{number\ of\ correct\ answers}{number\ of\ answers\ produced} \tag{6.2.1}$$

$$Precision_1 = \frac{TP_1}{TP_1 + E_{21} + E_{31}}$$

Nowadays, precision is commonly no doubt a hot benchmark measure for evaluating the effectiveness of the internet search engines such as Google and Yahoo [68]. Poor precision with high recall particularly discourages the use of a search engine. Most of users are generally

impressed by precise search performances even with a relatively low recall rate. In the process of executing a web search solution, although precision sometimes may be difficult to measure for some terms due to the subjective opinions about results, high precision is still the main goal for searchers satisfaction and has more influence than recall [69].

6.2.2 Recall

Recall, commonly also called as sensitivity, is a measure of the ability of a prediction model to successfully select instances of a certain class from a data set. Recall or the proportion of the relevant documents that are retrieved in the following equation (6.2.2) is calculated by the row percent in the confusion matrix of Table 6.1 [67]. For example, $Recall_1$ formula is the first row percent for Table 6.1 confusion matrix and estimates how well a model finds every possible document in class-1.

$$Recall = \frac{number\ of\ correct\ answers}{number\ of\ total\ possible\ corrects} \tag{6.2.2}$$

$$Recall_1 = \frac{TP_1}{TP_1 + E_{12} + E_{13}}$$

Recall is especially important in research, security and compliance applications where the user cannot afford the loss of related information [69]. A very low rate of recall means that the algorithm is very conservative and many relevant documents in the testing corpus still retained unidentified [70]. In security and compliance applications, low recall may hurt the reputation of the system since the users are worried about the ineffective in correctly classifying the hidden unsecure contents.

6.2.3 Accuracy

As a measurement of the overall correctness of the model, accuracy is the percentage of the correct predictions. The classification accuracy of a model is evaluated as the number of

correctly classified testing samples divided by the total number of testing samples. In our experiments, accuracy of a test is simply gauged by summing across all correct classifications over total number of test sets in the equation (6.2.3):

$$Accuracy = \frac{sum\ of\ correct\ classifications}{total\ number\ of\ classifications} \tag{6.2.3}$$

The goal of text classification is to minimize of the classification error on the test sets. The classification error is usually formulated by 1.0 minus the classification accuracy in (6.2.3). In some real world corpora, when most of the documents are in the non-relevant category, accuracy is not an appropriate measure for this kind of extremely skewed document classification case. In this circumstance, the measures of precision and recall, which mainly focus on the evaluation of the results of true positives (percentage of the correctly classified relevant documents) and false positives or a Type I error (percentage of mistakenly flagged irrelevant files), turn out to be other good alternatives to judge the model.

6.2.4 F-measure

There is always a tradeoff between precision and recall. Any algorithm needs to be tuned between the extremes of conservatism and liberalism so as to achieve fairly high precision and recall rates. Considering both the precision and recall in a single value, the F-measure merges them together and reflects the relative importance of these two measures.

F-measure is generally defined as the following equation (6.2.4):

$$F_\beta = \frac{(1+\beta^2)\cdot Precision \cdot Recall}{\beta^2 \cdot Precision + Recall} \tag{6.2.4}$$

Positive real parameter β is used to tune the relative weight between recall and precision rates. When β=1, the formula (6.2.4) is adjusted to the following equation (6.2.5). We call this

harmonic mean of precision and recall in (6.2.5) as traditional F-measure or balanced F-score (F_1 score) [71].

F_1 score given in formula (6.2.5) can be interpreted as a weighted average of the precision in (6.2.1) and recall in (6.2.2), where an F_1 score reaches its best value at 1 and worst score at 0.

$$F_1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{6.2.5}$$

The F_1 score is a traditional way to mix and balance two precision and recall numbers in a single score. Moreover, the macro-F_1 average, the simple average of all F_1 values assuming each category gets the same weight in the average, is used to combine all the F_1 values for different classes together.

$$macro\_F_1 = \frac{1}{|C|}\sum_{k=1}^{|C|} F_k$$

where $|C|$ is the total number of all classes.

The F- measure especially F_1 score is usually widely used in the field of information retrieval and data mining for evaluating search, document classification performance and query classification results [72]. Macro-average method calculates the F_1 score locally over each class first and then take the average over all categories [73]. The macro-F_1 average can be used to evaluate the overall performances of the model across the sets of data. When the dataset varies in size, another micro-average measure may be a useful substitution. In micro-averaging, the micro-F_1 measure is calculated globally over all categories.

6.3 Experimental Results

6.3.1 Experiments on Corpus 1: Five Classes Journal Papers

First experiments in this chapter use the real life corpus1 described in previous section 2.3.1 consisting of 710 documents unevenly distributed across five distinct classes. The document classification task is implemented to classify new test papers into the predefined five classes including mathematics education (math), aerosol research (env), text mining (txt), nuclear research (nuc), and image processing (img).

Followed the text preprocessing processes in Chapter 3, all abstracts of the 710 documents are extracted as SAS dataset first. The punctuations, 575 common English stopwords, mathematical notations and the words whose length less than three and larger than sixteen are deleted. The word suffices are then removed by the porter stemming algorithm. According to Zipf's law based dimension reduction, if the occurrence of words in all documents is lower than 1%, such rare words are deleted. Word counts of each document are scaled by TFIDF-LDA weighting in order to perform feature extraction and following semi-supervised learning experiments.

In the semi-supervised text classification experiments, the datasets are randomly split into three nonoverlapping parts. Two parts are respectively training data and testing data based on the five-fold cross validation methodology. The training data is randomly divided as a small labeled dataset and the remaining documents as the unlabeled training dataset. To verify whether the proposed method is effective when there is only a limited amount of labeled training data available, seven different experiments are performed based on the increasing labeled training data rates from 6%, 10%, 20%, 25%, 30%, and 40% to 50% of all training amount. Moreover, for checking the sensitivity and performance of the proposed hybrid model, additional

comparative experiments of three text classification methods including the conventional semi-supervised TFIDF weighting classification, TFIDF weight with our hybrid algorithm, and proposed TFIDF-LDA hybrid method are implemented. The comparative experimental results are tabulated the precision, recall and the F_1 score of five classes as the labeling rate increases in the following Table 6.3.

Table 6.3: Precision, recall and the F_1 score for the corpus 1 five-class document classification with 142 test sets.

Label	Class	Hybrid TFIDF-LDA			TFIDF+Hybrid Algorithm			TFIDF
		Recall	Precision	F1	Recall	Precision	F1	F1
35	1.math	0.777778	1	0.875	0.777778	1	0.875	0.8
	2.env	0.603774	0.888889	0.719101	0.509434	0.931034	0.658537	0.46376
	3.txt	0.862069	0.925926	0.892857	0.862069	0.757576	0.806452	0.84615
	4.nuc	0.545455	0.923077	0.685714	0.409091	0.9	0.5625	0.60606
	5.img	1	0.491525	0.659091	1	0.460317	0.630435	0.50434
56	1.math	0.666667	1	0.8	0.666667	1	0.8	0.71428
	2.env	0.660377	0.897436	0.76087	0.641509	0.944444	0.764045	0.61538
	3.txt	0.862069	0.961538	0.909091	0.862069	0.925926	0.892857	0.85714
	4.nuc	0.681818	0.9375	0.789474	0.681818	0.882353	0.769231	0.54545
	5.img	1	0.527273	0.690476	1	0.517857	0.682353	0.54368
113	1.math	0.666667	1	0.8	0.555556	1	0.714286	0.61538
	2.env	0.716981	0.95	0.817204	0.660377	0.972222	0.786517	0.62500
	3.txt	0.862069	0.892857	0.877193	0.896552	0.83871	0.866667	0.87273
	4.nuc	0.772727	0.944444	0.85	0.818182	0.9	0.857143	0.58824
	5.img	1	0.58	0.734177	0.965517	0.56	0.708861	0.54902
142	1.math	0.777778	1	0.875	0.777778	1	0.875	0.61538
	2.env	0.679245	0.972973	0.8	0.641509	1	0.781609	0.65823

	3.txt	0.862069	0.925926	0.892857	0.862069	0.925926	0.892857	0.88889
	4.nuc	0.863636	0.95	0.904762	0.863636	0.904762	0.883721	0.66667
	5.img	1	0.568627	0.725	1	0.54717	0.707317	0.56863
170	1.math	0.777778	1	0.875	0.777778	1	0.875	0.71429
	2.env	0.698113	0.973684	0.813187	0.641509	1	0.781609	0.65823
	3.txt	0.862069	0.925926	0.892857	0.862069	0.892857	0.877193	0.88889
	4.nuc	0.818182	0.947368	0.878049	0.863636	0.904762	0.883721	0.68571
	5.img	1	0.568627	0.725	0.965517	0.538462	0.691358	0.56863
227	1.math	0.777778	1	0.875	0.777778	1	0.875	0.8
	2.env	0.716981	1	0.835165	0.698113	1	0.822222	0.65822
	3.txt	0.862069	0.925926	0.892857	0.862069	0.862069	0.862069	0.87272
	4.nuc	0.863636	0.95	0.904762	0.863636	0.904762	0.883721	0.66666
	5.img	1	0.58	0.734177	0.931034	0.5625	0.701299	0.54902
284	1.math	0.777778	1	0.875	0.777778	1	0.875	0.8
	2.env	0.698113	1	0.822222	0.698113	1	0.822222	0.65822
	3.txt	0.862069	0.925926	0.892857	0.862069	0.925926	0.892857	0.88461
	4.nuc	0.909091	0.909091	0.909091	0.909091	0.833333	0.869565	0.70588
	5.img	1	0.591837	0.74359	0.931034	0.574468	0.710526	0.55769

The first column “label” in the Table 6.3 indicates the amount of the labeled training documents. For each of the five classes, we can see from the table that the proposed model has 54.55% lowest and 100% highest recall rate and the range of precision rate is from 49.15% to 100%. Basic TFIDF weight integrated with proposed method can achieve recall rate from 40.91% to 100% and 46.03% lowest and 100% highest precision value. Comparing the average F_1 score for each class, the performance of standard TFIDF method is even worse than the modified TFIDF weight with hybrid algorithm. For example, when 284 documents are added as the labeled training data, the F_1 score of proposed method achieves 90.91% at class-4, while that of modified TFIDF and basic TFIDF are 86.96% and 70.59% respectively. As the percentage of labeled

training documents increases, it can be observed from the table that the proposed hybrid TFIDF-LDA semi-supervised approach performs better on the whole. The detailed comparison of corresponding macro-F_1 curves for predicting five categories using three methods are plotted in the following figures.

