8-8-2017

A Multi-label Text Classification Framework: Using Supervised and Unsupervised Feature Selection Strategy

Long Ma

A MULTI-LABEL TEXT CLASSIFICATION FRAMEWORK: USING SUPERVISED AND UNSUPERVISED FEATURE SELECTION STRATEGY

by

LONG MA

Under the Direction of Yanqing Zhang, PhD

ABSTRACT

Text classification, the task of metadata to documents, needs a person to take significant time and effort. Since online-generated contents are explosively growing, it becomes a challenge for manually annotating with large scale and unstructured data. Recently, various state-or-art text mining methods have been applied to classification process based on the keywords extraction. However, when using these keywords as features in the classification task, it is common that the number of feature dimensions is large. In addition, how to select keywords from documents as features in the classification task is a big challenge. Especially, when using traditional machine learning algorithms in big data, the computation time is very long. On the other hand, about 80% of real data is unstructured and non-labeled in the real world. The conventional supervised feature selection methods cannot be directly used in selecting entities from massive data. Usually, statistical strategies are utilized to extract features from unlabeled data for classification tasks according to their importance scores. We propose a novel method to extract key features effectively before feeding them into the classification assignment. Another challenge in the text classification is the multi-label problem, the assignment of multiple non-exclusive labels to documents. This problem makes text classification more complicated compared with a single label classification. For the above issues, we develop a framework for extracting data and reducing data dimension to solve the multi-label problem on labeled and unlabeled datasets. In order to reduce data dimension, we develop a hybrid feature selection method that extracts meaningful features according to the importance of each feature. The Word2Vec is applied to represent each document by a feature vector for the document categorization for the big dataset. The unsupervised approach is used to extract features from real online-generated data for text classification. Our unsupervised feature selection method is applied to extract depression symptoms from social media such as Twitter. In the future, these depression symptoms will be used for depression self-screening and diagnosis.

INDEX WORDS: Multi-label Text Classification, Feature Selection, Word2Vec, Natural Language Processing, Depression Symptoms, Social Media

A MULTI-LABEL TEXT CLASSIFICATION FRAMEWORK: USING SUPERVISED AND UNSUPERVISED FEATURE SELECTION STRATEGY

by

LONG MA

A Dissertation Submitted in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

in the College of Arts and Sciences

Georgia State University

2017

A MULTI-LABEL TEXT CLASSIFICATION FRAMEWORK: USING SUPERVISED AND UNSUPERVISED FEATURE SELECTION STRATEGY

by

LONG MA

DEDICATION

To my beloved wife, Ruohan and my parents for their love, support, and encouragement over the years.

ACKNOWLEDGEMENTS

I first would like to express my special appreciation and thanks to my advisor Professor Dr. Yanqing Zhang, who is a tremendous mentor for me. I would like to thank him for encouraging me to do research and allowing me to grow as a research scientist. His advice on both research and my career is invaluable.

I would like to gratefully and sincerely thank Dr. Matthew Turner and Dr. Jessica Turner for their guidance, understanding, and patience for my research works. Their mentorships were extremely helpful in providing a well-rounded experience consistent with my long-term career and personal goal. They encouraged me to not only grow as a computer scientist but also an independent learner.

I would also like to thank Professor Dr. Raj Sunderraman, Professor Dr. Zhipeng Cai, and Professor Dr. Xin Qi for serving as my committee members even at hardship. I also want to thank them for letting my defense be an enjoyable moment, and for their brilliant comments and suggestions, thanks to them again.

I am grateful to my friends and colleagues, Dong Ren and Zhibo Wang for academic collaborations and personal support. Their presence and encouragement were very important to me.

I also thank my co-authors: Dr. Peter Fox and Dr. Angela Laird for their guidance and help.

I am very grateful to the NIMH/NIH grant that supported me.

Last, I would like to acknowledge the continued financial support from the Computer Science Department, and the Brain and Behavior (B&B) fellowship at Georgia State University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 Background

Text classification [2] is a task of assigning classes or labels to a public text document. It provides a view of document collections and has many real applications. For example, each news can be annotated with distinct categories, such as political issues, sports, business, and so on. The text classification or annotation gives business companies insights for them to make reliable business decisions. On the other hand, searching the text categories online provides a convenient way for people to choose their interests. Another interesting application of text classification is spam detection, which classifies emails into two categories (spam and normal emails). The text mining and text classification promise huge economic and research benefits.

Our new framework focuses on the textual data classification. Many text mining methods normally extract keywords from a text corpus and use them for proceeding mining tasks [22, 24]. However, there is not a general vocabulary that indicates which words in the corpus are important, and which are not. In fact, a few specific words are useful in a document but meaningless to another document. A naïve but general method that is used in the text mining preprocessing step is to remove stop words and punctuations. In this case, we assume that stop words and punctuations are not important to text mining. While, there is not even a common stop word list for all text mining tasks. A method of only removing the stop words and punctuations may generate a massive number of features for the text classification. This issue may affect the classification performance if the number of training instances is relatively small.

In the text annotation, single-label classification is a common and simple problem to learn from a set of examples, each associated with a class label or category. The multi-class classification method assigns one category to each instance or data sample for each class label.

Nowadays, multi-label classification approaches are increasingly required by real applications, such as gene classification and article annotation. People explore many algorithms to improve multi-label classification. A common method is to convert a multi-label problem into single label tasks without considering the relationship between labels [12].

As machine learning has been widely used in a variety of fields, text mining is mostly related to machine learning applications. Many well-known machine learning algorithms, such as Logistic Regression, Support Vector Machines (SVM) [28,29], *K*-Nearest Neighbors [80], Artificial Neural Networks (ANN), and many others, are used for text mining applications. Comparing with the fact that many new machine learning algorithms have been developed for text classification, there is a trend that data processing and transformation become more and more important. Even in many cases, the need for good quality of data is more than the need for building a strong machine learning algorithm. Such a case is very common in the data mining research.

Data can be any forms including textual data, digital data, streaming data and so forth. Many of these kinds of data are public, but others are not. The source of public data, e.g. social media, which produces billions of data every day. They are public, free and innumerous, yet mining underlying knowledge of public data is difficult because of these data, in general, are complex and unstructured. The private data source is able to provide good quality of data, e.g. clinical data, which are more valuable and meaningful. However, these data are definitely more expensive and inaccessible than the public data. An interesting research topic in the data science field is the data properties. To utilize any types of data in our text mining framework, we not only focus on the labeled data for evaluating text classification performance, but unsupervised text mining algorithm is explored and developed. A few unsupervised or semi-supervised algorithms are used in the specific tasks, such as *K*-means [49]. The other difficulty in unlabeled data mining is the feature selection. Some supervised learning algorithm, like Random Forest (RF) [34-36], can be used individually to select important features, but it is a challenge to define which features are key features from the large-scale of unlabeled data. In this case, we cannot select features based on supervised learning; therefore, we develop an ensemble method to discover key features.

Text mining and machine learning require big enough data for information retrieval and classifier training. Recently, the big data has attracted more and more attention in many fields, such as scientific research and economical applications. The big data [45] even changes the way people working and thinking, and constructs a world in which business and computer science put their efforts to realize the value of data. In business, a company can find customers' actual interests through integrating and analyzing the big data. In the Artificial Intelligence (AI), the powerful machine learning and data mining algorithms are beneficial from a large dataset. Big data is a huge dataset with complex structures that are generated from many fields such as biology, medicine, social networks, and so forth. In fact, many data intelligence applications have highly competitive performance when training model from a large amount of data. However, that approach makes building a machine learning model more expensive. Training a machine learning system using large-scale data is a time-consuming task, it usually takes a couple of hours, even a few days to build a machine learning model. After the model has been built, the classifier also needs to be re-trained again in a specific period. Thus, it is not easy to process a big dataset using traditional data processing tools.

Considering the computation cost in mining the big data, we will use the machine learning and statistical learning strategies together to reduce the cost. Another method to decrease the computing cost of big data is to apply the parallel computing tools to process the big data. Currently, Hadoop [54] and Sparks [55] are open source big data computing frameworks that are adopted by many enterprises, such as Yahoo, Baidu. Spark provides a framework that is used for cluster computing [60] for many kinds of data type and file format. In addition, it supports HDFS which is a Hadoop stored filed type. By parallel computing, the driver node assigns tasks into cluster manager that user defines, the cluster master then divides the whole tasks into several partitions and spreads them out to worker machines. In this way, the computing cost will be decreased but keeping the computing performance stable.

1.2 Challenges for Multi-label Text Classification Framework

In order to properly classify the published documents or even a short text, it is essential to have automated categorization systems that can clearly label published articles. It is more convenient for researchers to search or ask relevant questions on this kind automatic classification system. Currently, such labeling or categorization requires the time and effort of professional experts who have strong background and experience. Manually annotating articles is a time-consuming work, and the annotation accuracy cannot be guaranteed either. Therefore, it clearly requires being automated given the rate of publication of research papers.

Another challenge in the text mining is the quantity of data. It is true that data is everywhere, you can find many kinds of data online. However, there are many other data which has good qualities but less quantity. In one of our project [27], we only have 247 article abstracts that were labeled by an expert annotator [1]. The amount of data that we can use in the text classification is relatively small. In order to build a classifier, we extracted 3,606 features from these 247 article abstracts. In this case, it is an example of an $n \ll p$ (n much less than p) problem: where the number of training instances (n) is much less than the number of features (p)

[5]. In fact, many types of high-quality data suffer from this $n \ll p$ problem; however, most methods in text mining have been developed for situations where there is a plenty of low-quality data, rather than a limited high-quality data [27].

Our research is not limited to build a supervised learning classifier by the golden-standard labeled data, the proposed framework can be used to process the real and unlabeled data. To collect data for specific text mining tasks, cleaning and formatting data are challenges in the preprocessing step. Besides, extracting key features and discovering the hidden information are other interesting topics. It is not easy to do unsupervised feature selection, but we can use the statistical methods to discover the important features. Because there is not a golden-standard vocabulary to evaluate selected features, it is another challenge in our framework.

The last challenge is the multi-label classification framework, each label in the training represents a different classification task, but the tasks are somehow related. Therefore, it includes cases that allow a collection of labels to be assigned to an instance. For instance, an article in a newspaper or wire service may be assigned to the different categories. Currently, the major approach to improve the multi-label classification performance is to transform the multi-label classification task into multiple single label tasks [39-41]. However, there are two existing issues need to be solved if using this problem transformation strategy. The first issue is that the feature space is large, and the other one is that how to keep the relationship among different labels on training.

1.3 Problem Statement

For text classification, the problem is two-fold [61]: (1) there is no pre-existing and reliable vocabulary to utilize in the sciences that guarantees that important ideas are always expressed in the same terms. In addition, (2) underlying ideas in research are rarely easily expressed in simple “keyword” terms; they are concepts that require multi-word explanations that contain multiple underlying concepts. Our challenge is to mine the scientific documents for the words that indicate the underlying concepts and then assign labels that make these concepts express explicitly [27]. In general, there is not a standard keyword collection to be used for all text classification tasks. For example, some keywords appear in a computer science paper do not play an important role in the biology article. Simply searching for a fixed list of keywords is not be sufficient for a complex text structure. In this situation, we can only remove the stop words and punctuations. However, this leaves a collection of features that are too large to use effectively as a starting point for applying standard text mining algorithms; most of the algorithms work better with lower feature dimensions [61]. The relative big feature space problem is particularly critical when the number of training examples is very small, as often happens in the case of expertly labeled documents used for supervised learning classification problems. In particular, many types of high-quality data suffer from this $n \ll p$ phenomenon [5], while most methods in data mining are proposed for situations where there is an abundance of relatively crude data. Thus, it is better to apply a feature reduction or feature selection technique in training data when feature space is relatively large [61].

Big data [45] can tell us more information, while it can also bring us the computing problem. Considering the situation when mining underlying information on a big dataset, it is needed to reduce the data dimension to make computation cost more efficiently, even effectively improves the classification performance. Because our previous works [27,61] showed the feature reduction and feature selection were helpful in the text classification, we would use the similar feature reduction techniques to process the other training dataset. Generally, feature selection methods are mainly based on the supervised statistical learning, such as model performance.

However, most data are not labeled, which are unstructured and complex, traditional supervised learning approaches are not useful for selecting features. To discover the key features of big data is also a challenge.

Our framework is used to implement the multi-label classification [2]. This is a type of problem where each instance can be labeled with, in principle, any combination of the labels is used in the task. In other work [3], we solved this problem by using a Multi-Instance Multi-Label (MIML) algorithm [4]. Then we used principal component analysis (PCA) [21] and stemming to pre-process the data before classification. PCA [21] and word stemming are utilized to reduce the feature dimension because we have a relatively larger number of features than that of the data instances. The experiments showed that the PCA components, used as features, allowed a dramatic reduction in feature space dimension. As reported in the paper [27], the PCA was a powerful technique for reducing the size of the feature space, while the performance was slightly improved.

1.4 Organization

The rest of paper is organized as follows. Chapter 2 provides a literature review of related work. Chapter 3 presents an overview of the framework. Chapter 4 presents the hybrid feature selection methods. Chapter 5 shows how to use the Word2vec to lower the feature dimension for huge training dataset. Chapter 6 illustrates a method to extract key features from massive unstructured data. Finally, Chapter 7 concludes the research work and points out the future work.