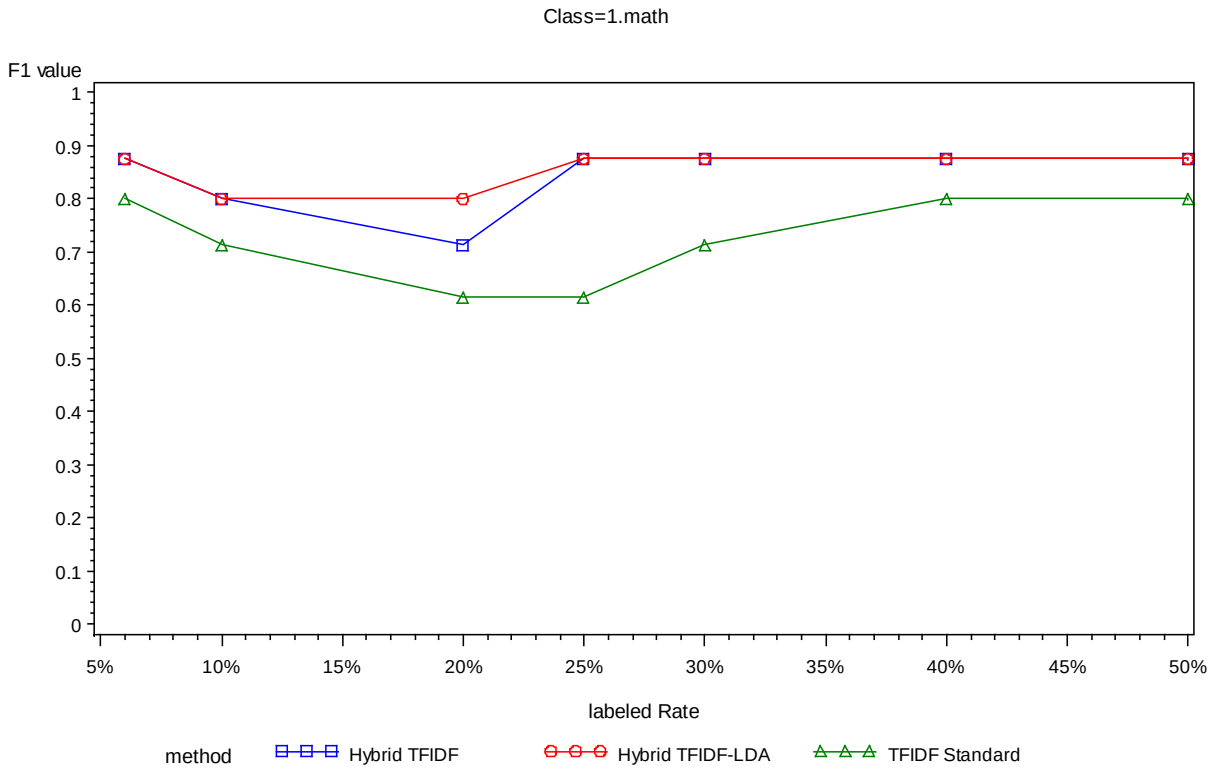


Figure 6.1: Comparison of F1 score for class-1 mathematical education on corpus 1

The above Figure 6.1 shows the specific comparison results of F_1 score for class-1 mathematical education on corpus 1. The horizontal axis illustrates the percentage of labeled training data and the vertical axis represents the average classifier F_1 score on test sets. The amount of labeled training data varies from 6% to 50% and three distinct colored curves show the different methods used on corpus 1. Comparing the classification F-measure curve, the proposed hybrid TFIDF-LDA method is substantially better than the traditional TFIDF methods. Except 20% labeling rate, the F_1 scores of class-1 for modified hybrid TFIDF and proposed hybrid TFIDF-LDA semi-

supervised model are almost the same. The highest F_1 score for class-1 mathematical education reaches 87.5%. This might due to the fact that the experimental data are real-world textual data which contains only a very few amount of labeled mathematical education documents.

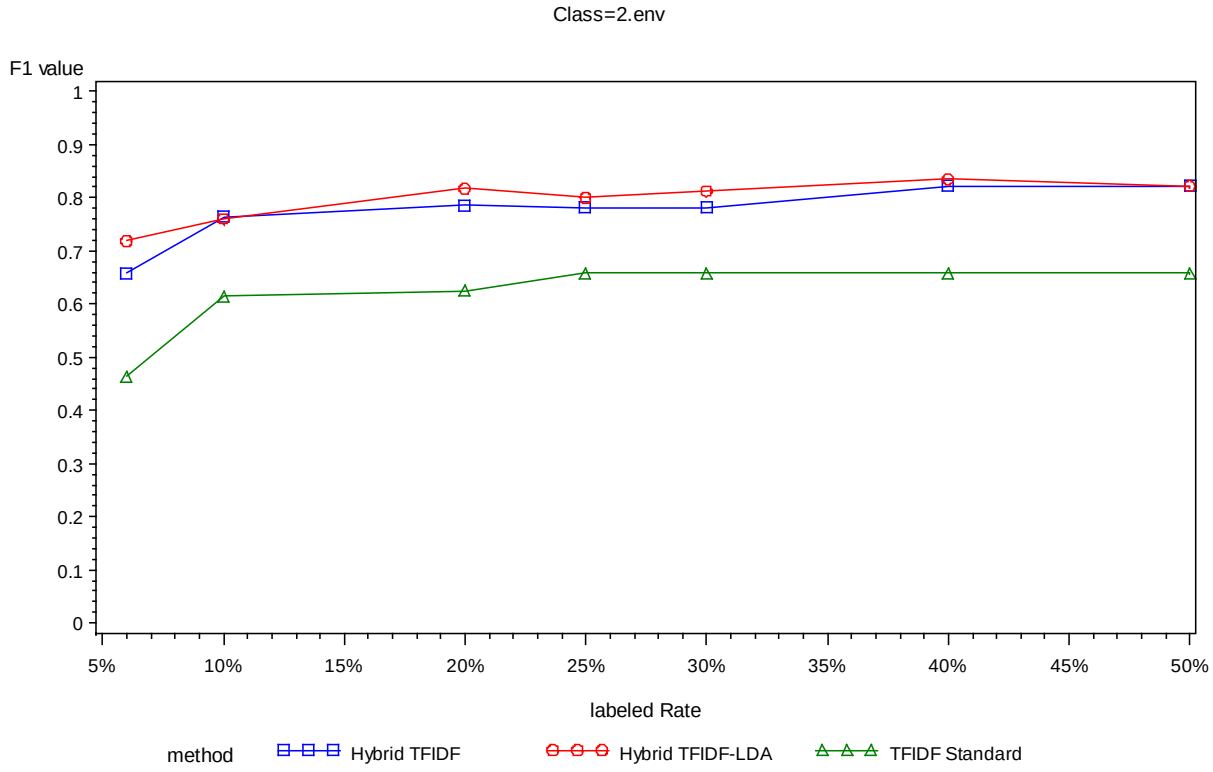


Figure 6.2: Comparison of F1 score for class-2 aerosol research on corpus 1

The above Figure 6.2 illustrates the comparison results of F_1 score for class-2 aerosol research on corpus 1. The highest F_1 score for class-2 reaches 83.5% using hybrid TFIDF-LDA semi-supervised model. We can observe that, as the number of training data varies, the hybrid TFIDF-LDA achieves better performance than other two baseline methods. However, the F_1 score of all three methods fluctuate slightly as the rate of labeled data increases around 20%. This phenomenon might occur due to the random noise or the manually labeling error of the real-world experimental data.

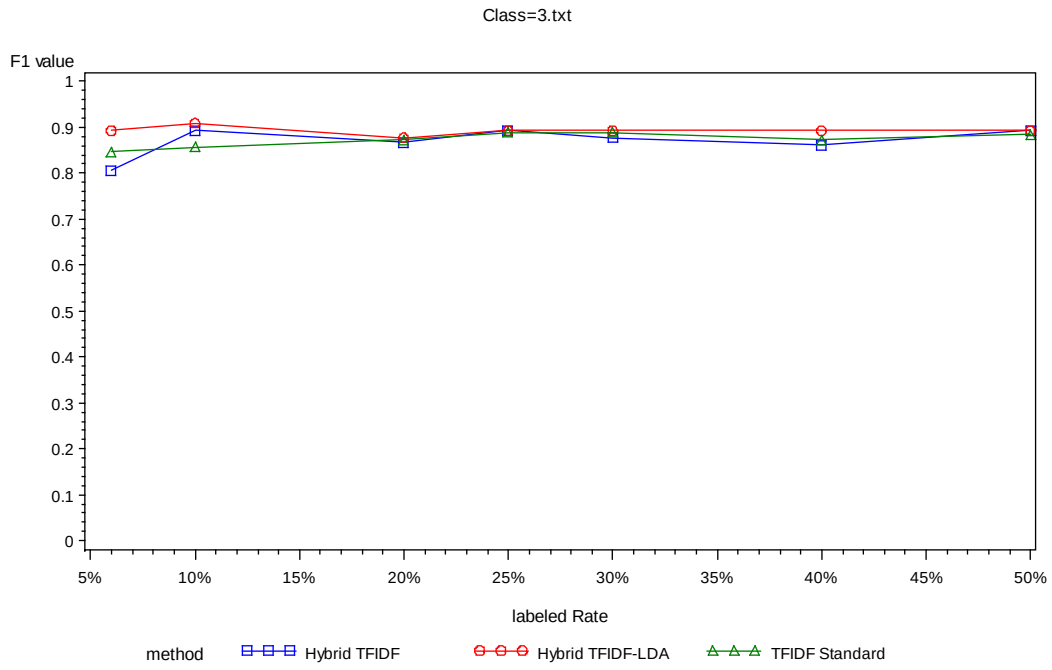


Figure 6.3: Comparison of F1 score for class-3 text mining documents on corpus 1

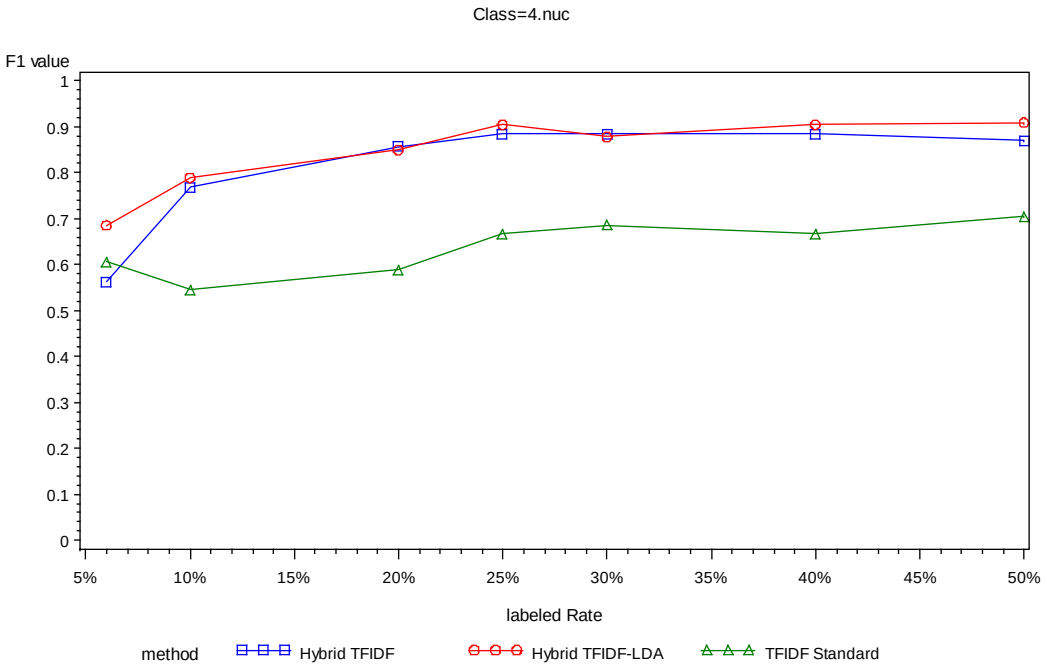


Figure 6.4: Comparison of F1 score for class-4 nuclear research documents on corpus 1

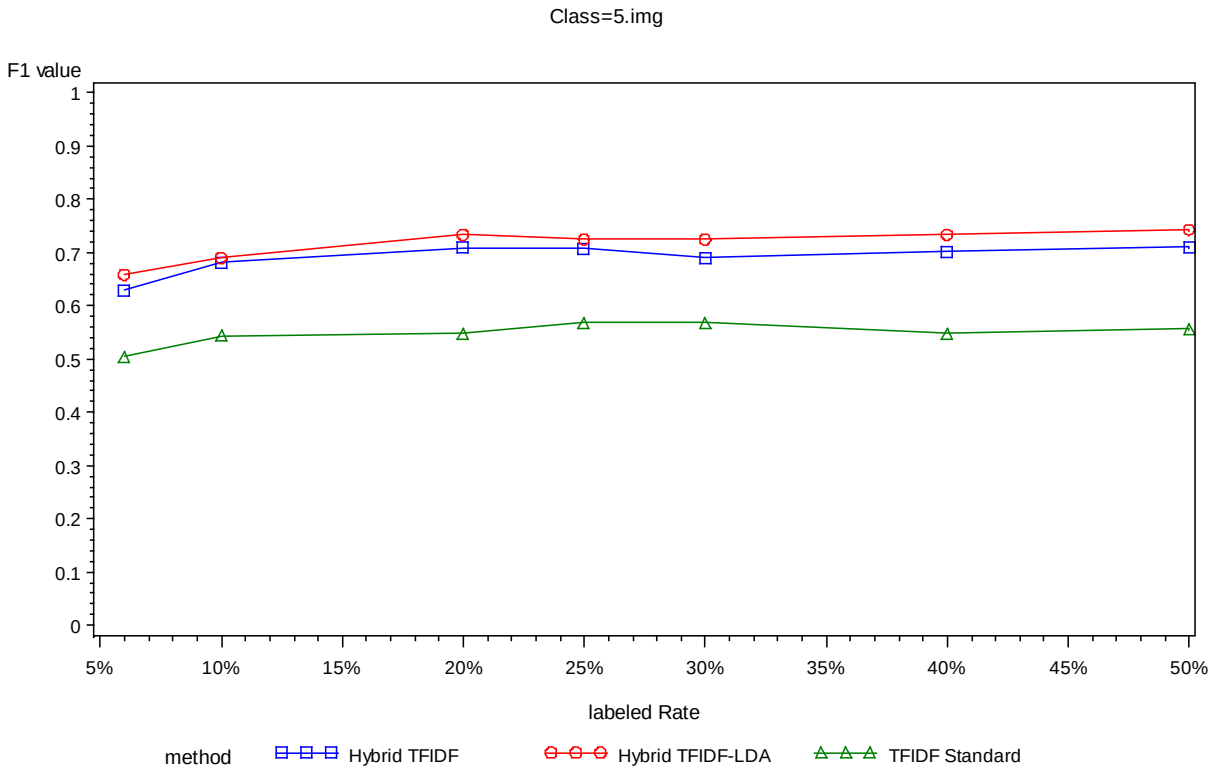


Figure 6.5: Comparison of F1 score for class-5 image processing documents on corpus 1

For the class-3 text mining, the Figure 6.3 shows more stable F_1 score curves. The highest F_1 score for class-3 reaches 90.9% using hybrid TFIDF-LDA semi-supervised algorithm. We can find that, when the labeled training data number is small especially less than 10%, the hybrid TFIDF-LDA achieves much better performance than other two methods for text mining document classification.

The above Figure 6.4 shows the F_1 score curves for the class-4 nuclear research. The highest F_1 score for class-4 reaches 90.91% using hybrid TFIDF-LDA semi-supervised model. When less labeled training data is available, the advantage of hybrid TFIDF-LDA is still more obvious for the nuclear research documents.

Figure 6.5 shows the F_1 score curves for the class-5 image processing document classification. The highest F_1 score for class-5 reaches 74.36% using hybrid TFIDF-LDA semi-supervised algorithm, which demonstrates the effectiveness of proposed hybrid method in predicting document categories.

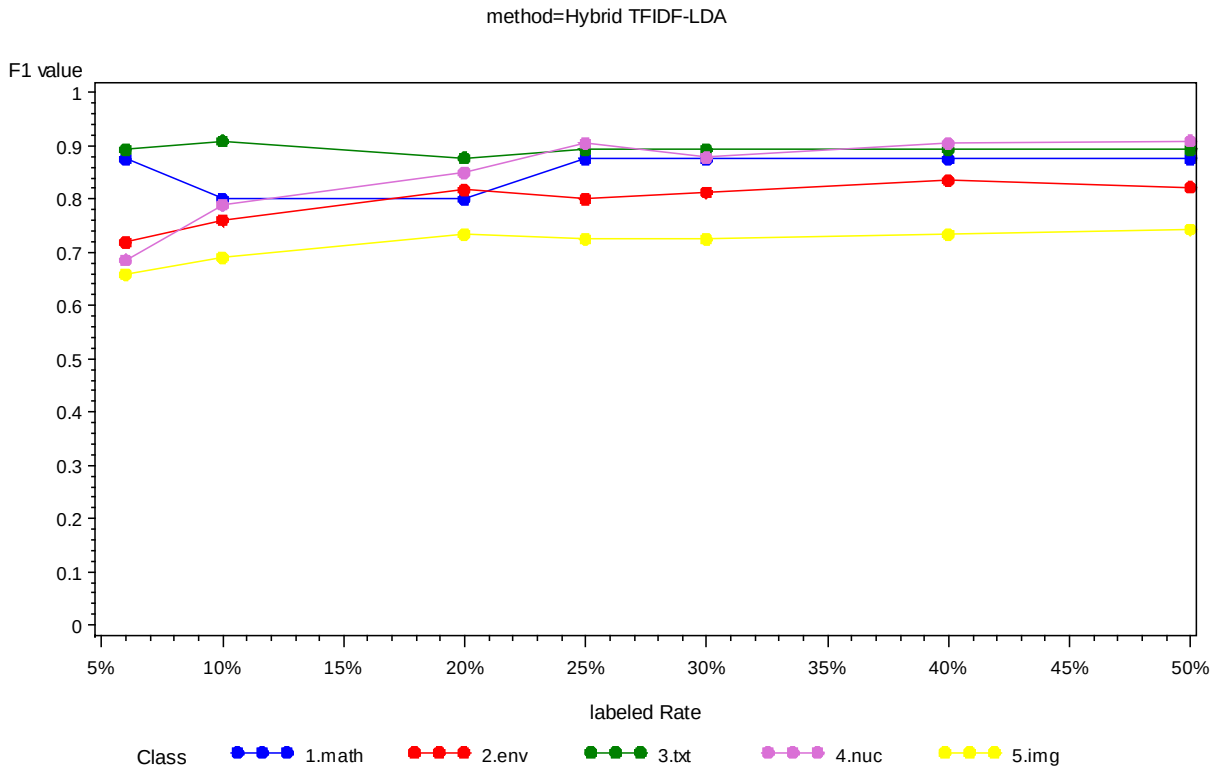


Figure 6.6: Comparison of F1 score for all 5-classes document classification on corpus 1 using proposed hybrid TFIDF-LDA model

The above Figure 6.6 summarizes the F_1 score curve for all 5-classes document classification on corpus 1 using proposed hybrid TFIDF-LDA model. The class-5 image processing document classification has the relatively lower accuracy comparing other four classes. This might due to the similar cross contents between some image processing and mathematical education documents. Since some class-1 documents present the curriculum and mathematical teaching methods in the disciplines of mathematics, science, engineering, and computing.