2 RELATED WORK

2.1 Multi-label Classification

In our previous collaboration project [27], we implemented the Multi-Instance Multi-Label (MIML) algorithm for the multi-label classification. MIML is proven to perform better for complex data with a large number of features than the general multi-label classification approaches [10]. Figure 2.1 [10] introduces the existing instance-label classification structures. Multi-instance learning and Multi-label learning are the *N*-1 and 1-*N* model; they are also used in text/image classification. The key in the MIML is to discover and keep the relationship between each label and corresponding instances. Zhou *et al.* propose the MIML algorithm for a successful image classification [8-10]. In their framework, the MIML problem is transformed into two tasks. The first solution is using multi-instance as the bridge, the goal is to learn a function $f_{MIL}\colon 2^{x} \times y \rightarrow \{-1, 1\}$. In this sign function, $f_{MIL}(X_{i,}y) = +1$ if $y \in Y_i$, and $f_{MIL}(X_{i,}y) = -1$ otherwise. Different from the first solution, the second solution is utilizing multi-label learning as the bridge. To learn a function $f_{MLL}\colon Z \rightarrow 2^{y}. For\ any\ Z_i \in Z, f_{MLL}(Z_i) = f_{MIML}(X_i)$ if $Z_i = \emptyset(X_i)$, $\emptyset\colon 2^{x} \rightarrow Z$ [8-10]. Based on the two solutions, Zhou *et al.* propose the MIMLBoost and MIMLSVM algorithms for supervised learning [10]. Their algorithms perform well both in multi-label training and single label (multi-class) training. We extend the MIMLfast algorithm designed by Zhou *et al.* [10] by providing the new bagging schemes [27], which are designed by the label combination and resampling with replacement. However, the classification performance is not improved well due to the limited dataset [27].

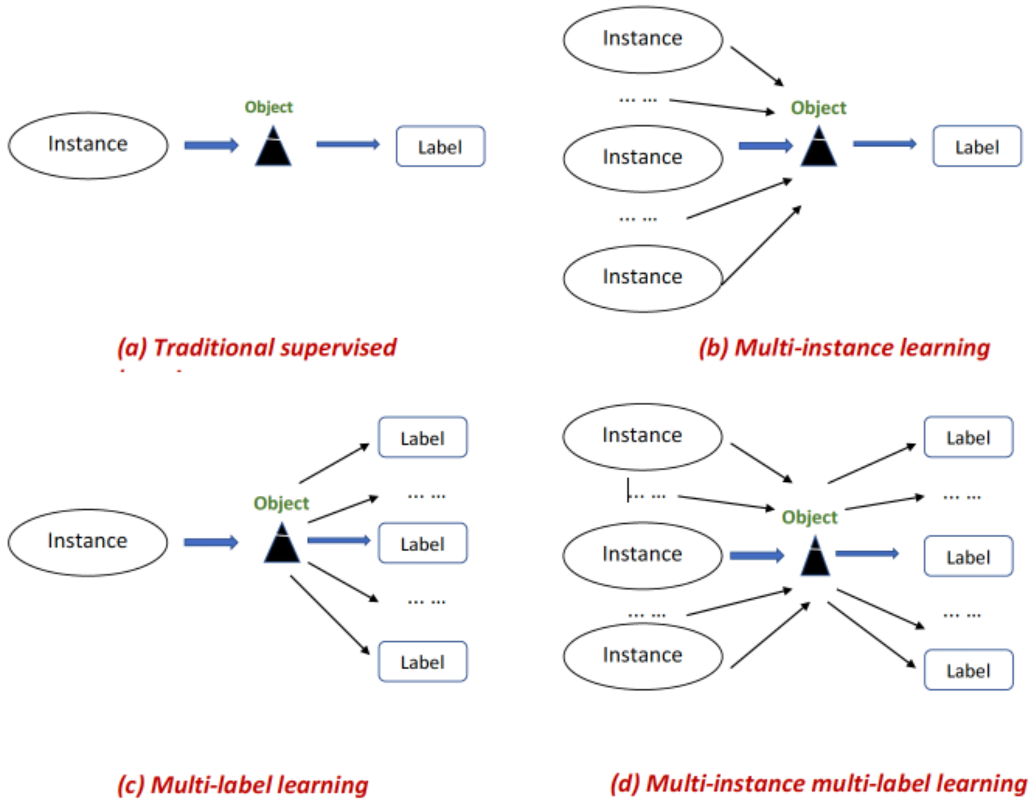

Figure 2.1 Four Different Learning Framework [10]

2.2 Reduce Feature Dimension

In machine learning, feature engineering is a popular area. The quality and quantity of features substantially influence the performance of machine learning methods. Noisy features negatively affect the performance of machine learning algorithms, while important useful features can improve the performance of supervised learning and unsupervised learning algorithms. There are a lot of features extracted from the articles or corpus, both feature reduction and feature selection methods can be utilized for feature dimensionality reduction. The primary goal is to select a subset from the original feature set. The advantage of reducing feature dimension includes: (1) reduce the computing time and storage space in training; (2) it is easier to implement the data visualization when feature dimension is reduced into 2D or 3D.

2.2.1 Feature Reduction

Feature reduction usually can be viewed as the feature extraction strategy. The transformation can be a linear mapping or non-linear projection. There are many supervised and unsupervised approaches to reduce the original feature dimension, such as the Principle Component Analysis (PCA) [21] and Singular Value Decomposition (SVD) [59]. Through projecting the initial feature dimension into a lower dimension, new features can be a good representative of the original feature set.

Data clustering is another a feasible method to reduce the feature dimensionality through grouping the similar data instances and select a few most important ones to represent each group. We assume that the data instances belonging to the same cluster are similar to each other, such an assumption can also be applied to feature clustering. In this approach, we can use a collection of features to represent the similar features.

2.2.2 Feature Selection

The straightforward way in a feature selection method is selecting the most important features from the original feature set, but the challenge is how to define the term “important”. In many text mining applications, we simply count the word frequency and choose the most frequent words as features in text mining. However, the frequent words usually are not important words according to their contributions to text classification task. Thus, we make the pre-existing feature selection approaches which are proven useful and reliable in the mathematics and statistic fields.

Recursive Feature Elimination (RFE) is usually used with the SVM classifier to recursively eliminate features according to each feature’s importance [30, 44]. Feature usefulness is defined in the RFE as the features’ weights in the SVM algorithm. Feature importance is determined by sequentially re-training an SVM classifier and, in each iteration, the least useful features are discarded [61]. RFE proceeds until the target number of features are left after discarding most of the least useful features. In addition, we can eliminate a fixed percentage of least important features rather than only one feature in each step.

Select K Best (SKB) is a procedure that constructs the χ^2 (chi-square) statistic between each element of the feature space and the labels to determine which features are correlated with which labels [31]. Compute chi-squared stats between each non-negative feature and class, this score can be used to select the *n* features with the highest values for the test chi-squared statistic from χ. More specifically in the feature selection, we use it to test whether the occurrence of a special term and the occurrence of a specific class are independent. Thus, we estimate the following quantity for each term and then rank them by their scores. High scores on χ^2 indicate that the null hypothesis (H_0) of independence should be rejected and thus that the occurrence of the term and class are dependent. If they are dependent, then we select the feature for the text classification. Thus, it rejects features with the smallest χ^2 statistics [32,33].

Many ensemble learning methods for classification or regression can be used as a feature ranking method if a relevant importance score can be defined. Random Forests (RF) [34-36] is a classifier that includes two methods: bagging and a random subspace. Suppose we have *n* number of trees in a forest, then we create *n* datasets created from randomly resampled data in an original dataset with replacement. In order to build a tree, we randomly select subsets of features. There are a lot of methods to calculate the feature importance, for example, we can calculate the entropy for each node. The averages of these scores for features are used to order them by importance.

Combinations of methods of individual feature selection strategies were proposed by researchers. Li *et al.* introduced an approach of combining multiple feature selection techniques by using the Combinational Fusion Analysis (CFA) [37]. Li *et al.* [37] showed that a combination method could outperform an individual feature selection method if each one had scoring function [61]. In another paper, Neumayer *et al.* [38] also presented the results of a combination of individual feature selection methods. Due to these combinations of methods performed better than a single feature selection method, we will evaluate whether combinations of our feature selection methods can improve performance compared with a single method [61].

We used RFE and SKB individually and sequentially, also in different orders. The first method in a sequence selects a subset of the original features, and then the second selects a smaller subset from the selected subset. We used Random Forest as a filter to select feature subsets with the procedure described above. Because RF generates random subspaces and a random number of features are selected, it was not used as the first method for feature selection [34]. This is due principally to the random (with replacement) missing features from the original feature space. However, RF can be successfully used after either RFE or SKB [61].

Above feature selection methods are based on the supervised learning strategy, however, we are usually required to select features from millions of unstructured and unlabeled data. Such task is a big challenge if people do not have domain knowledge or professional working experience in selecting "keywords". In fact, a limited number of NLP techniques are used to identify the important entities, because they are very limited to specific uses. Kim *et al.* [68] proposed an evolutionary local selection algorithm (ELSA) for large-scale feature selection with unsupervised learning. Under the *K*-means guidance, Kim *et al.* [68] use ELSA to search all possible combinations of features and numbers of clusters.

2.3 Artificial Neural Network

Artificial Neural Network (ANN) [81] is a popular machine learning algorithm for many applications. ANN processes information in a similar way the human brain does. The network is composed of several or many of highly interconnected neurons working in parallel to solve a specific problem. The neural network allows complex nonlinear relationships between the response variable and its predictors. Normally, ANN consists of two or three hidden layers, but the deep neural network usually contains a large number of hidden layers and each layer consists of a number of neurons [81]. The type of neural network can be viewed as deep multi-layers if the number of layers is larger than five. In the recent decade, more advanced neural network structures have been developed for specific tasks. A deep neural network is not a simple neural network structure with multiple hidden layers, but a powerful framework. Recurrent Neural Network (RNN) [57] and Convolutional Neural Network (CNN) [56] have outstanding performance on various machine learning tasks compared with other machine learning algorithms. Furthermore, RNN and CNN are used in the advanced artificial intelligence project, such as the self-driven car and Robert.

The ANN [81] is not only used for machine learning tasks, but also used for the natural language process. Bengio *et. al* [69] proposed the terminology "word embedding" and used training data to build an artificial neural network model. The feedforward neural network takes the words from the corpus as inputs and converts them into a lower dimension space. Through back-propagation with a fine tune, the word embedding is generated. The neural network language model (NNLM) can learn a probability distribution over words in the corpus. The NNLM is trained to produce the vector representations of a word to capture the semantic information. Through iterating over the whole document, every word is learned and represented by a vector with a fixed length. Figure 2.2 illustrates the structure of the Bengio's NNLM, NNLM is similar to the regular ANN but an addition projection layer is included. The softmax [82] activation function is used in the output layer; thus, the major computing cost of NNLM is between the projection layer and the output layer.

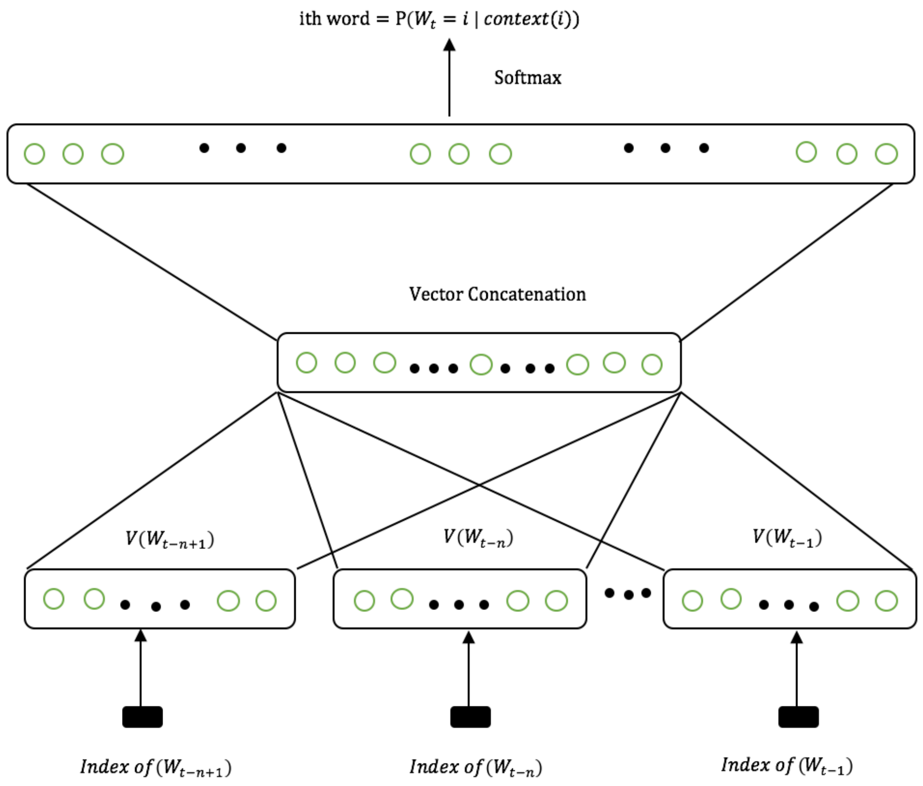

Figure 2.2 Structure of NNLM [69]

Word2Vec [46-48] similar to the NNLM can also be used to learn vector representations of words, which is known as word embedding. The most straightforward but powerful way to represent words in articles is to transform words into word vectors, then they can be used to train the statistical model to calculate the relationship among words from a mathematical point of view. Especially in the neural network, the input data are the word vectors. There are two common learning models in neural network training, the Continues Bag of Words (CBOW) and

Skip-gram [47]. In the NNLM model, input word vectors are concatenated into the projection layer, so the word vector size in the projection layer is $N \times M$, N is the number of words surrounded by the word *w*; M means vector size for every input word. However, the CBOW in the Word2Vec only sums and averages the input vectors into the projection layer instead. The Skip-gram is different from the *N*-gram [66] model in the NLP, the former one predicts word *w* by the surrounding words; while, the *N*-gram predicts word *w* only calculate previous *N* words. Thus, the Skip-gram is able to capture the relationship in the surrounding words, but the word order is ignored.