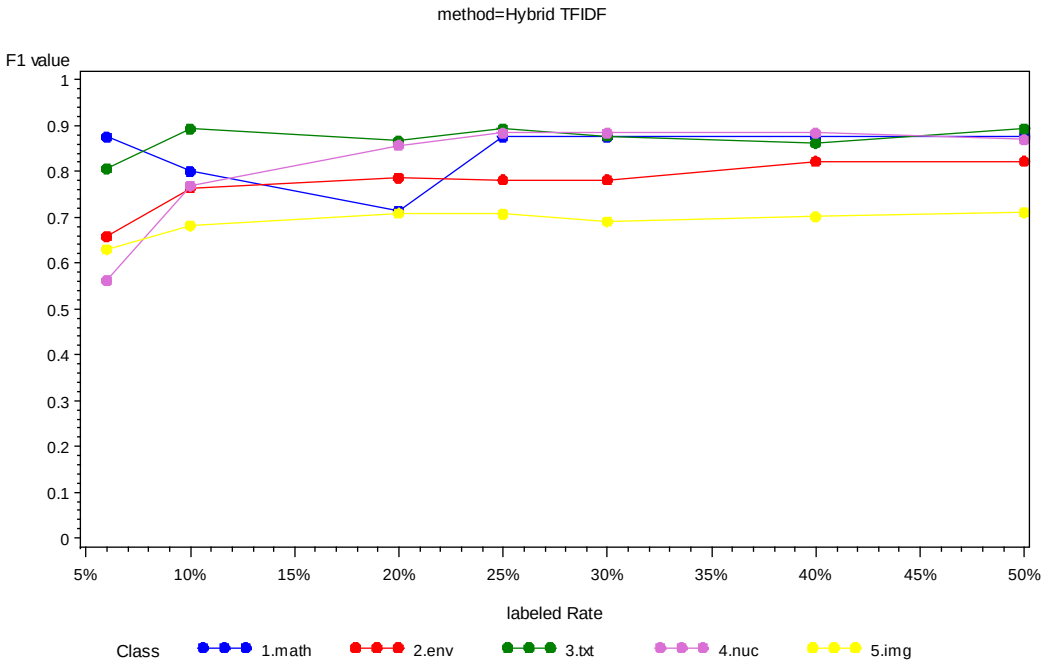


Figure 6.7: Comparison of F1 score for all 5-classes document classification on corpus 1 using modified hybrid TFIDF model

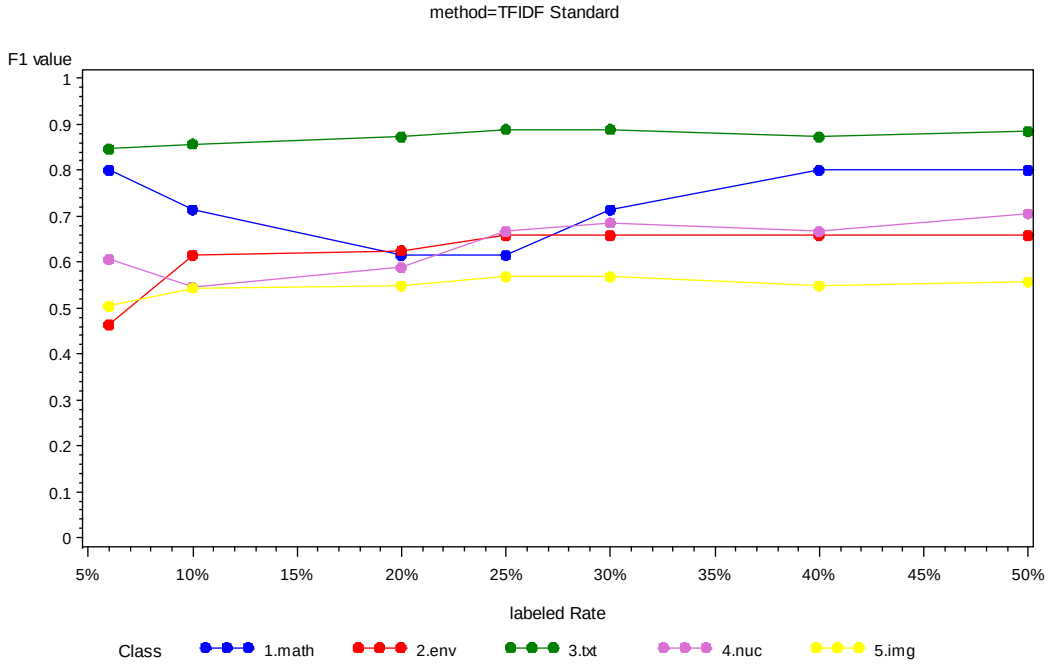


Figure 6.8: Comparison of F_1 score for all 5-classes document classification on corpus 1 using standard TFIDF algorithm.

Comparing with the F_1 score curves for all 5-classes document classification using modified hybrid TFIDF algorithm in Figure 6.7 and using standard TFIDF algorithm in Figure 6.8, it is easy to check from the picture that the proposed model is more stable even when the ratio of labeled to unlabeled documents is relatively small around 6%. The following Table 6.4 also summarizes the number of used features and average macro-F_1 score for three different methods in details.

Table 6.4: Comparison of three different document classification methods on the 5-classes corpus 1 using number of used features and macro-F_1 score.

Labeled	**TFIDF Standard**		**TFIDF+Proposed model**		**Proposed Hybrid TFIDF-LDA**	
	Features	**macro-F_1**	**Features**	**macro-F_1**	**Features**	**macro-F_1**
6%(35)	7245	0.644066	4597	0.706585	1851	0.766353
10%(56)	7852	0.655191	4595	0.739794	1942	0.789982
20%(113)	7850	0.650073	4595	0.786695	1873	0.815715
25%(142)	7819	0.679559	4575	0.828101	1650	0.839524
30%(170)	7819	0.703149	4575	0.821776	1672	0.836819
40%(227)	7781	0.709328	4575	0.828862	1991	0.848392
50%(284)	7810	0.721284	4581	0.834034	1650	0.848552

Each row indicates a specific experiment at a given percentage of labeled training data. Comparing proposed hybrid method with the most commonly used TFIDF base run on the corpus 1, the highest macro-F_1 average of 84.86% and smallest amount of used features of 1650 in the above table show the superiority of the hybrid semi-supervised document classification approach. Only labeling 6% training data using proposed model in corpus 1, we can achieve the similar or even much higher accuracy than labeling 50% training documents using TFIDF standard method. We can see that having more labeled training documents improves the

classification F-measure in most cases although when there is a little rise and fall at very few labeled rates, the differences of macro-F_1 average are small. It can also be observed from the Figures 4.1 that as the count of the labeled training data increases in the document classification experiments for corpus 1, proposed hybrid TFIDF-LDA model and modified hybrid TFIDF method perform generally stable and much better than the standard TFIDF approach even when the labeling rate is small.

6.3.2 Experiments on Corpus 2: Seven Classes Journal Papers

Although the experimental results shown in Chapter 6.3.1 are quite encouraging for the document classification on the 5-class corpus 1, it would be even better to check the model performance with larger data sets with more classes. Thus the real world document collection corpus 2 described in previous section 2.3.2 is employed to make further experiments and comparative study. For maintaining the fair comparison, the corpus 2 consists of 1133 documents unevenly distributed across seven different classes, including mathematics education (math), aerosol research (env), text mining (txt), nuclear research (nuc), image processing (img), transportation (tra) and robotics (rob).