2.4 Depression Diagnosis System

In our research, we build an AI-based depression diagnosis system by developing an algorithm to extract and summarize the uncommon but potentially helpful depressive symptoms from the social media data. We explore Twitter and Web Blogs to collect depression-related data. Although the online data is innumerous, the social media data is complex and unstructured, data processing and data analysis are not easy. To clean the data collected from the social media, we apply the NLP tools to eliminate the noisy data. For data analysis, categorizing data can help us discover the underlying relationships of depression symptoms. Thus, the extracted depression symptoms through our framework are used as references when recognizing the clinical depression. Earlier works for extracting depression symptoms on literature help people learn the knowledge of depression detection. Wang *et.al* [70] applied the Latent Dirichlet Allocation (LDA) [71] as the topic categorizing tool to many of texts on adolescent substance use. Through separating the collections of articles into distinct themes by LDA [17], the known depressive facts were captured. Their work demonstrated that the topic modeling could be a useful approach to learn knowledge of depression prediction on the structured documents. In contrast to Wang's work [71], our challenge is to collect and process the unstructured and more complicated social media data. Mitigating the negative effect of the noisy data is an important task. To extract the depression symptoms from the social media data, we use a hybrid method like the one [7]. The Word2Vec [46,47] is used to convert each word in the corpus to a corresponding vector. Moreover, their framework has an ability to group words that share semantic similarities by using a clustering method, *K*-means. We apply a similar but more advanced approach to extract the facts of depression in our research. The relationship of these depression symptoms can also be found from the data. Another research that conducted by the Wang *et al.* [13] is to discover depression symptoms by mining depression related publications. Wang *et al.* propose a hybrid machine learning method first eliminating the outlier publication which is not relevant to the depression symptoms extraction. Utilizing the document summarization and topic learning by LDA, more accurate depression symptoms that relevant to corresponding publications can be located.

Nowadays, machine learning is being used in bioinformatics and health care [84-86]; the machine learning algorithms can help a doctor do depression diagnosis. Doctors confirm patients' depression by inputting the symptoms into a medical machine, the results will tell the doctor if this patient is in the risk of depression or even classify the patient into a digressional patient category. A classification method was used in the social media mining [72]. Authors used the wider variety of features, such as bag-of-words. An approach was developed for depression diagnosis by analyzing the records of user's activities in Twitter [73]. The features were extracted from the history activities of users, such as depressive tweets' frequency. The data were collected from Twitter users who report that they were truly diagnosed with clinical Major

Depressive Disorder (MDD)[1] [74]. The classifier was built to predict if a person was vulnerable to depression. This paper [72] focuses on selecting the reliable depression symptoms for building an intelligent depression diagnosis system for medical doctors and a convenient depression self-screening system for ordinary people.

[1] https://about.twitter.com/company

3 MULTI-LABEL TEXT CLASSIFICATION FRAMEWORK

3.1 Introduction

The great motivation for building a multi-label text classification framework is that text mining is becoming popular and useful in our real lives. For example, the online query or some artificial intelligent Q&A systems. The object of text mining is a short phrase or even a corpus. Originally, most of the documents are classified manually by the experts with diverse knowledge. However, this work is limited by the large scale of textual data and shortness of human knowledge. Considering the time consuming and expensive labor, relevant machine learning algorithms and natural language processing tools can be applied to text classification tasks. Through training a classifier by the dataset under a supervisor, the classifier is built by the machine learning and statistics learning. Therefore, given a new and unknown document, the classifier can assign a corresponding category to each article or discover the underlying knowledge from the dataset. Many machine learning algorithms such as Random Forest, and SVM perform well on data mining, where the data type is not limited to the textual data; image data and digital data can also be used by machine learning algorithms. Especially, the neural network and its dependencies attract more and more attention because they are successfully utilized in many commercial applications and scientific research. Additionally, deep learning structures [63] have been very popular since they were developed. Many real projects use deep learning structures, such as RNN [57] and CNN [56] in image and text classification tasks. Machine learning algorithms push the current world into an artificial intelligence era.

Although a machine learning algorithm plays an important role in the many text mining tasks, training data quality is even more important to text classification and annotation. The "data quality" itself is a vague term, it is not simple to define what dataset has good quality or not. In the text classification task, we build a classifier that divides all documents into different categories. On the other hand, in the document clustering problem, the aim is to group the similar documents. Another challenge for text classification is the document quantity. In a real situation, we have good quality and gold-standard training data, however, the number of data instances are far less than that of features extracted from the original dataset. To avoid the overfitting risk and improve the classification performance, we design a method to reduce the feature dimension. Data dimensionality reduction can be fulfilled by feature selection or feature reduction strategies. We develop a new hybrid method of feature selection to reduce the feature space when it is used in the big textual dataset; Besides, our framework can select features by supervised and unsupervised approaches for text classification task.

3.2 Overview of Framework

In this paper, we introduce a multi-label text classification framework that is used for the different types of text data. Our multi-label classification framework includes two major components. One is used to deal with labeled data with feature reduction, the other one discusses a method to extract features from unlabeled online-generated data. To implement the multi-label text classification, we develop new approaches to make our framework powerful when the data size is large. The section 1 to 5 present new techniques to reduce the feature dimension in training data. To evaluate the proposed framework, we apply the 10-fold cross-validation and the F1-micro score to test framework's performance on the large-scale textual dataset. The sixth section introduces a method to extract key features from social media and then use these features to future text classification or prediction tasks. Figure 3.1 shows the primary components of the proposed framework.

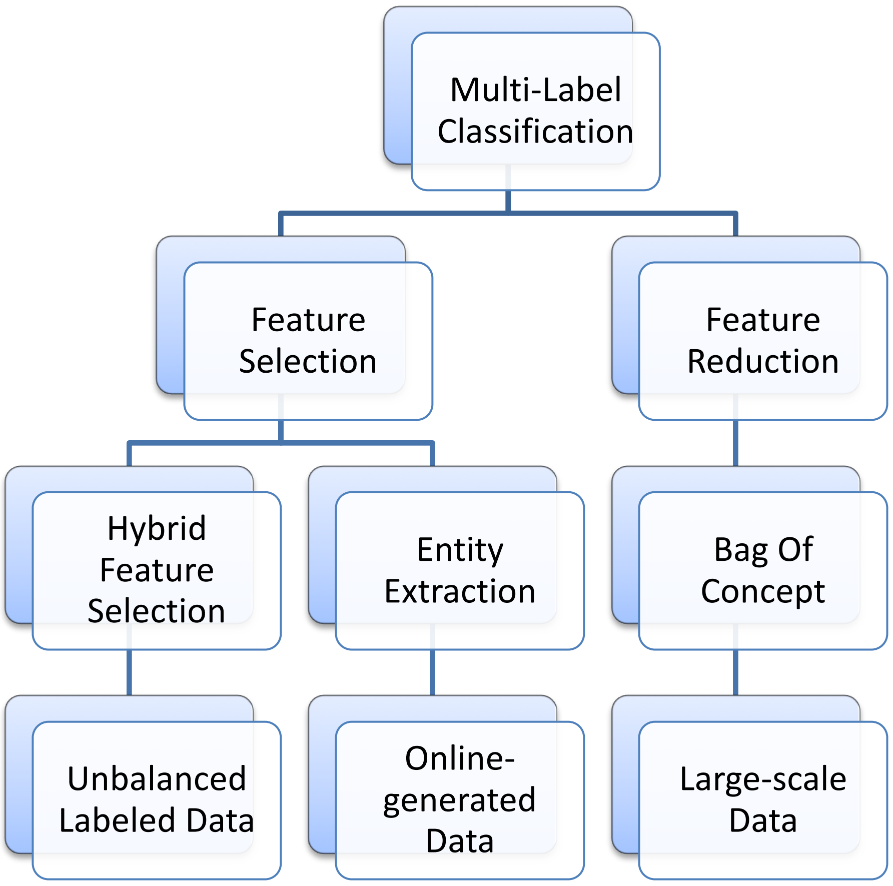

Figure 3.1 Overview of Proposed Framework

The main purpose of our framework is to annotate a document, such as an article, a tweet or a web blog. The left part of the framework is to reduce the training data dimension by employing the entity extraction and hybrid of feature selection under unsupervised and supervised learning guidance. Another highly competitive way to reduce the feature dimension is that we combine the Word2Vec [45] with a clustering algorithm, *K*-means, to achieve this goal. This semi-supervised learning approach uses a novel way to project each document into a new lower dimension.

To deal with the multi-label classification task, we built new bagging schemes in the MIMLfast algorithm for training [27]. Apart from the traditional binary label classification task, the new MIMLfast algorithm keeps the correlations among the multiple labels. The proposed bagging schemes are constructed based on the multiple different label combinations [27]. Besides, we applied the feature reduction (PCA) to reduce its feature dimension. This work was done in our previous project, we obtained the valuable results and proved that this approach was feasible though the performance was not improved significantly. In addition, we used Binary Reverence (BR) [16] and One-vs-all [11] together to be fitted into the LinearSVC [32] algorithm to implement the multi-label classification.

4 HYBRID FEATURE SELECTION

Our previous work [27] utilized the PCA to reduce the original feature dimension to a lower feature space. Unfortunately, the performance was not improved substantially compared with the old results in the paper [1]. Because this project [27] was suffered from the $n \ll p$ problem and there are some research works show the combination of the method of single feature selection approaches perform better than the individual feature selection method, we propose a new hybrid feature selection method to improve the multi-label classification.

4.1 Introduction

The reason we use the feature selection in the text classification is that we can make training more effectively through eliminating the noisy and less useful features. This approach is able to reduce the dimensionality of the training data. In this project [61], we applied a hybrid feature selection method to process the dataset used in the previous projects [1, 27]. The training data in this research are the abstracts of 247 published human neuroimaging journal articles and their corresponding metadata labels. The text of the abstracts was obtained from the PubMed publication database system (www.ncbi.nlm.nih.gov/pubmed) [1, 25]. These metadata labels were created as part of the BrainMap database (www.brainmap.org) and were annotated using the standard set of Cognitive Paradigm labels (www.cogpo.org) [61]. These labels were originally added to the BrainMap database by trained expert annotators, based on their readings of the entire text of each journal article [1, 27].

In this research, there are eight label dimensions, each dimension has multiple labels represented in our dataset, such as Behavior Domain Labels (BDL, 40 labels), Stimulus Type Labels (STL, 17 labels), Instruction Type Labels (ITL, 14 labels) and others [27, 61]. Because this is a multi-label classification task, each training instance may have more than one labels.

Our classification task is difficult, because (1) the training data (247 article abstracts) is really small, but the feature dimension (2,317 words for training) is much more than the data instances; and (2) the number of labels in most label dimensions is large. We proposed a new multi-label classification algorithm, but performance was not improved well [27]; thus, in this project, we applied a hybrid of feature selection methods to increase the classification accuracy.

4.2 Data Preparation

At first, the stop words and punctuations were removed from the 247 article abstracts by using the Natural Language Tool Kit stop list (www.nltk.org) stop words list [20]. After removing them, there are 3,606 words remaining. We use these 3,606 words to build a vocabulary. Each abstract is then represented by a bag-of-words vector through the one-hot encoding: for each article, we index words from the article in the vocabulary and then simply count the occurrences of each word. Because most abstracts are lengths from 100-300 words so these vectors are sparse; each vector has less than 10% non-zero elements in the vector representation [61]. Before forming the word counts, we reduce the number of words by using word stemming as a preprocessing step; this procedure maps morphological variants onto their stems. For instance, the words "argue", "argued", and "arguing" all have the same root "argu-". Even though "argu-" is not a real word, we consider it the stem of all those words, and it represents the general underlying meaning in all the variants [61]. The software we used for word stemming and lemmatizing is a Natural Language Toolkit (NLTK) text mining library [20]. We used the Porter Stemmer and WordNet Lemmatizer, both of them are built in the NLTK Stem package [20]. Besides, we are only interested in the words at least more than length 2. Through above data preprocessing, the vocabulary size was decreased from 3,606 to 2,317. At this step, the original feature dimension is already reduced significantly. This results in a final data matrix with 2,317 rows (one per word) and 247 columns (one per abstract; these are the count vectors) and with the counts of words present in each abstract populating the body of the matrix. This matrix is the input on the following steps.

4.3 Multi-Label Classification

In our previous project [27], we utilized the MIMLfast algorithm [10] to implement the multi-label classification. However, the MIMLfast algorithm did not perform well on our dataset because it was developed for a large dataset; we then used other multi-label classification approaches [14, 15]. One of these methods, Binary Relevance (BR) decomposes multi-label problems with *n* labels into *n* independent binary classification tasks [16, 41]. In the BR, labels are predicted independently and label correlations are not considered. In order to use a binary relevance method, we implement One-versus-the-Rest (OvR) multi-label strategy on our training data. In this approach, the classifier predicts multiple labels by fitting training instances with labels against all instances without the label [61]. BR is a meta-method that requires an underlying binary classifier. We used the LinearSVC (Linear Support Vector Classifier) implemented in Scikit-Learn [32]. This classifier is based on LIBLINEAR [43].