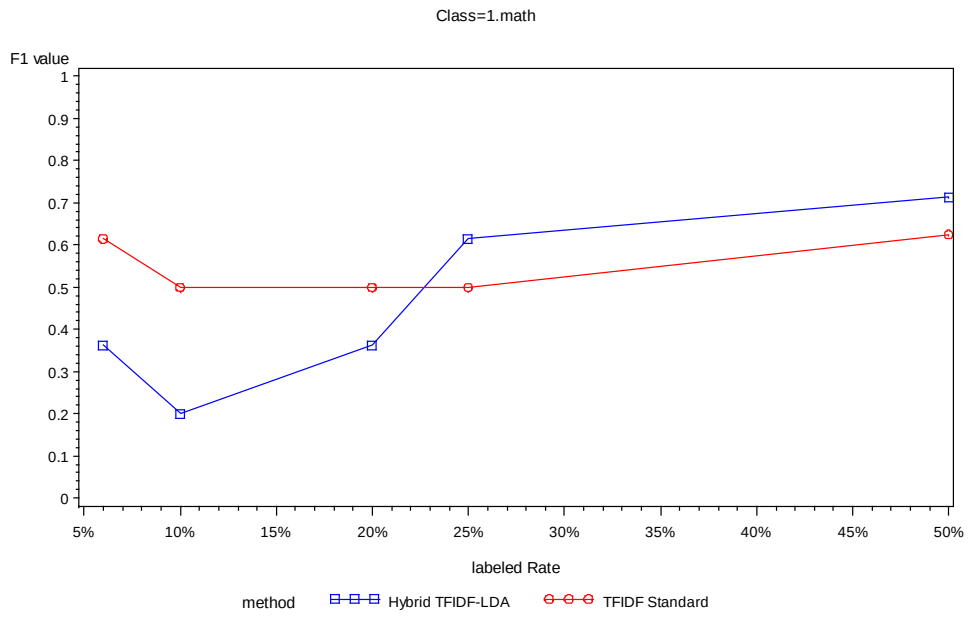
Using the same text preprocessing and feature selection procedures as the 5-class corpus 1, five experiments based on different amount of labeled training data are implemented to compare the performances of proposed hybrid model and the TFIDF base run. As the labeled training data rate increases from 6%, 10%, 20%, 25% to 50%, the document classification measurements containing precision, recall and the F_1 score for each class in corpus 2 are presented in the following Table 6.5.

Table 6.5: Precision, recall and the F_1 score for the corpus 2 seven-class document classification with 226 test sets.

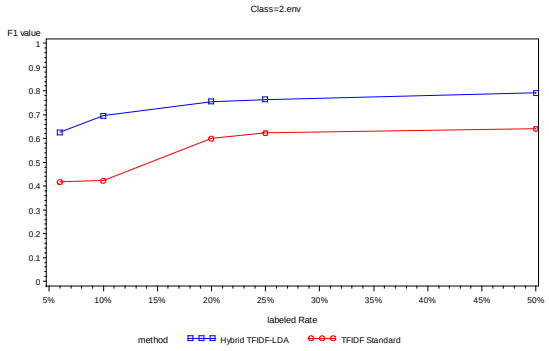
Label	Class	TFIDF-LDA			TFIDF Standard		
		Recall	Precision	F1	Recall	Precision	F1
6%(58)	1.math	0.222222	1	0.363636	0.444444	1	0.615385
	2.env	0.490566	0.866667	0.626506	0.264151	1	0.41791
	3.txt	0.724138	0.84	0.777778	0.793103	0.884615	0.836364
	4.nuc	0.318182	0.875	0.466667	0.318182	0.875	0.466667
	5.img	0.689655	0.909091	0.784314	0.37931	0.916667	0.536585
	6.tra	0.833333	0.875	0.853659	0.738095	0.939394	0.826667
	7.rob	0.904762	0.383838	0.539007	0.952381	0.310078	0.467836
10%(96)	1.math	0.111111	1	0.2	0.333333	1	0.5
	2.env	0.584906	0.861111	0.696629	0.283019	0.833333	0.422535
	3.txt	0.793103	0.958333	0.867925	0.758621	0.956522	0.846154
	4.nuc	0.5	0.916667	0.647059	0.363636	0.727273	0.484848
	5.img	0.689655	0.909091	0.784314	0.37931	0.6875	0.488889
	6.tra	0.833333	0.945946	0.886076	0.785714	0.970588	0.868421
	7.rob	0.928571	0.414894	0.573529	0.880952	0.305785	0.453988
20%(193)	1.math	0.222222	1	0.363636	0.333333	1	0.5
	2.env	0.641509	0.918919	0.755556	0.45283	0.888889	0.6
	3.txt	0.862069	0.961538	0.909091	0.827586	0.827586	0.827586
	4.nuc	0.681818	0.9375	0.789474	0.454545	0.833333	0.588235
	5.img	0.758621	0.956522	0.846154	0.413793	0.857143	0.55814
	6.tra	0.833333	0.921053	0.875	0.761905	1	0.864865
	7.rob	0.928571	0.464286	0.619048	0.833333	0.321101	0.463576
25%(242)	1.math	0.444444	1	0.615385	0.333333	1	0.5
	2.env	0.641509	0.944444	0.764045	0.45283	1	0.623377
	3.txt	0.827586	0.923077	0.872727	0.827586	0.96	0.888889
	4.nuc	0.727273	0.8	0.761905	0.454545	0.833333	0.588235
	5.img	0.689655	0.952381	0.8	0.448276	0.866667	0.590909
	6.tra	0.880952	0.902439	0.891566	0.809524	0.971429	0.883117
	7.rob	0.880952	0.474359	0.616667	0.904762	0.339286	0.493506
50%(454)	1.math	0.555556	1	0.714286	0.555556	0.714286	0.625
	2.env	0.679245	0.947368	0.791209	0.471698	1	0.641026

	3.txt	0.862069	0.961538	0.909091	0.827586	0.923077	0.872727
	4.nuc	0.818182	0.9	0.857143	0.5	0.846154	0.628571
	5.img	0.793103	1	0.884615	0.448276	1	0.619048
	6.tra	0.833333	1	0.909091	0.714286	1	0.833333
	7.rob	0.97619	0.518987	0.677686	0.952381	0.357143	0.519481

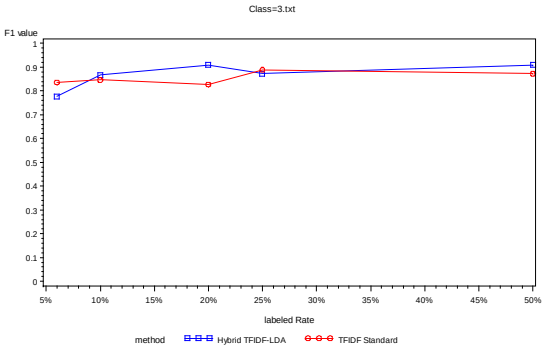
Table 6.5 tabulates the results for each of the seven classes in details. We can observe that the proposed model has 97.62% highest recall rate and standard TFIDF method has 95.24%. Comparing the highest F_1 score for each class, the proposed model can reach 90.91% and TFIDF baseline can get 88.89%. However, the Table 6.5 also presents that at the lowest percentages of labeled training data, the performances for the proposed model in class one and class three are inferior to the standard TFIDF baseline. The following Figure 6.9(a) shows the changes of two F_1 score curves for class-1 mathematical education.



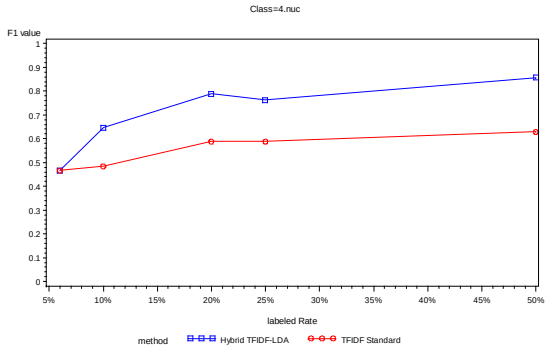
(a) class-1 mathematical education (math)



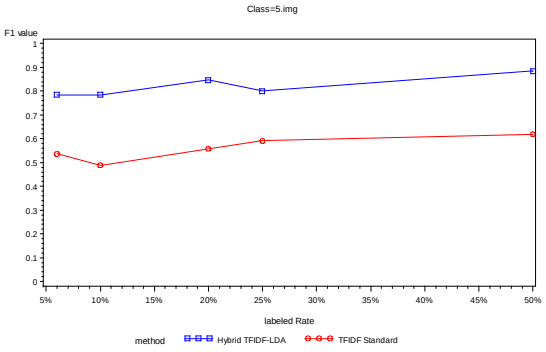
(b) class-2 aerosol research (aer)



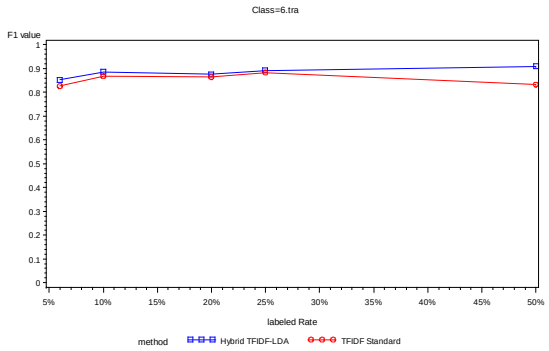
(c) class-3 text mining (txt)



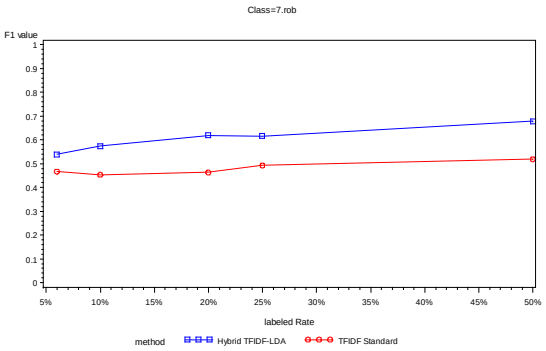
(d) class-4 nuclear research (nuc)



(e) class-5 image processing (img)



(f) class-6 transportation (tra)



(g) class-7 robotics (rob)

Figure 6.9: F_1 score for each class on corpus 2

In the beginning, the F_1 score for class 1 is lower and unstable using proposed methods due to the limited amount of the labeled class-1 training document. As the labeled training percentage increases, the performance of the proposed hybrid approach in class-1 improves rapidly at 25% labeled case and is better than the performance with the standard TFIDF baseline. We make closer analysis of the experimental results for each class from Figure 6.9. It shows that except the variability in class-1 and class-3, the proposed hybrid TFIDF-LDA semi-supervised approach generally outperforms the basic TFIDF method.