4.4 Feature Selection Methods

In this section, we apply three individual feature selection methods to our training data, Recursive Feature Elimination (RFE), Select K Best (SKB) and Random Forest (RF). However, we used RFE and SKB individually and in different orders. The first method in a sequence selects a subset of the original 2,317 words, and then the second method selects a subset from the selected subset [61]. Because RF generates random subspaces, it was not used as the first method for feature selection [34]. It is due primarily to the random (with replacement) missing features from the original feature space. However, RF can efficiently be used after either of the other methods. The purpose for using the hybrid feature selection method is that we can hierarchically select the important features in the sequence; besides, we build a voting system that can select majority features from features selected by different feature selection methods.

- **RFE** – The RFE method is used as the only feature selection method. It was run five times, selecting 50, 100, 150, 200, and 250 features and these were used as inputs for the BR Linear SVC classifier.
- **SKB** – The SKB method is used only as the feature selection method. The details are as for the RFE method above.
- **RFE (300) → SKB**– First, the RFE method selects the 300 most important features; then from these 300 features, SKB selected the 50, 100, 150, 200, and 250 most important ones. These five final feature sets are used for the classification as above.
- **SKB (300) → RFE** – The same as the previous procedure, but in the other order.
- **RFE (50-250) → RF** – In this condition, RFE first selects a fixed size subset of the original features; then RF is applied to this subset. Because of the randomness of RF, the final size of the feature subsets it selects are undefined; the nominal sizes (those produced by the RFE step) are reported. Note that the actual final feature sets were all much smaller, ranging from 14 to 73 features after the RF step; varying by dimensions.
- **SKB (50-250) → RF** – This is the same as the previous procedure, except with SKB as the first feature selector. Here the actual number of features produced by the RF step ranged from 7 to 68; these also varied by label dimensions.

Figure 4.1 [61] shows classification performance on all eight label dimensions using the above six feature selection methods and combinations. The results presented are 10-fold cross-validated F1-Micro scores. The F1-micro score is calculated by the precision and recall [42] and its score in the range of 0 and 1, with scores closer to 1 being better. Micro averaging the F1 scores, across instances, for better comparisons across datasets; however, more complex datasets intrinsically have lower F1 scores [61].

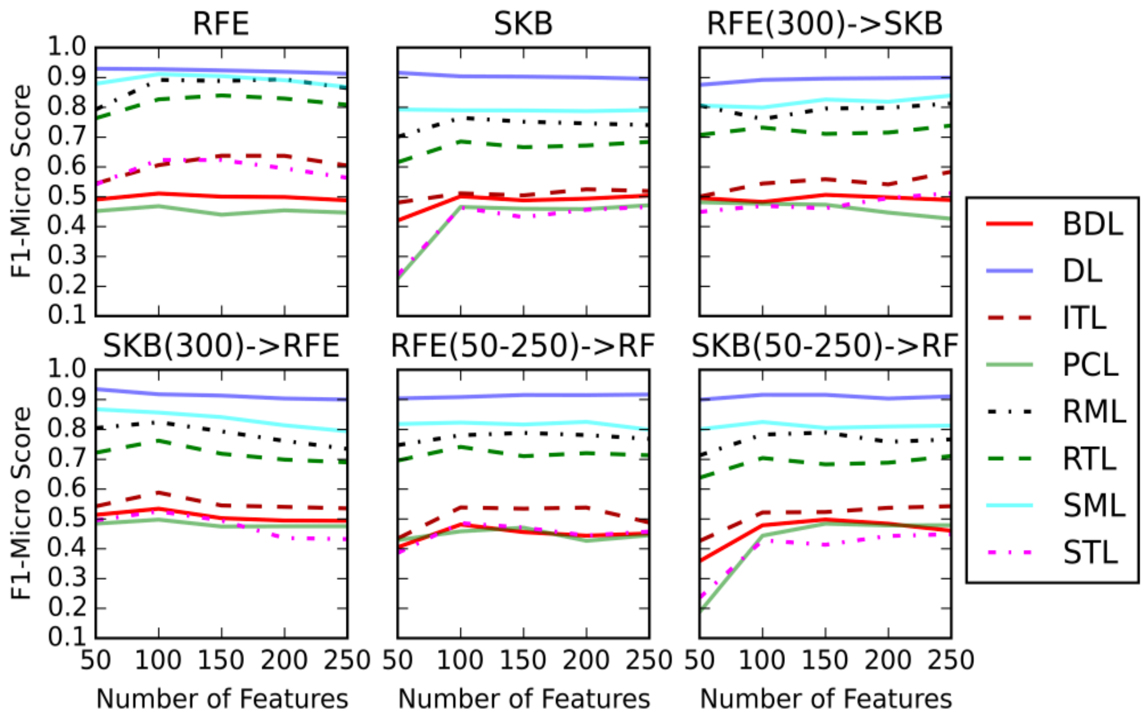

Figure 4.1 Feature Selection Methods on Eight Label Dimensions [61]

The results in Figure 4.1 [61] illustrate there is no distinguishable difference among the feature selection methods when applied to our training data. RFE and SKB (300) → RFE do better overall than the other methods, but most methods show some improvements over the same classification performance with the entire original feature space as an input [61]. (See Table 4.4 and the discussion of overall results in the section below.)

The last two conditions for SKB (50-250) → RF and RFE (50-250) → RF are shown in Figure 4.2 [61] in terms of nominal (input) numbers of features [61]. For comparison, Figure 4.2 [61] shows the same results plotted in terms of actual numbers of features. The main discovery is that the actual number of features in use in either of these conditions is always less than 75.

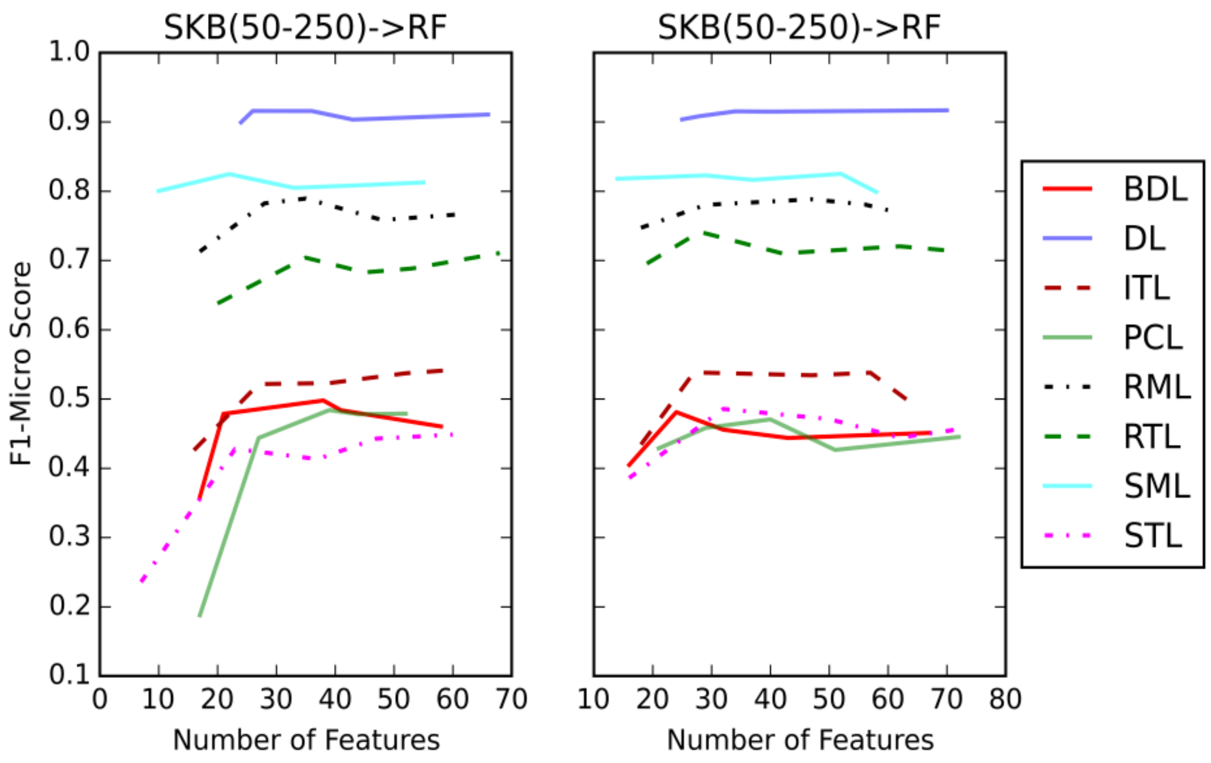

Figure 4.2 Feature Selection Methods Using RF [61]

4.5 Common Features Selected by Multiple Methods

Because feature selection methods choose features by different feature ranking methods, such as statistical tests or classification performance, the selected features are different. In this project, we considered two different combinations of this type:

- **Common Features (RFE&SKB)** – we first selected 50, 100, 150, 200, and 250 features using RFE and corresponding numbers from SKB and then paired up the corresponding sets. We obtained the intersection of the sets, and the features falling into the intersection were used as features for training the LinearSVC classifier.
- **Common Features (RFE&SKB&RF)** – Same as the previous method, but also with the top-ranked 50, 100, 150, 200, 250 features from RF and a three-way set intersection.

Table 4.1 [61] shows the number of common features that are selected from the combination of RFE and SKB. The first row in Table 4.1 is the top K selected features through RFE and SKB methods [61]. For example, we apply the RFE and SKB individually to select 50 features in DL, 14 common features are generated by RFE and SKB.

Table 4.1 The Number of Features Selected by Combination of RFE, SKB [61]

RFE & SKB	**50**	**100**	**150**	**200**	**250**
BDL	13	35	53	75	96
DL	14	36	53	82	115
ITL	12	31	51	75	101
PCL	10	31	56	75	99
RML	8	30	45	65	94
RTL	3	22	35	42	58
SML	12	39	65	85	117
STL	7	19	44	66	91

Table 4.2 [61] lists the common features that are selected from the combination of SKB, RFE, and RF. Because the RF selects features randomly, thus the number of common features in the combination of RFE, SKB and RF is usually less than that of the combination of RFE and SKB. In the experiment, we use these features for multi-label text classification task [61].

Table 4.2 The Number of Features Selected by Combination of RFE, SKB and RF [61]

RFE & SKB & RF	**50**	**100**	**150**	**200**	**250**
BDL	6	15	38	41	51
DL	6	13	18	25	34
ITL	9	17	28	34	47
PCL	7	20	26	38	49
RML	4	14	24	25	41
RTL	2	8	16	24	29
SML	10	18	24	25	34
STL	4	13	26	29	50

In Figure 4.3 [61], the *X*-axis is the number of features that are selected and used in classification, *Y*-axis is the F1-Micro score calculated by the classifier using the common feature sets that described above. The left-hand side shows the common features in five feature sets (50-250) that are selected only by RFE and SKB; while, the right-hand side shows the common features selected from the three feature selection methods, RFE, SKB, and RF.

In general, the performance improves as the number of features increase. Figure 4.3 [61] is plotted based on the actual number of features used for the classifier, not in the nominal 50-250 range. It is important to notice that the number of features in these conditions is much smaller than that in most other conditions presented here; the left the maximum number of common features is 117, and the right it is 51. Performance at the largest number of features in Figure 4.3 [61] is in the general range of performance in the previous figures, but there is no improvement, for the most part. On only one label dimension (Paradigm Class Labels, PCL) did this collection of methods achieve better performance than that in other conditions. 48 labels generated by PCL is a conceptually complex dimension compared with some of the other methods.

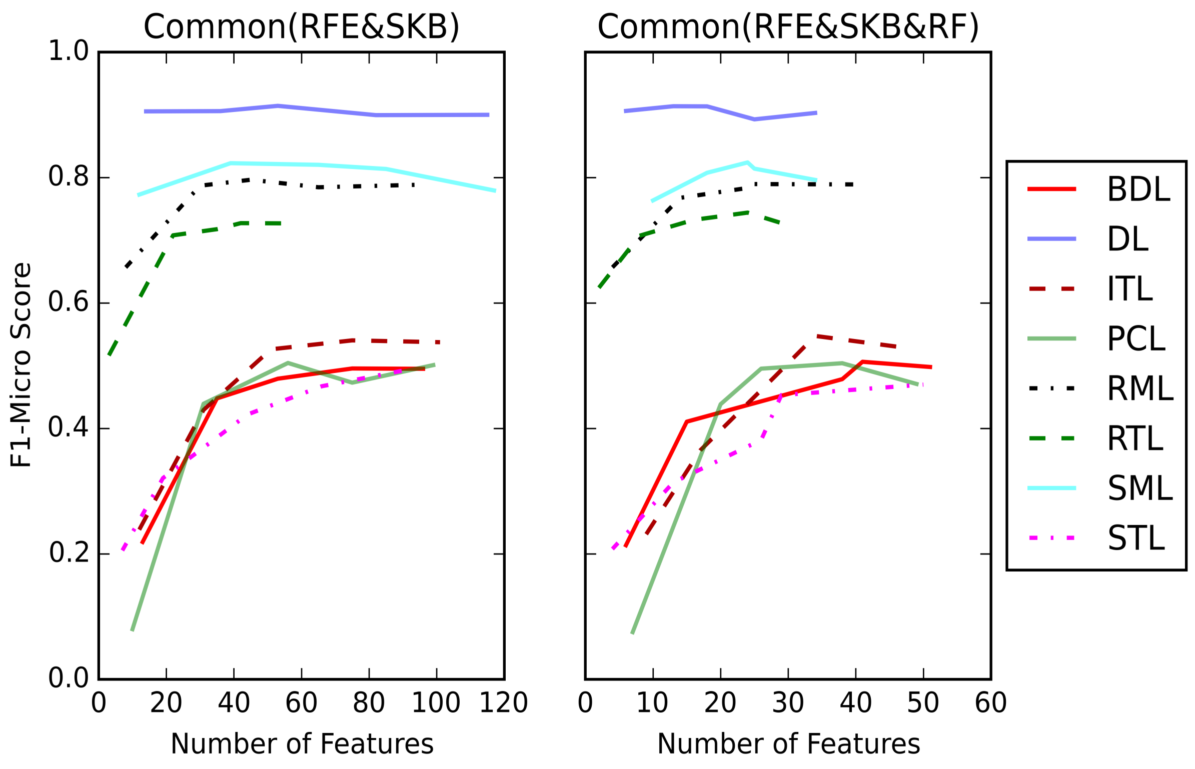

Figure 4.3 Common Features Among Eight Label Dimensions [61]

4.6 Overall Performance

Table 4.3 [61] displays all the results that performed by feature selection methods on the eight label dimensions. The number of selected features start from 50 through 250 as same as the above experiments.

Table 4.3 Feature Selection Performance on Eight Label Dimension

Methods	BDL	DL	ITL	PCL	RML	RTL	SML	STL
RFE	0.4904	**0.9292**	0.5440	0.4518	0.7924	0.7630	0.8788	0.5412
	0.5108	0.9277	0.6060	0.4683	0.8917	0.8265	**0.9107**	**0.6234**
	0.5003	0.9235	**0.6374**	0.4397	0.8881	**0.8395**	0.9044	0.6228
	0.4989	0.9188	0.6371	0.4543	**0.8936**	0.8291	0.8907	0.5951
	0.4878	0.9127	0.6043	0.4467	0.8639	0.8081	0.8678	0.5635
SKB	0.4204	0.9156	0.4804	0.2251	0.7008	0.6155	0.7925	0.2377
	0.5013	0.9036	0.5113	0.4657	0.7644	0.6852	0.7897	0.4637
	0.4878	0.9025	0.5045	0.4596	0.7515	0.6660	0.7892	0.4314
	0.4935	0.9000	0.5249	0.4585	0.7458	0.6717	0.7870	0.4563
	0.5038	0.8952	0.5189	0.4715	0.7402	0.6840	0.7905	0.4658
RFE (300) -> SKB	0.4956	0.8750	0.5005	0.4815	0.8059	0.7072	0.8055	0.4492
	0.4825	0.8917	0.5442	0.4764	0.7605	0.7318	0.7992	0.4695
	0.5062	0.8960	0.5587	0.4732	0.7956	0.7107	0.8259	0.4616
	0.4977	0.8975	0.5413	0.4468	0.7982	0.7151	0.8185	0.4955
	0.4889	0.8992	0.5839	0.4261	0.8126	0.7386	0.8391	0.5118
SKB (300) -> RFE	0.5134	0.9349	0.5421	0.4838	0.8044	0.7218	0.8675	0.4950
	0.5340	0.9177	0.5881	0.4974	0.8241	0.7621	0.8564	0.5251
	0.5023	0.9133	0.5449	0.4744	0.7933	0.7183	0.8411	0.4944
	0.4944	0.9035	0.5402	0.4753	0.7614	0.6987	0.8137	0.4358
	0.4935	0.8999	0.5352	0.4753	0.7347	0.6896	0.7937	0.4322
SKB (50-250)->RF	0.3584	0.8994	0.4260	0.1877	0.7129	0.6379	0.8004	0.2357
	0.4785	0.9159	0.5215	0.4437	0.7825	0.7040	0.8247	0.4278

	0.4978	0.9158	0.5229	0.4839	0.7894	0.6828	0.8048	0.4132
	0.4837	0.9034	0.5371	0.4780	0.7584	0.6883	0.8096	0.4427
	0.4603	0.9107	0.5425	0.4788	0.7665	0.7107	0.8127	0.4485
	0.4048	0.9037	0.4342	0.4287	0.7472	0.6955	0.8180	0.3859
	0.4810	0.9084	0.5383	0.4582	0.7803	0.7413	0.8228	0.4856
RFE (50-250)->RF	0.4556	0.9153	0.5342	0.4705	0.7883	0.7102	0.8163	0.4709
	0.4437	0.9149	0.5379	0.4264	0.7809	0.7204	0.8251	0.4442
	0.4510	0.9168	0.4876	0.4452	0.7684	0.7131	0.7989	0.4574
	0.2190	0.9055	0.2386	0.0800	0.6568	0.5164	0.7728	0.2055
	0.4479	0.9059	0.4294	0.4393	0.7870	0.7079	0.8229	0.3208
Common(RFE&SKB)	0.4793	0.9143	0.5262	**0.5044**	0.7965	0.7177	0.8203	0.4222
	0.4956	0.8995	0.5404	0.4730	0.7845	0.7273	0.8137	0.4673
	0.4951	0.9000	0.5375	0.5010	0.7884	0.7269	0.7794	0.4932
	0.2138	0.9062	0.2314	0.0753	0.6568	0.6243	0.7635	0.2078
	0.4108	0.9137	0.3651	0.4386	0.7676	0.7072	0.8076	0.3139
Common(Three)	0.4786	0.9135	0.4824	0.4953	0.7834	0.7325	0.8241	0.3816
	0.5063	0.8928	0.5475	0.5040	0.7896	0.7441	0.8142	0.4532
	0.4980	0.9030	0.5291	0.4709	0.7890	0.7274	0.7962	0.4699

Table 4.4 [61] summarizes the overall results of the study, feature selection improves the classification performance in general. In Table 4.4 [61], the first row shows the F1-Micro score computed by the classifier when applied to the entire feature space of all 2,317 words without any feature selection methods. The second row shows the best F1-Micro score obtained across all methods that are described previously [61]. In every case, at least one feature reduction strategy performs better than all the features used without selection, and it is even a dramatic

improvement. The smallest improvement is just over 0.05 F1-Micro units, and the largest is almost 0.18 [61].

Table 4.4 F1-Micro Scores with Wining Methods [61]

	Label Dimensions							
	BDL	**DL**	**ITL**	**PCL**	**RML**	**RTL**	**SML**	**STL**
F1-Micro Score (All Features)	0.4294	0.8830	0.4746	0.3672	0.7187	0.6808	0.7418	0.3983
Best F1-Micro Score	0.5340	0.9349	0.6374	0.5044	0.8936	0.8395	0.9107	0.6234
Method	SKB(300) →RFE	SKB(300) →RFE	RFE	Common (RFE&SKB)	RFE	RFE	RFE	RFE
Number of Selected Features	100	50	150	56	200	150	100	100

The last two rows show the winning method and the number of features that used to reach that score. In general, RFE alone and RFE proceeded by SKB achieved the best results and less than 200 features were used. The RFE is the wining method in most of the cases, but an interesting discovery in Table 4.3 [61] is that the combination of feature selection methods has better performance than individual feature selection one when the number of the label is large. Both BDL and PCL have more than 40 labels, and the winning feature selection method is the combination method.

4.7 Artificial Data Evaluation

We evaluated the proposed hybrid feature selection methods on the small dataset, which had 247 labeled article abstracts and 2,317 features. We showed these ensemble methods performed well in multi-label text classification, where the number of the instances is less than the number of features. In the next step, we will evaluate our hybrid feature selection methods on an artificial dataset, Genbase [58], to prove that the proposed feature selection method also performs well in this dataset.

The Genbase dataset contains 662 data instances and 1,186 features [58], the number of the labels is 27. Each data instance in the Genebase data is assigned by one or more labels, it is also the $n \ll p$ problem in the multi-label classification. Therefore, we applied the ensemble feature selections for the Genebase dataset to evaluate their performances. Table 4.5 shows the part of the overall performance on Genbase data classification. The second row is the F1-Micro score without using any feature selection methods. Because the data itself has good quality, the improvement is not substantial, but it proves that the ensemble feature selection can be used to solve the $n \ll p$ problem in the multi-label classification tasks.

Table 4.5 Multi-label Classification on Genbase Dataset

Methods	**F1-Mirco**
ALL (No feature selection)	0.9711055
RFE (100)	0.987105505
Random Forest	0.985112082
SKB (100)	0.987105505
RF-> RFE (100)	0.988674132
RFE (100) ->RF	0.970608959
SKB (100)-> RF	0.970608959
SKB (200)->RFE (100)	0.987105505
RFE (200)->SKB (100)	0.987105505

Figure 4.4 displays all the feature selection performance on Genbase dataset [58]. Although the improvement is not huge, the proposed hybrid feature selection methods can be utilized in many kinds of datasets.

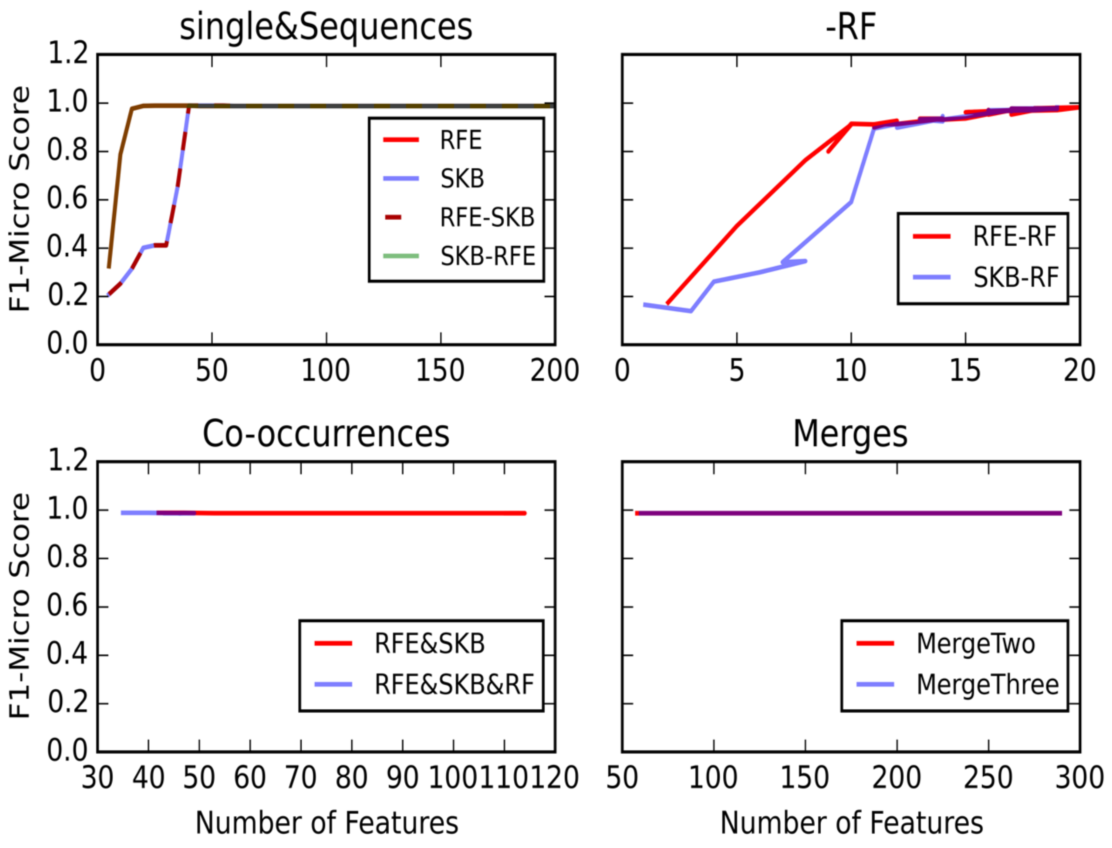

Figure 4.4 Multi-label with Feature Selection Performance on Genbase Dataset

4.8 Conclusion

This chapter presents an approach to feature reduction in complex textual feature spaces using standard methods both alone and in combination. The principal results are surprising, i.e., chaining together multiple methods does not necessarily produce an improvement. However, the results obtained do suggest that more work needs to be done in this area [61].

In this project, we mainly focus on the n ≪ p problem, the problem of classifying small quantities of high-quality data. The data may have an imbalance between *n* and *p*, that is, there are more features than training instances. Performance of the classification algorithm on the raw feature space is not convincing, but several combinations of feature selection methods can improve the performance dramatically. Besides, the proposed hybrid feature selection methods improved performance in another dataset, Genebase [58].

Currently, text classification takes a lot of time and effort by the limited number of experts with the requisite knowledge. This is a very expensive work, manually annotating documents becomes more difficult due to more documents are generated every day. People have to learn more and more knowledge on machine learning and text mining fields [61].

5 WORD2VEC PROCESS BIG DATA

Word2Vec [45, 46] is used in many applications, especially in the natural language understanding, such as machine translation and sentiment analysis. Most applications generate word vectors by the Word2Vec. Here, we use the Word2Vec to reduce the feature dimension in a relatively big dataset.

5.1 Overview of Framework

Figure 5.1 [62] shows a structure using Word2Vec to process a big text dataset, this is another method we proposed to reduce the feature dimension in the text classification. The general process of this framework: (1) the whole documents are used as input examples for Word2Vec to generate word vectors; (2) created word vectors are merged into different groups by calculating their distances; And (3) all documents vectors are transformed into a new lower feature dimension for text classification.