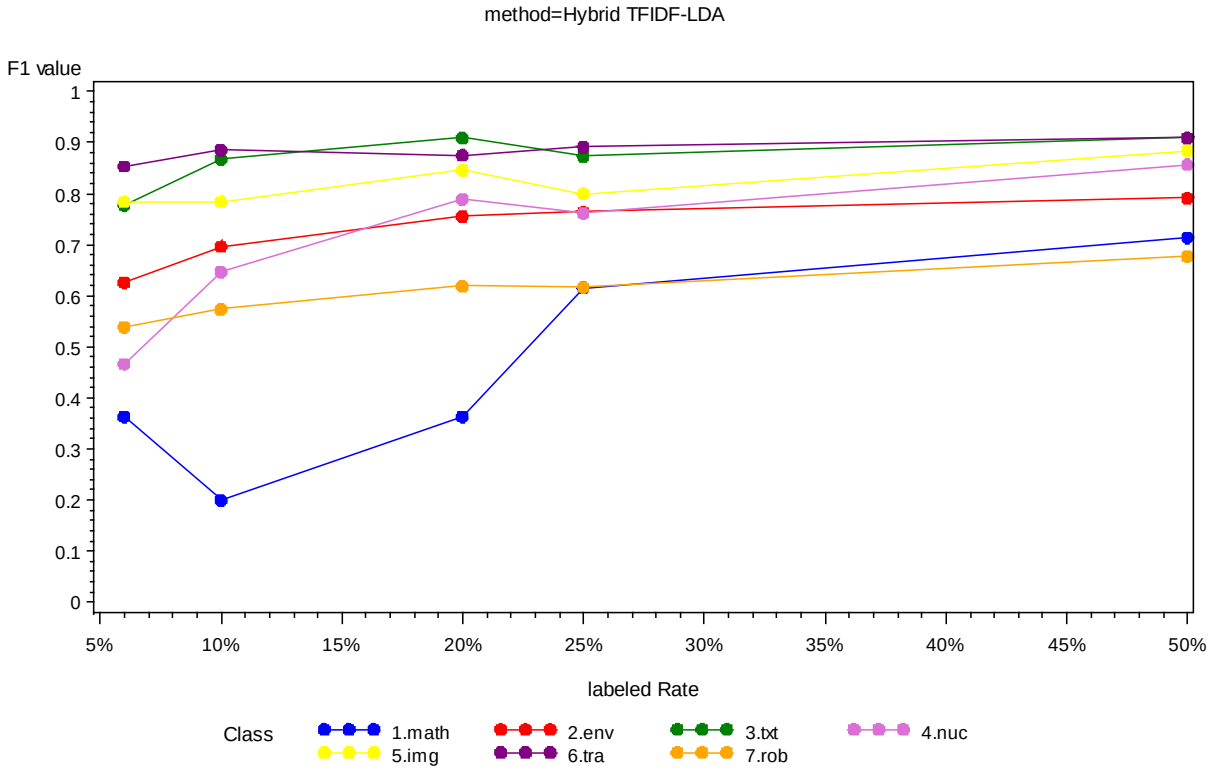


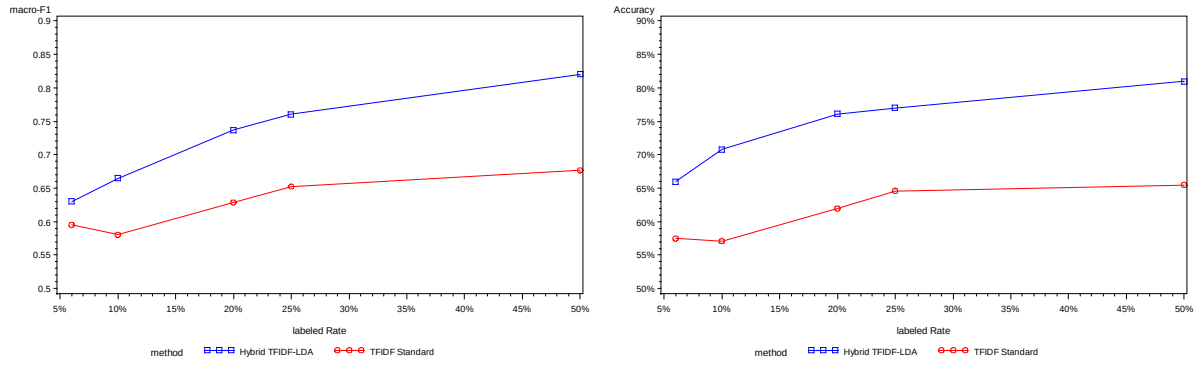
Figure 6.10: F1 score for all seven classes on corpus 2 using proposed hybrid model

From the above overview of the classification performances of all classes using proposed hybrid model in Figure 6.10, the result of class-1 mathematical education is the worst. As we discussed in previous section, this might due to the fact that some class-1 documents have cross related contents with other classes. Since this is only one real world document collection and empirical

classification performance is likely to change substantially over different classes. After comparing the results under different categories, we then see the full story of our experiments using two approaches as shown in Table 6.6.

Table 6.6: Comparison of two document classification methods on corpus 2 using number of used features, macro-F_1 score and accuracy.

Labeled	**TFIDF Standard**			**Proposed Hybrid TFIDF-LDA**		
	Features	**macro-F_1**	**Accuracy**	**Features**	**macro-F_1**	**Accuracy**
6%(54)	10254	0.595345	57.5%	2566	0.630224	66%
10%(90)	9723	0.580691	57.1%	2566	0.665076	70.8%
20%(181)	9723	0.628915	62%	2566	0.736851	76.1%
25%(226)	9703	0.652576	64.6%	2357	0.760328	77%
50%(454)	10031	0.677027	65.5%	2610	0.820446	81%



(a) macro-F_1 (b) accuracy

Figure 6.11: Comparison of macro-F_1 and accuracy curves on corpus 2

The detailed comparison of corresponding macro-F_1 and accuracy curves for predicting seven classes are plotted in Figure 6.11. We see that in both two plots, the performances were improved when more labeled training data were added. The improvements of the basic TFIDF

methods are reasonably modest comparing with the substantial modifications from proposed hybrid TFIDF-LDA approach. The proposed hybrid model reaches 81% accuracy on corpus 2 while the TFIDF baseline achieves its highest accuracy at 65.5%. From the intensive experiments on the real life corpus 1 and corpus 2, we can find that proposed hybrid TFIDF-LDA semi-supervised model provides a promising solution for real world document classification problem and offers higher accuracy with less human labeling effort.

6.3.3 Experiments on Corpus 3: 20 Newsgroups

Considering the training efficiency, the 20 Newsgroups dataset used in this dissertation randomly select 3000 documents with 6 categories from the benchmark 20 Newsgroups. There are three versions of the 20 Newsgroups data set and the 18828 documents version with "From" and "Subject" headers is used in the experiments. Using the same text preprocessing and feature selection procedures as previous two corpora, the proposed method and TFIDF base run are implemented in the corpus 3. As the data description given in Table 2.1, four different experiments are implemented on corpus 3 using increasing labeled training document rate from 10%, 20%, 30% to 50% of total 2700 training documents.

Since this dataset is collected from the internet, which are typically not very well written or edited by users and might have some random noise due to the typo and carelessness [74]. Some classes of 20 Newsgroups data are ambiguous and how to clearly extract the clusters of different classes is a difficult problem. Table 6.7 shows the experimental precision, recall and the F_1 score for each class in corpus 3. As the amount of labeled training set increases, we can observe from the table that all measurements gradually improve. The proposed model has 50% lowest and 98% highest recall rate and the range of precision rate is from 51% to 100%.

Table 6.7: Precision, recall and the F_1 score for the corpus 3 with 300 test sets.

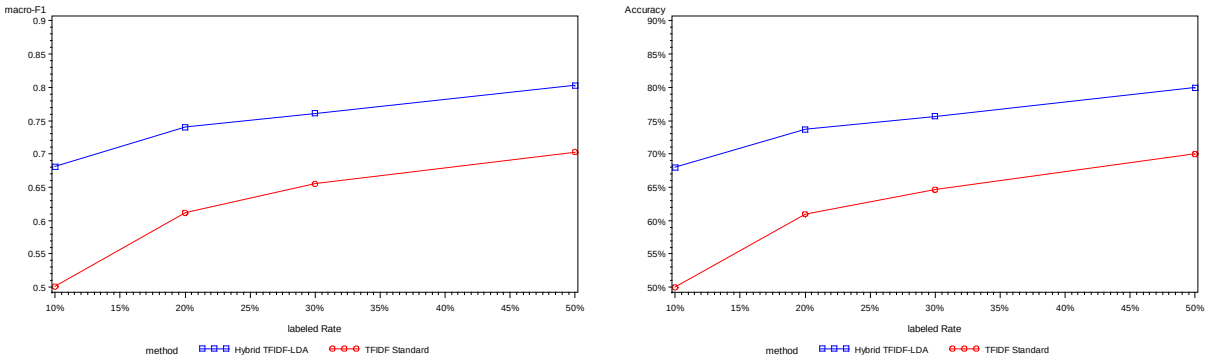
Label	Class	Proposed model		
		Recall	Precision	F1
10%	1. comp.graphics	0.56	0.736842	0.636364
	2. comp.os.ms-windows.misc	0.72	0.75	0.734694
	3. comp.windows.x	0.5	0.595238	0.543478
	4. misc.forsale	0.7	0.795455	0.744681
	5. rec.sport.hockey	0.62	0.96875	0.756098
	6. sci.crypt	0.98	0.510417	0.671233
20%	1. comp.graphics	0.72	0.765957	0.742268
	2. comp.os.ms-windows.misc	0.78	0.866667	0.821053
	3. comp.windows.x	0.56	0.717949	0.629213
	4. misc.forsale	0.72	0.837209	0.774194
	5. rec.sport.hockey	0.66	0.970588	0.785714
	6. sci.crypt	0.98	0.532609	0.690141
30%	1. comp.graphics	0.74	0.787234	0.762887
	2. comp.os.ms-windows.misc	0.8	0.851064	0.824742
	3. comp.windows.x	0.64	0.8	0.711111
	4. misc.forsale	0.72	0.857143	0.782609
	5. rec.sport.hockey	0.66	0.970588	0.785714
	6. sci.crypt	0.98	0.544444	0.7
50%	1. comp.graphics	0.8	0.833333	0.816327
	2. comp.os.ms-windows.misc	0.82	0.911111	0.863158
	3. comp.windows.x	0.68	0.894737	0.772727
	4. misc.forsale	0.8	0.869565	0.833333
	5. rec.sport.hockey	0.7	1	0.823529
	6. sci.crypt	0.98	0.556818	0.710145