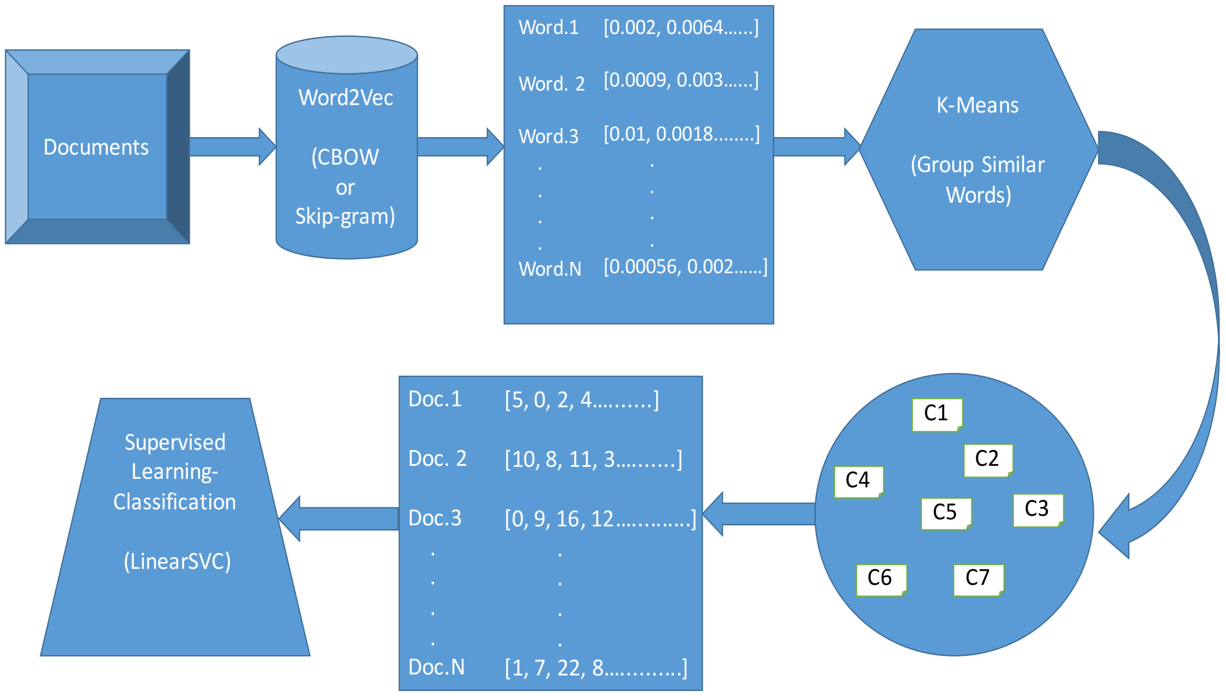
Documents
Word2Vec
(CBOW
or
Skip-gram)
Word.1 [0.002, 0.0064......]
Word. 2 [0.0009, 0.003......]
Word.3 [0.01, 0.0018........]
Word.N [0.00056, 0.002......]
K-Means
(Group Similar
Words)
C1
C2
C4
C5
C3
C6
C7
Doc.1 [5, 0, 2, 4..........]
Doc. 2 [10, 8, 11, 3..........]
Doc.3 [0, 9, 16, 12.............]
Doc.N [1, 7, 22, 8............]
Supervised
Learning-
Classification
(LinearSVC)

Figure 5.1Working Process [62]

5.2 Word2Vec Structure

Word2vec [45, 46] is not a single algorithm, but a three-layer neural network that processes the textual data. In fact, Word2vec contains two different learning models (CBOW and Skip-gram). Compared with the neural network language model, its output layer uses either the hierarchical softmax or negative sampling instead of the softmax activation function [45, 46]. Given a word, the skip-gram model predicts the neighboring words according to the given word. In contrast, the CBOW model predicts the current word if given the neighboring words in the surrounding window. Figure 5.2 illustrates the structure of Word2Vec. In the CBOW learning model, for example, the inputs are initialized word vectors, then they are summed and averaged as $\mathrm{Context(w)}$ into the project layer. In the output layer, Word2Vec originally builds a Hoffman tree based on the word frequency. Each leaf in the tree has a unique path from the tree root to it. Because the Hoffman tree is a binary tree, thus, there is a binary classification through passing a node to its children. The node *w*, Word2Vec calculates *w*'s probability of $\mathrm{P(w|Context(w))}$ according to the Maximum Log likelihood $\mathcal{L} = \sum_{w\in C} \log \mathrm{p}\left(w\middle|Context(w)\right)$ and combine this with binary classification (like sigmoid). In order to increase the probability of $\mathrm{P}\left(\mathrm{w}\middle|\mathrm{Context(w)}\right)$, Word2Vec uses the Scholastic Gradient Ascend (SGD) [83] to update the weights. Next, each word vector is updated based on its contribution to construct the vector of $\mathrm{Context(w)}$. At last, each word in the article is updated to produce the new vector. Therefore, the purpose of Word2Vec is to obtain a better representation for each word under two learning methods, CBOW and Skip-gram. After training all the words in the article, each word has a distinct vector with a defined length. We use the generated word vectors to calculate the word similarities between two words; thus, the underlying relationship among words can be discovered.

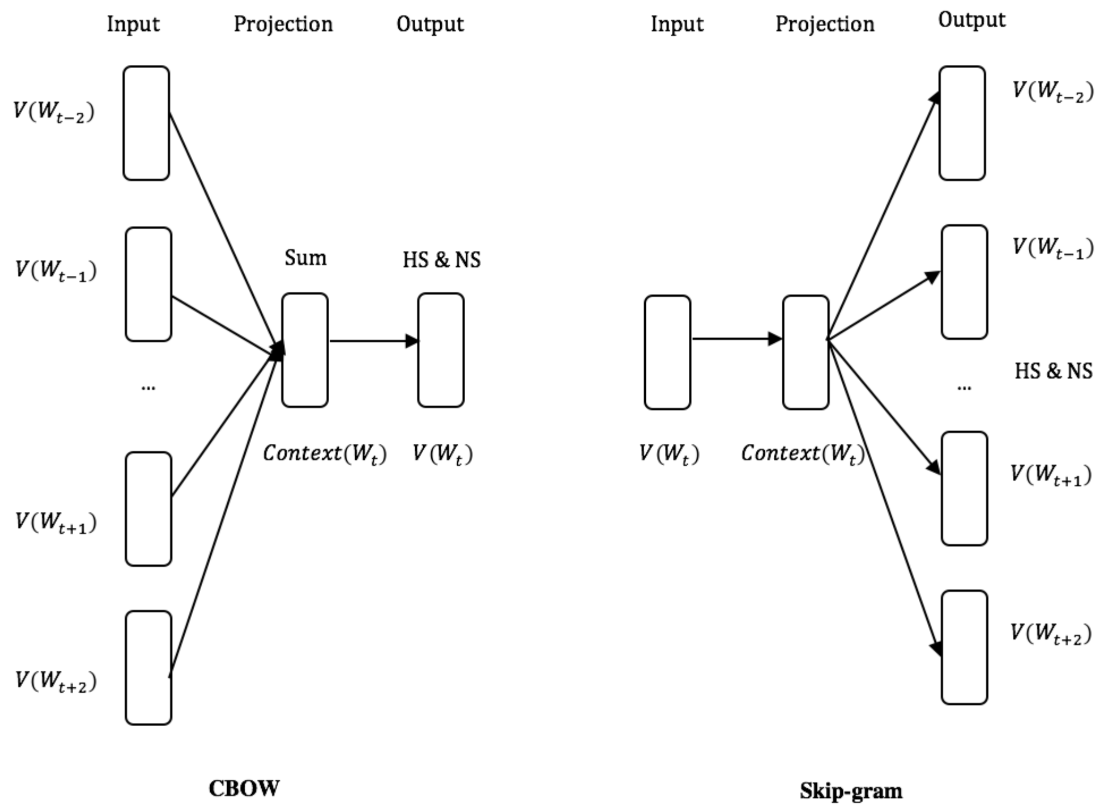

Figure 5.2 Structures of CBOW and Skip-gram [45, 46]

5.3 Word Similarity

The training data is the 20 Newsgroups dataset [50], which was originally collected by Ken Lang. The data is organized and split into 20 categories (newsgroup), and each category represents the distinct news domain, such as, sci.med and rec.autos. In our project, we only used a subset of the entire dataset, 11,314 training data. Although the training data size is not too big, it should be a good example to evaluate our method. In the future, we will apply our approach in the entire 20 Newsgroup dataset. Before inputting the training data to Word2Vec, we only removed punctuations for each training instance. Stop words remain in this project because we assume that they are useful to a piece of news. We apply the gensim [51], which is a python library to help us implement the Word2Vec model. We first build a vocabulary from the entire training data. To generate the word vectors, we employ the Skip-gram model because it has a better learning ability than CBOW if we do not take computing speed into account, see [46] for some details. After training, each word has attached a vector with a fixed length. Next, we construct a high dimensional matrix, where each row represents a specific word and the columns are the generated word vectors. As a result, the word has multiple degrees of similarity, it can be computed via a linear calculation. For example, vector ("Beijing") – vector ("China") + vector ("America") produces a vector that represents the word "Washington". Besides, the vector arithmetic can be represented as the format: vector ("Beijing") – vector ("China") = vector ("America") - vector("Washington"). Thus, Word2Vec calculates the similarity between words by the following.

$$\arg\max_{b^*}(\cos(b^*, b - a + a^*)) =$$

$$\arg\max_{b^*}(\cos(b^*, b) - \cos(b^*, a) + \cos(b^*, a^*))$$

Word2Vec generates two numerical vectors X and Y for two different words, the cosine similarity [65] between the two words is defined as the normalized dot product of X and Y:

$$similarity = \cos(\theta) = \frac{X \cdot Y}{||X||_2 \, ||Y||_2} \qquad \mathbf{(2)}$$

To test the word similarity, we have a few examples in Table 5.1 [62], and we only list the two most similar words and their cosine distances. For example, given a word "computer", we obtain the two similar words "evaluation" and "algorithm" with their distances to "computer". Although there is not a gold-standard vocabulary to evaluate the quality of word vector in NLP, the results of our experiment show that the Word2Vec finds the semantically related words successfully.

Table 5.1 Similar Words and Cosine Distance [62]

Words	**Words Similarity**			
	1st Word	***Cosine Distance***	***2nd Word***	***Cosine Distance***
computer	*evaluation*	*0.9325*	*algorithm*	*0.9292*
president	*property*	*0.9306*	*population*	*0.9123*
American	*Churches*	*0.9438*	*Greece*	*0.9365*
bank	*finance*	*0.9502*	*interests*	*0.9473*
football	*team*	*0.931*	*Champion*	*0.9061*

5.4 Word Clustering

In the natural language, a word or a combination of words usually express the meanings that are not difficult to understand for humans. However, a computer cannot understand the meaning a word very easily. Therefore, we should train the computer to understand our natural language in the mathematics perspective. A word can be represented by a vector for specific use in the computer, e.g. text mining.

Word2Vec [45, 46] produces the good quality of word vectors after training a whole document and can capture the semantic relationship between words by the arithmetic operation on word vectors. In mathematics, statistics, and physics, a vector presents a geometric object that has direction and length [53]. Because each word has multiple degrees of similarity and it can be computed by a linear calculation, we use Word2Vec for text mining.

Similar words tend to be close to each other, to group semantic similar words, we use the *K*-means [49, 75] to partition the *N* objects into *K* (*K* << *N*) clusters depending on their geometric locations. Initially, *K*-means algorithm randomly generates *K* seed points as the centers of *K* clusters, each object is calculated by the Euclidean distance between its location and the *K* seed points. Then assigning the object to one cluster whose distance from the center of the cluster is the minimum of all the seed points. Next, recalculate the new cluster centers via the formula:

$$v_i = \frac{1}{S_i}\sum_{j=1}^{s_i} x_j \qquad \textbf{(3)}$$

where s_i is the number of objects in the i^{th} cluster [75]. After new cluster centers are found, compute the distance between each object and the new centers again. Above iteration is operating until there is no object is reassigned to a new cluster.

In NLP [51], semantic similar words are grouped together [52] for a collection of these words. The word clustering helps discover the relationship between words. Given different *K* values, we have *K* different number of clusters. Therefore, all words are grouped into *K* clusters and each word attaches its index to the cluster.

Table 5.2 [62] shows a few examples of contents of clusters when *K* equals to 500. Although there is not a general vocabulary about the categories for each word, we can find the similar words that are almost correctly grouped into corresponding clusters as shown in Table 5.2.

Table 5.2 Cluster Contents [62]

Cluster Index	**Cluster Contents**
1	my, has, is, in, the, had, for…
2	components, speed, trigger, applications, stage, developers, manufacturers …
3	Medical, cigarettes, usma, smoked, food, Laboratory, chewing…
4	population, federal, laws, treasure, crime, organizations, Pope…
5	bank, financial, interests, client, card, credit …

5.5 Bag of Concept (BoC)

In Figure 5.3, we show that the original Bag of Words (BoW) problem is transformed to the Bag of Concepts (BoC). Each document in the text classification is represented by the fixed length vector. In fact, the number of instances (11,314) is much less than the number of original features (61,189). Thus, we apply the feature reduction method to reduce feature dimension by converting the BoW to BoC. The primary purpose of this approach is using some topics to represent a document with a small number of words in the vocabulary [62].

This transformation is implemented by grouping original words into *K* clusters, each cluster can be viewed as one topic. The contents (words) of every cluster are the semantic similar words. At last, the original feature dimension (*61,189*) is projected into a new and lower dimension (*K*). As a result, the reduced feature dimension is 11,314 * *K* (*K* << 61,189). In this case, each document is represented by several topics. Next, we use the dimension-reduced matrix in multi-class classification, where values of *K* are 500, 1000, 1500, and 2000.

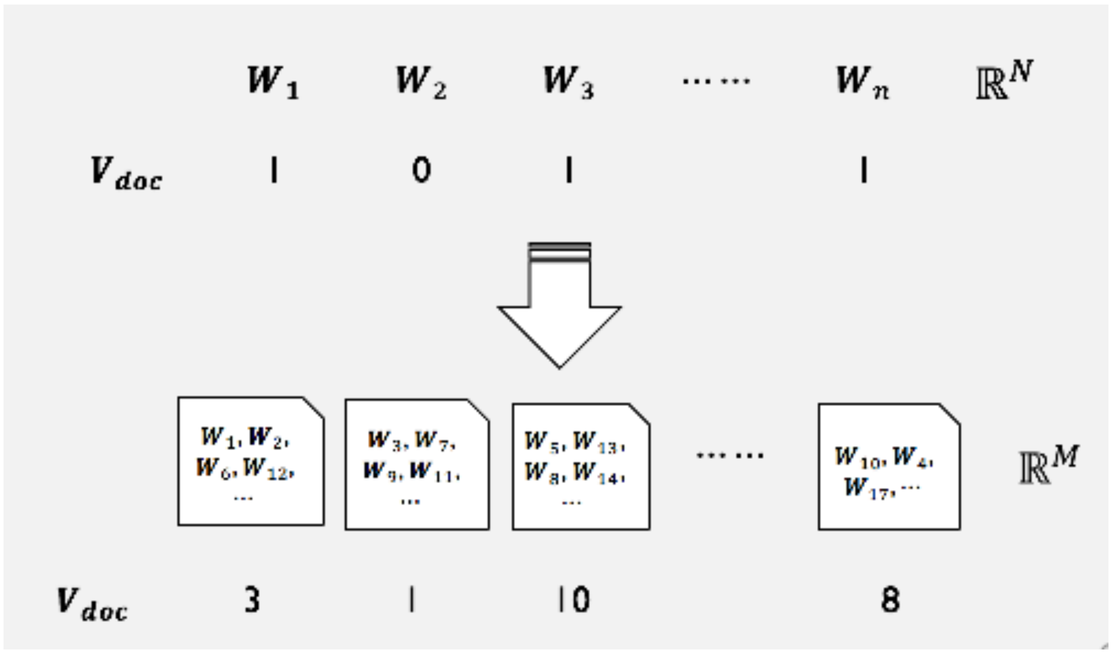
W_1
W_2
W_3
……
W_n
$\mathbb{R}^N$
V_{doc}
1
0
1
1
$W_1, W_2, W_6, W_{12}, \cdots$
$W_3, W_7, W_9, W_{11}, \cdots$
$W_5, W_{13}, W_8, W_{14}, \cdots$
……
$W_{10}, W_4, W_{17}, \cdots$
$\mathbb{R}^M$
V_{doc}
3
1
10
8

Figure 5.3. BoW is Transformed into BoC

5.6 Multi-class Classification

In order to evaluate our model's performance, we use the data for multi-class classification as well. First, we convert the multiclass classification into multiple binary classifications. Here, we apply One-vs-Rest technique [11] to train a single classifier for each class. The instance of one class is viewed as a positive class; the others are considered as negative. We then apply the LinearSVC (Linear Support Vector Classifier) as a classifier based on LIBLINEAR [43].

To evaluate classification performance, we calculated the F1-micro score for each training experiment and recorded the running time. In addition, we implemented the 10-fold cross-validation in the training. Table 5.3 [62] shows the performance of multi-class classification on dimension-reduced datasets. The first row shows the F1-micro score and time consumption without applying our method. The rest rows present the dimension-reduced data classification performance. We can find that the performance results using selected features are slightly better than that using the original features, but the time cost has a relatively big improvement.

Table 5.3 Multi-label Text Classification Performance

Data Dimension	**Classification Performance**	
	F1-Micro Score	***Time Cost (s)***
*11,314 * 61,189*	*0.7524*	*5 * 10^3*
*11,314 * 2000*	*0.774*	*8 *10^2*
*11,314 * 1500*	*0.7619*	*6 * 10^2*
*11,314 * 1000*	*0.753*	*4 * 10^2*
*11,314 * 500*	*0.7506*	*3 * 10^2*

5.7 Conclusion

In our work, we apply the Word2Vec technique to the big data processing. Considering the huge data dimension issue when dealing with large-scale training data, Word2Vec provides a way to cluster the similar words. This strategy can be used to reduce the feature dimension.

On one hand, training data are processed by Word2Vec to generate the word vectors. Through applying the linear calculation on word vectors, we can find semantically related words. On the other hand, we group similar words together using K-means algorithm. By giving values to K, we can construct K clusters. Thus, instead of creating word vectors via vocabulary, we make word vectors based on contents in clusters. This strategy decreases the feature dimension and speeds up the multi-class classification. The new feature dimension reduces the time cost without affecting learning ability from the training data, or even improving classification performance. In the future, we will use the whole dataset and try the different values of K.

6 AI-BASED DEPRESSION DIAGNOSIS SYSTEM

Mental illness is prevalent in the world, and depression is one of the most common psychological problems. Untreated depression increases the chance of dangerous behaviors. Studying the depression report[2], one in four of those who suffered from the depression or receiving any kind of treatments. Such a higher rate attracts more studies on detecting and curing depression. Accurate depression diagnosis is a very complex long-term research problem.

For clinic depression diagnosis, doctors may evaluate a patient via the depression test, such as a physical exam or a lab. Even a communication between doctors and a patient is used to detect his or her depression. On one hand, the current depression diagnosis methods are not accurate due to no gold standard with officially approved symptoms. Thus, it is necessary to discover useful depression features (symptoms) for an accurate depression diagnosis. On the other hand, the significant challenge of detecting depression is the recognition that depressive symptoms may differ from patients' behavior and personalities [76] such as age, sex, environment, and countries. Unfortunately, there is not a general gold standard to help doctors make a decision for a patient's depression. Since it is extremely difficult to have private clinical data from a hospital for our research, we collect public social media data for further depression feature selection.

Public data is being generated every day, countless data are free and huge. Especially, the social media produce big data. Moreover, people discuss all kinds of topics and knowledge on social media. It is beneficial to extract useful information via text mining tools on social media [77], such as Twitter, Facebook, Weibo, and Instagram.

[2] http://cep.lse.ac.uk/pubs/download/special/depressionreport.pdf

To build an AI-based depression diagnosis system, we focus on two major steps. In the first step, we develop an approach to select specific features under the unsupervised learning guidance. Next, we apply selected features from the first step to construct a depression diagnosis system. Therefore, the final step is a text classification or prediction process, which fits our framework [19].

6.1 Data Integration

6.1.1 Data Collection

Social media provides real and massive data; it is an essential public resource for precision medicine research. In our research, we gather data from two public social media platforms: Twitter and Web Blogs to extract new depression symptoms.

- **Tweets (TW)**

Twitter rapidly has become one of the most popular social media since it launched. Its short 140-character messages that are posted by a smartphone or a personal computer are known as "tweets", which can be shared by the entire community. Twitter advises 313 million active users who produce 6,000 tweets on Twitter every second as June 2016[3]. Because of this tremendous volume of data, our research is beneficial for it. We kept monitoring each streaming tweet that includes the word "depression" for almost two weeks in an entire Twitter community. Totally, we roughly gathered 54 millions of tweets that discussed depression. These data are original raw data that transmitted by users, and they are biased and noisy [78].

[3] https://about.twitter.com/company

- **Professional Twitter Accounts (PTA)**

Although we have collected the depression data from the general Twitter users, another extension of Twitter data collection is that we can gather tweets that are posted by professional mental health accounts. The purpose of collecting these specific tweets is that PTAs are knowledgeable and professional in a mental health field. Starting to web scraping the initial webpage[4], thousands of professional mental health tweets were accumulated at the end. We found that many of these tweets discussed the depressive behaviors and their treatments. Therefore, mining tweets given by the professional twitter account is useful.

- **Depression Blog (DB)**

Other than collecting data from the professional Twitter accounts and active Twitter users, another data resource comes from the depression web blogs. Similarly, web scraping begins at the specific webpage[5] to gather relevant data. These blogs and their deep links are almost referred to the depressive symptoms and relative treatments. This set of data excels both in quality and quantity. Investigating the blog data is another appropriate method to discover the hidden insights of the depression symptoms.

6.1.2 Data Preprocessing

Because the data collected from the tweets and web blogs are too biased and noisy, they need to be cleaned in the first step. Considering the complexity of data format, we combine several tools and techniques to make data as clean as possible. Table 6.1 shows the example of noisy data and preprocessing tools we have used.

[4] http://treat-depression.com/top-mental-health-accounts-to-follow-on-twitter
[5] http://www.healthline.com/health/depression/best-blogs-of-the-year

Table 6.1 Data Preprocessing

Noisy Data	**Data Example**	**Techniques**
Specific Characters	@RT, http://	Regular Expression
Punctuations	Comma, colon	Regular Expression
Stop words	is, the	NLTK Toolkit
Non-Words	lmao, hrt	NLTK Toolkit
non-Nouns	eat, wonderful	NLTK Toolkit

Generally, the special characters, such as retweet tag “@RT: xxx”, and link address “http://www.”, contain less information, so they are removed at the beginning. In the next step, stop words and punctuations are filtered by a stop word list that is aggregated by us. Although stop words exhibit high word frequency, they may not contain useful information. Non-words are very common in social media data due to any typos or acronyms, for instance, “hrt”, and “lmao”. These words are removed by checking the English dictionary that is pre-built in the NLTK toolkit [20]. The last step of data preprocessing is to clean the non-Nouns, such as verbs and proposition words. Through employing the Part-of-Speech (PoS) tagging [20] tool on each word, nouns are extracted. At last, we have cleaned data. Table 6.2 shows the number of left words after each step of data preprocessing procedure.

Table 6.2 Word Counts

Steps	TW	PTA	DB
Raw data	54M	18M	46M
Non-words removal	7M	2.6M	8M
Stop words removal	3M	1.2M	3.4M
Nouns	0.72M	0.2M	0.74M

6.2 Entity Extraction

The most difficult task in our research is to extract meaningful entities related to depression from unstructured data because the data itself contains tedious and useless information. Machine learning algorithms and statistical learning methods are used to select the important features according to the model performance or correlation between each entity and target class. Since there are no any labels and classes for the collected depression oriented tweets, we use the unsupervised strategy to select entities. We create a new hybrid algorithm integrating the statistical analysis and the NLP strategy to select major depression symptoms. Our ensemble method can find out the common depression symptoms for the clinical depression detection, and also discover other unfamiliar but useful depression symptoms.

6.2.1 Entity Co-occurrence

The co-occurrence distribution displays the importance of a term in text data. If the probability distribution of co-occurrence between a specific term and the frequent terms is biased to a subset of the frequent terms, then this specific term is likely to be a keyword. Because we have collected all tweets that contain the term "depression", counting the co-occurrence frequency of each entity that associates with the term "depression" is used to find the distribution of them. This approach is able to discover which entities are common with the term "depression".

To discover the most frequent entities that co-occur with the term "depression", a co-occurrence matrix is created. In Figure 6.1, the most frequent entities associated with the term "depression" are shown. For instance, one of the most common symptoms of depression is "anxiety", it is prevalent in the collected tweets. Depression symptoms are selected by calculating the entities' counts associated with the term "depression".