The highest F_1 score achieves 86.3% at class-2, when the percentage of labeled training data is 50%. As shown in the following Table 6.8, a whole picture of model performance is tabulated using macro-F_1 average and accuracy rate. The proposed model reaches 80% accuracy with 6870 word features at 50% labeled training rate, while as shown in the following Table 6.8, the tuned

TFIDF base run can get 70% accuracy with more features at the same amount of the labeled training for corpus 3.

Table 6.8: Proposed document classification method on corpus 3 using macro-F_1 score and accuracy.

Labeled	**TFIDF Standard**		**Proposed Hybrid TFIDF-LDA**	
	macro-F_1	**Accuracy**	**macro-F_1**	**Accuracy**
10%	0.501346	50%	0.681091	68%
20%	0.611875	61%	0.74043	73.67%
30%	0.655281	64.67%	0.761177	75.67%
50%	0.703068	70%	0.803203	80%



(a) macro-F_1 (b) accuracy

Figure 6.12: Comparison of macro-F_1 and accuracy curves on corpus 3

The above Figure 6.12 illustrates the accuracy and the macro-F_1 measures for two methods. As the amount of labeled training data increases, the performance of proposed method is much better than the TFIDF base run. Hybrid semi-supervised model improves the classification accuracy and at the same time reduces the need for large amount of manually labeled training documents. Therefore, you will suffer less from the demand of scarce labeled training data in

real life projects. Incorporating unlabeled documents with desired labeled training documents together can generally improve the classification accuracy when the class boundaries of documents are relatively clear. Considering the tradeoffs in accuracy and labeling efficiency, proposed hybrid model can be a promising solution for real life text classification tasks.

6.3.4 Comparative Experiments with Unsupervised, Supervised and Semi-Supervised Learning

Traditional supervised learning method uses only the labeled data to build classifiers. Unsupervised learning discovers the meaningful clusters in the raw unlabeled data. Can we use the unlabeled data to modify and learn a preferable classifier? Semi-supervised learning approach employs additional large amount of unlabeled data together with the labeled data to train and help create better classifier.

To check whether the semi-supervised learning with both labeled and unlabeled data can improve the classification accuracy of the supervised learning using only the labeled data. Additional experiments on the 20 Newsgroups dataset were implemented. Four classes of documents including the comp.graphics, comp.os.ms-windows.misc, misc.forsale, and rec.sport.hockey are randomly selected from the 20 Newsgroups collection.

In the experiments, a total of 980 newsgroup postings from four categories are used for the semi-supervised learning model. This dataset has the same balanced amount of documents per class including total of 80 documents (20 each per class) as the test set, and 100 documents (25 each per class) as the labeled training set. The unlabeled training sets are respectively 200, 500 (125 each per class) and 800 documents in three experiments. For the fair comparisons, a total of 188 newsgroup postings from four categories including the exactly same balanced amount of 80

testing documents and 100 labeled training documents are used for the supervised learning approach. Without the additional unlabeled training data, the performance of supervised learning method is also evaluated and compared with the completely unsupervised learning approach using only the clustering of the same 80 test documents.

Table 6.9: Comparative experimental results with Unsupervised, Supervised and Semi-Supervised learning methods

Methods	Labeled Train	Unlabeled Train	Test	Accuracy
Unsupervised			80	55%
Supervised	100		80	67.5%
Semi-Supervised	100	200	80	71.25%
	100	500	80	73.75%
	100	800	80	77.5%

The above Table 6.9 shows the comparative experimental results with these three machine learning document classification methods based on the same preprocessing and proposed feature extraction algorithms. The classification performance on the selected 20 Newsgroups dataset using 100 labeled documents with additional unlabeled documents in the semi-supervised setting is more accurate than the supervised learning without the use of unlabeled data. Increasing the unlabeled training data in the experiments can also further improve the performance. The unsupervised learning model discovered four clusters in the data set resulting in a lowest clustering accuracy of 55%.

Unsupervised learning rarely has higher accuracy than the semi-supervised learning method since it uncovers the latent structures in the documents without the help of any labeled information and is very difficult to automatically get the exact desired conceptual categories. If there are inherent relationships between the labeled and the unlabeled data distribution [76], as illustrated in the experimental results of Table 6.9, the semi-supervised learning can beat over the supervised learning. With the small amounts of training data and relatively large amount of unlabeled training data, using semi-supervised learning normally can achieve more accurate classifiers than the supervised learning without the help of unlabeled training data.

Good matching of problem structure with model assumption can improve the classifier performance [24] [75]. Several semi-supervised learning methods including the transductive support vector machines (TSVMs) and information regularization assume that the decision boundary should go through a low density region of the data domain [53] [76]. However, some researchers found that the unlabeled data cannot be always useful to improve the classification accuracy in some cases [24] [53] [77] [78]. At this time the sound theoretical explanation of the advantages of unlabeled data in the semi-supervised learning approach is difficult and is still an open question.

Chapter 7

Conclusion and Future Work

7.1 Experiment Findings

In this dissertation, a hybrid semi-supervised text classification approach, integrating porter stemming, new adaptive TFIDF-LDA weighting, Zipf's law, multinomial Naive Bayes classifier, and expectation-maximization algorithm, is proposed for literature classification. After multiple experiments on the proposed new approach, the following major findings bring us a deeper and broader understanding of the text mining.

1. The words could be excluded for being too short, too long or too common. A string whose length is less than three and larger than sixteen can be removed.

2. If the occurrence of words is less than 1% of all documents in the corpus, deletion of such “rare words” can improve the classification accuracy.

3. The proposed hybrid semi-supervised text classification model can greatly reduce the feature dimension and meanwhile make feature selection robust.

4. The combination of multinomial naive Bayes classifier and EM algorithm allows the co-training of label and unlabeled documents more straightforward and reliable especially for the high dimensional inputs and the small amount of training data.

7.2 Future Work

In my future research, the unlabeled training data are planned to extend which means the future experiments will be focused on the use of a relatively small amount of labeled data with a very large amount of unlabeled data. More experiments concerning the impact of IDF weighting on

the improvement of the empirical performance of LDA model will also be implemented. How can people effectively remove the less informative words will be always in the lists of the future work. Doing text mining well for the documents is an on-going exploration. The methods used in this dissertation will keep updating and renewing in the future.

References

[1] B. Sheridan. (2012). *Is Cue the Cure for Information Overload?* Available: http://www.businessweek.com/articles/2012-06-19/is-cue-the-cure-for-information-overload

[2] A. Alexander. (2012). *How Technology Usage Causes Stress in the Brain*. Available: http://ansonalex.com/infographics/how-technology-usage-causes-stress-in-the-brain-infographic/

[3] M. Hearst. (2003). *What Is Text Mining?* Available: http://people.ischool.berkeley.edu/~hearst/text-mining.html

[4] M. Radovanovic and M. Ivanovic, "TEXT MINING: APPROACHES AND APPLICATIONS," *Novi Sad Journal of Mathematics*, vol. 38, pp. 227-234, 2008.

[5] IBM. (2012). *What is big data?* Available: http://www-01.ibm.com/software/data/bigdata/

[6] Wikipedia. *Big data*. Available: http://en.wikipedia.org/wiki/Big_data

[7] Oracle. *White paper: Big Data Strategy Guide* Available: http://www.oracle.com/us/technologies/big-data/index.html

[8] C. C. Consortium, "Challenges and Opportunities with Big Data," 2012.

[9] R. F. i. Cancho and R. V. Sole, "Least Effort and the Origins of Scaling in Human Language," *Proc. US National Academy of Sciences*, vol. 100, pp. 788-791, 2003.

[10] S. P. Christian. (2011). *Machine Learning Text feature extraction (tf-idf)* Available: http://pyevolve.sourceforge.net/wordpress/?p=1589

[11] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, pp. 613-620, 1975

[12] D. T. Peter and P. Pantel, "From Frequency to Meaning: Vector Space Models of Semantics," *Journal of Artificial Intelligence Research* vol. 37, pp. 141-188, 2010.

[13] V. V. Raghavan and S. K. M. Wong, "A critical analysis of vector space model for information retrieval," *Journal of the American Society for Information Science*, vol. 37, pp. 279-287, 1986.

[14] G. Salton, "Automatic Text Processing," *Addison-Wesley Publishing Company*, 1988.

[15] I. Antonellis, C. Bouras, and V. Poulopoulos, "Personalized news categorization through scalable text classification," presented at the Proc 8th Asia-Pasific Web Conf, 2006.

[16] N. Liu, B. Zhang, J. Yan, Z. Chen, W. Liu, F. Bai, and L. Chien., "Text representation: From vector to tensor," in *Proc. 5th IEEE Int'l. Conf. Data Mining (ICDM'04)*, 2005, pp. 725-728.

[17] A. Samuel, "Some Studies in Machine Learning Using the Game of Checkers," *IBM Journal*, vol. 3, pp. 210-229, 1959.

[18] T. Mitchell, *Machine Learning*: McGraw Hill, 1997.

[19] N. Andrew. *Machine Learning*. Available: https://www.coursera.org/course/ml

[20] A. Peter, T. Jaksch, and R. Ortner, "Near-optimal regret bounds for reinforcement learning " *Journal of Machine Learning Research*, vol. 11, pp. 1563-1600 2010.

[21] N. Grira, M. Crucianu, and N. Boujemaa, "Unsupervised and Semi-supervised Clustering: a Brief Survey," *A Review of Machine Learning Techniques for Processing Multimedia Content, Report of the MUSCLE European Network of Excellence*, 2004.

[22] P. E. Jones, "Post code digit recognition employing low-level features " in *TENCON '97. IEEE Region 10 Annual Conference. Speech and Image Technologies for Computing and Telecommunications., Proceedings of IEEE*, 1997, pp. 523-526.
[23] Zhao, "What is semi-supervised learning?," dmml ASU presentation2005.
[24] Z. Xiaojin, "Supervised Learning Literature Survey," Computer Sciences, University of Wisconsin-Madison2005.
[25] X. Zhu and A. Goldberg, "Introduction to Semi-Supervised Learning. Synthesis Lectures on Artificial Intelligence and Machine Learning," 2009.
[26] E. Song, D. Huang, G. Ma, and C. C. Hung, "Semi-supervised multi-class Adaboost by exploiting unlabeled data," *Expert Systems with Applications*, vol. 38, pp. 6720-6726, 2011.
[27] J. Rennie. (2008). *The 20 Newsgroups data set*. Available: http://qwone.com/~jason/20Newsgroups/
[28] K. Nigam, A. McCallum, and T. Mitchell, "Semi-supervised Text Classification Using EM," *Semi-Supervised Learning. MIT Press: Boston*, 2006.
[29] N. Kamal, M. Andrew, T. Sebastian, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," *Machine Learning*, vol. 39, pp. 103-134, 2000.
[30] SAS, "SAS(R) Text Miner(TM) 4.2 Reference," 2009.
[31] K. Lang, "NewsWeeder: Learning to Filter Netnews," in *Proceedings of the 12th International Machine Learning Conference*, 1995.
[32] M. Porter. (2006). *The Snowball Porter stemming algorithm*. Available: http://snowball.tartarus.org/index.php
[33] G. K. Zipf, *Selected Studies of the Principle of Relative Frequency in Language*. Cambridge, MA.: Harvard University Press, 1932.
[34] D. M. W. Powers., "Applications and Explanations of Zipf's Law," in *Proc. NeMLaP3/CoNLL*, USA, 1998, pp. 151-160.
[35] T. Berka and M. Vajtersic, "Dimensionality Reduction for Information Retrieval using Vector Replacement of Rare Terms," in *Proceedings of the Ninth Workshop on Text Mining Eleventh SIAM International Conference on Data Mining*, Mesa, AZ, 2011.
[36] H. Luhn, "A statistical approach to the mechanized encoding and searching of literary information," *IBM Journal of Research and Development* vol. 1, pp. 309-317, 1957.
[37] K. Spark-Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, pp. 111-121, 1972.
[38] G. Salton and M. McGill, "Introduction to Modern Information Retrieval," *McGraw-Hill*, 1983.
[39] P. Soucy and G. W. Mineau, "Beyond TFIDF weighting for text categorization in the vector space model," in *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, San Francisco, CA, USA, 2005, pp. 1130-1135.
[40] W. HC, L. RWP, W. KF, and K. KL, "Interpreting tf–idf term weights as making relevance decisions," *ACM Transactions on Information Systems*, vol. 26, pp. 1-37, 2008.
[41] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing and Management*, vol. 24, pp. 513-523, 1988.
[42] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, pp. 993-1022, 2003.
[43] G. T. L. and S. Mark, "Finding scientific topics," *Proceedings of the National Academy of Sciences*, pp. 5228–5235, 2004.

[44] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, pp. 391-407, 1990.

[45] H. Thomas, "Probabilistic latent semantic analysis," in *Proceedings of the 15th Annual Conference on Uncertainty in Artificial Intelligence*, San Francisco, CA, 1999, pp. 289-296.

[46] D. Mimno, H. M. Wallach, E. T. M. Leenders, and A. McCallum, "Optimizing Semantic Coherence in Topic Models," in *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, 2011, pp. 262-272.

[47] T. Griffiths, "Gibbs sampling in the generative model of Latent Dirichlet Allocation," 2002.

[48] M. Steyvers and T. Griffiths. (2011). *Matlab Topic Modeling Toolbox 1.4*. Available: http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm

[49] C. E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379-423 & 623-656, 1948.

[50] M. C. Thomas and A. T. Joy, *Elements of information theory*. New York: Wiley-Interscience, 1991.

[51] A. Aizawa, "An information-theoretic perspective of tf–idf measures," *Information Processing and Management*, pp. 45-65, 2003.

[52] I. Cohen, F. G. Cozman, N. Sebe, M. C. Cirelo, and T. S. Huang, "Semisupervised Learning of Classifiers: Theory, Algorithms, and Their Application to Human-Computer Interaction," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, vol. 26, pp. 1553-1567, 2004.

[53] X. Zhu, "Semi-supervised learning literature survey," *Technical Report 1530, Computer Sciences, University of Wisconsin-Madison*, 2008.

[54] A. McCallum and K. Nigam, "A comparison of event models for naive bayes text classification," presented at the AAAI-98 Workshop on Learning for Text Categorization, 1998.

[55] P. Cheeseman and J. Stutz, "Bayesian classification (AutoClass):Theory and results," in *Advances in Knowledge Discovery and Data Mining* ed: MIT Press, 1996, pp. 153-180.

[56] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *Journal of Machine Learning Research*, vol. 5, pp. 361-397, 2004.

[57] S. M. Kamruzzaman, "Text Classification using Artificial Intelligence," *Journal of Electrical Engineering*, vol. 33, 2006.

[58] A. Ng and M. Jordan, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in Neural Information Processing Systems*, Cambridge, MA, 2002.

[59] G. McLachlan and T. Krishnan, "The EM Algorithm and Extensions," *John Wiley & Sons, New York*, 1996.

[60] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society: Series B*, vol. 39, pp. 1-38, 1977.

[61] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with cotraining," in *Proceedings of the 11th Annual Conference on Computational Learning Theory*, 1998, pp. 92-100.

[62] T. Zhang and F. Oles, "A Probability Analysis on the Value of Unlabeled Data for Classification Problems," in *Proc. Int'l Conf.Machine Learning (ICML)*, 2000, pp. 1191-1198.
[63] Z. yao, *Business cases of SAS Programming and Text Mining*: China Machine Press, 2010.
[64] V. S. Stehman, "Selecting and interpreting measures of thematic classification accuracy," *Remote Sensing of Environment*, vol. 62, pp. 77-89, 1997.
[65] M. John, K. Francis, S. Richard, and W. Ralph, "Performance measures for information extraction," in *Proceedings of DARPA Broadcast News Workshop*, Herndon, VA, 1999.
[66] D. M. W. Powers, "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation," *Journal of Machine Learning Technologies*, vol. 2, pp. 37-63, 2011.
[67] B. Hjørland, "The foundation of the concept of relevance," *Journal of the American Society for Information Science and Technology*, vol. 61, pp. 217-237, 2010.
[68] B. T. S. Kumar and J. N. Prakash, "Precision and Relative Recall of Search Engines: A Comparative Study of Google and Yahoo," *Singapore Journal of Library & Information Management*, vol. 39, pp. 124-137, 2009.
[69] S. T. Corp., "Precision & Recall Tuning," 2013.
[70] *visual algorithms: precision and recall*. Available: http://www.filosophy.org/post/7/visual_algorithms_precision_and_recall/
[71] X. Li, Y. Y. Wang, and A. Acero, "Learning query intent from regularized click graphs " *Proceedings of the 31st SIGIR Conference*, 2008.
[72] S. M. Beitzel., "On Understanding and Classifying Web Queries (Ph.D. thesis). IIT.," 2006.
[73] Y. Yiming, "A study of thresholding strategies for text categorization," *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 137-145, 2001.
[74] S. Agarwal, S. Godbole, D. Punjani, and S. Roy, "How Much Noise Is Too Much: A Study in Automatic Text Classification " in *Data Mining, ICDM. Seventh IEEE International Conference* 2007, pp. 3-12.
[75] Z. Xiaojin, "Semi-Supervised Learning with Graphs," Doctoral Thesis, School of Computer Science, Carnegie Mellon University, 2005.
[76] T. T. Lu, "Fundamental Limitations of Semi-Supervised Learning," Master of Mathematics, Computer Science, University of Waterloo, 2009.
[77] D. Elworthy, "Does Baum-Welch re-estimation help taggers?," *Proceedings of the 4th Conference on Applied Natural Language Processing*, 1994.
[78] F. Cozman, I. Cohen, and M. Cirelo, "Semi-supervised learning of mixture models," *ICML-03, 20th International Conference on Machine Learning*, 2003.