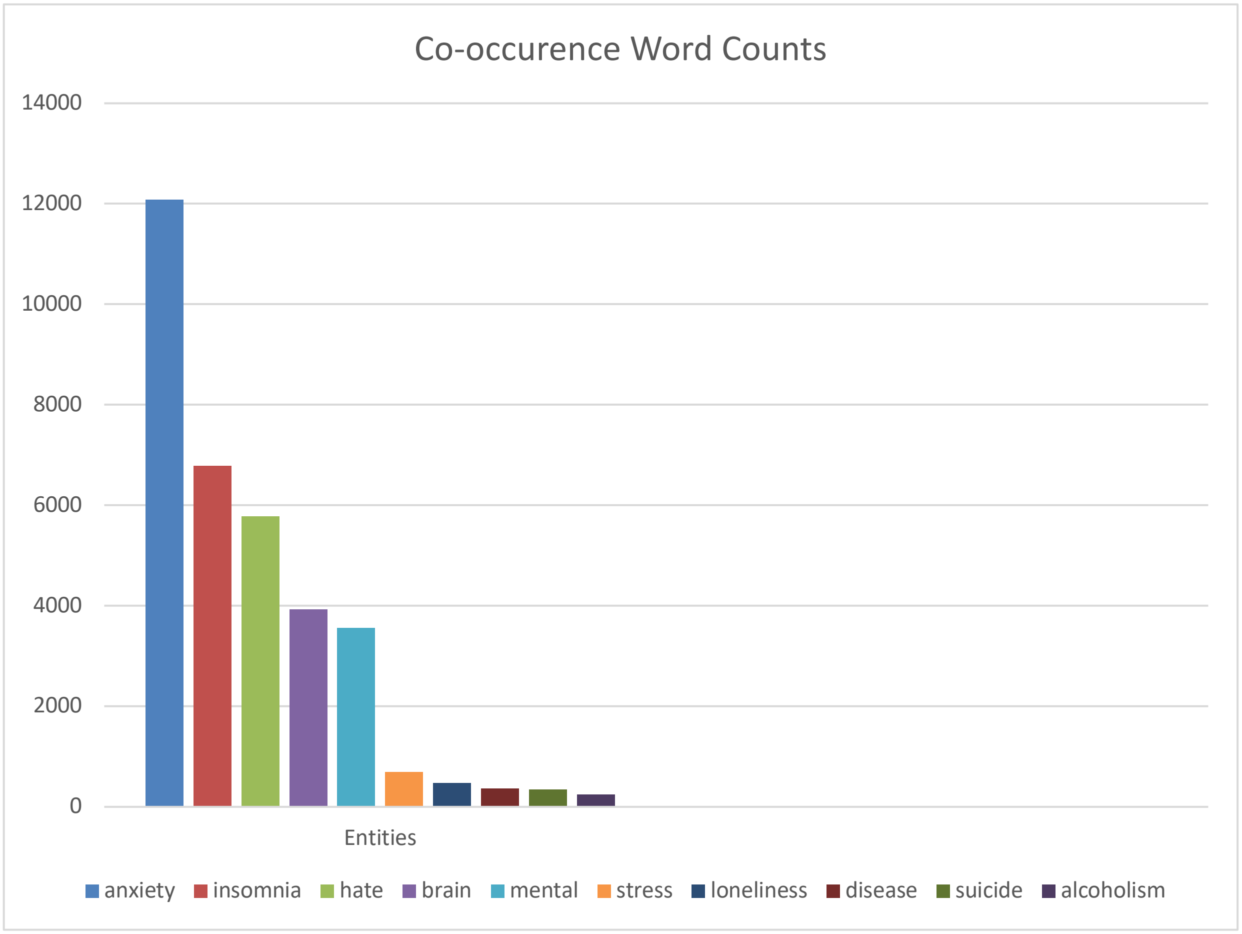

Figure 6.1 Entity Co-occurrence counts that associate with the term "depression"

6.2.2 Entity Similarity

To analyze the relations between words in the documents, words are initially transformed into vectors. Currently, there are two common strategies to generate word vectors. One is one-hot encoding, and the other one is the word embedding (e.g. Word2Vec). The essential idea of one-hot encoding [79] is to associate each word in the vocabulary with vector representation. Words are represented in a high and sparse dimension; each word corresponds to a point in the vector space. The produced word vectors have a higher feature dimension (vocabulary size *N*), so it is difficult to capture the "relationship" between words.

Mikolov *et al*. [46, 47] extended Bengio's NNLM [69] and developed Word2Vec to generate the word representations via different learning methods. The Word2Vec contains two distinct learning models: Continues Bag of Words (CBOW) and Skip-gram [46,47]. Figure 5.2 shows the structures of them. The CBOW model predicts the current word when the neighboring words are given in the surrounding window. In the opposite of the CBOW model, the Skip-gram model learns the context words by giving a word in the input layer and predicts its surrounding words in the output layer.

The reason for applying the word similarity as another factor when choosing the key entities is that the Word2Vec performs well on generating the good quality of word representations. Words can be represented by fixed length vectors. Calculating the cosine similarity between words' vectors can be used to find the similar words of a target term. For example, Figure 6.2 shows the similar words and corresponding cosine similarities given a target word "depression". This method may discover the key depression related entities from the natural language knowledge perspective.

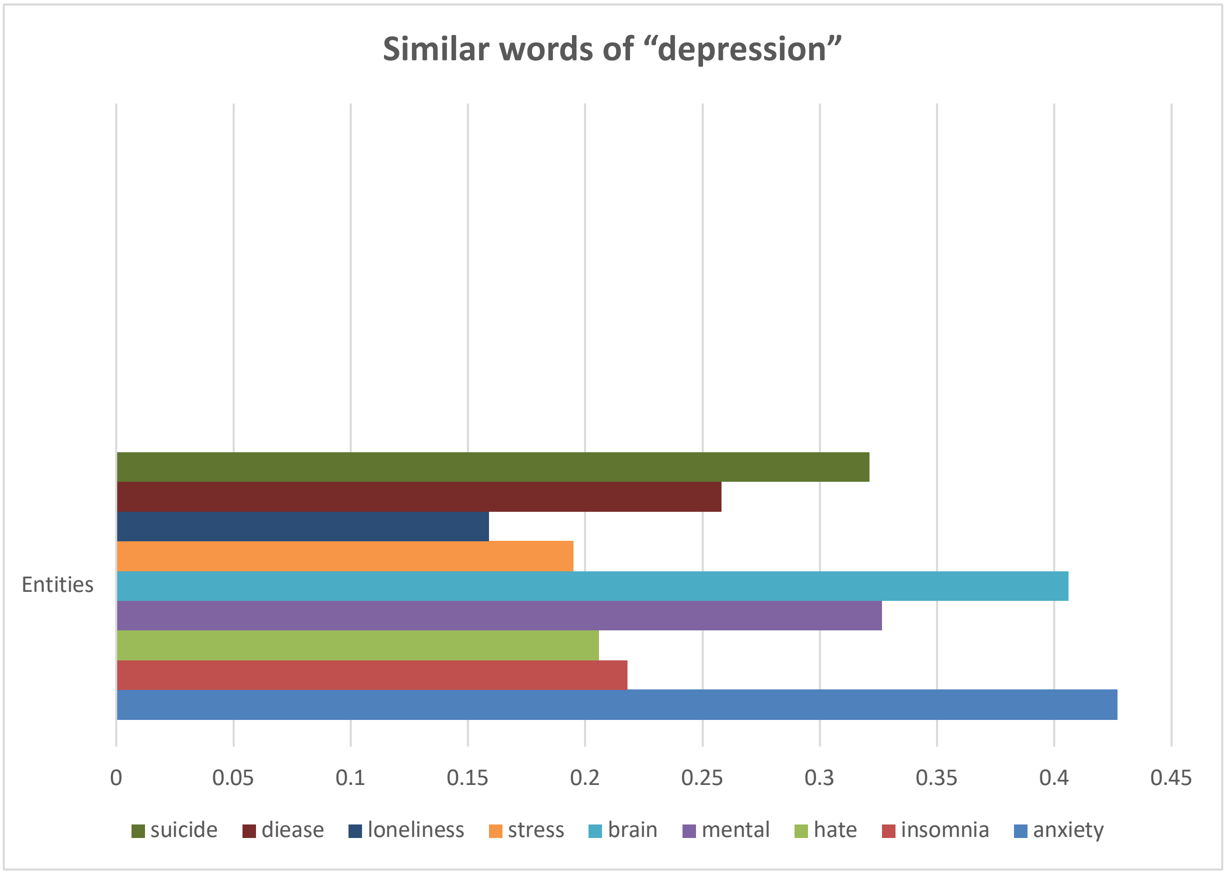

Figure 6.2 Similar words related to "depression"

Another experiment on entity similarity in Figure 6.3 illustrates the hierarchical structure and relationships among the depression facts learned from the whole data. In this tree, we extract the four most frequent depression symptoms from whole data and they are shown in the second level. In the third layer, we show depression facts that are related to one of four common depression symptoms individually. These depression facts are calculated and accumulated by Word2Vec given one depression symptom. We believe that extracted depression facts are good references when recognizing the depressive behaviors.

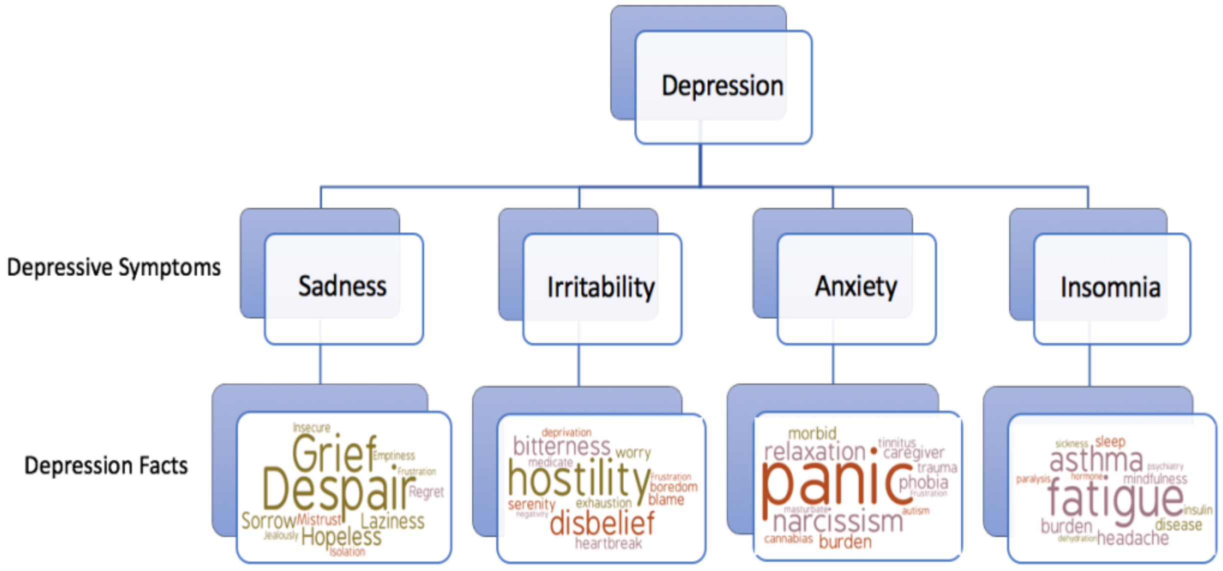

Figure 6.3 Hierarchical Structure of Depression and its Symptoms

6.2.3 Rapid Automatic Keyword Extraction Algorithm

The **R**apid **A**utomatic **K**eyword **E**xtraction (RAKE) algorithm is described [78]. Candidates are selected from the text by finding all possible strings that do not include stop words or phrase delimiters. The RAKE algorithm generates a list of candidate keywords and phrases. For each candidate, The RAKE algorithm calculates properties to identify the candidate's importance. Firstly, compute the frequency of each word. Then, find the degree of every word. The degree of a word is the number of how many times a word is used by other candidate keywords. An individual word's score is calculated by a formula:

$$\frac{T + D}{T} \qquad (\mathbf{3})$$

where T is the word frequency, and D is the degree of a word. A score is calculated for each phrase that is the summation of the single word's score. Therefore, the RAKE score can be used in our new approach as a factor to extract the key entities.

Both long and short phrases reveal meaningful information for the information retrieval in NLP task, the RAKE algorithm calculates the scores for individual words. Table 6.3 shows the entity and its importance score calculated by the RAKE algorithm. Because the scores shown in Table 6.3 are close to each other, the AKEW algorithm assigns a relatively smaller weight to a RAKE score than other factors.

Table 6.3 RAKE scores for single entity

Entities	**RAKE Scores**
anxiety	*1.0121*
insomnia	*1.0018*
hate	*1.0*
hunger	*1.0*
brain	*1.0009*
stress	*1.0011*
loneliness	*1.0004*
alcoholism	*1.0*

6.3 Automatic Extract Keyword for Specific Terms

We propose an ensemble method to extract depression symptoms from social media. The relationships between a key entity and its related entities are shown in a semantic graph. To extract important entities for the term "depression", we develop an Automatic Extract Keyword for specific terms (**AEKW**) algorithm that combines the co-occurrence count between the term "depression" and other entities, Word2Vec, and RAKE. Thus, the AEKW integrates the statistical analysis and word embedding technique to select the key entity related to the term "depression". The AEKW algorithm is given below.

```
AEKW Algorithm

Input: term
Output: Semantic Graph G (V, E)
$X_i =$ entity in Tweets
$T_i =$ the co-occurrence counts between entity and the term
$R_i =$ the RAKE score for entity
$W_i =$ the similarity between term and entities
$S_i$ = the score for entities that relevant to term
  G (V, E): V = entities, E = importance scores
begin
    For entity in Tweets:
          Calculate the T for $X_i$: $\{(X_1,T_1)\}, \{(X_2,T_2)\},\ldots,\{(X_n,T_n)\}$
          Calculate the R for $X_i$: $\{(X_1,R_1)\}, \{(X_2,R_2)\},\ldots,\{(X_n,R_n)\}$
          Calculate the distance between term and $X_i$, $\{(X_1,W_1), \{(X_2,W_2)\},\ldots,\{(X_k,W_k)\}$
      $S_i = \frac{1}{2}Normalize(T_i) + \frac{1}{3}R_i + \frac{1}{6}W_i$
          Calculate the S for X: $\{(X_1,S_1)\}, \{(X_2,S_2)\},\ldots,\{(X_n,S_n)\}$
          Sort the entity based on the value $S_i$ , top k entities are extracted
          K number of entities left: $\{(X_1,S_1)\}$ , $\{(X_2,S_2)\},\ldots,\{(X_k,S_k)\}$, $k < n$
```

```
            Build a graph G:
            $V$ = (term, $X_1, X_2, \dots, X_k$)
            $E$ = $\{(X_1, term), S_1\}$, $\{(X_2, term), S_2\}, \dots, \{(X_k, term), S_k\}$
end.
```

The AKEW algorithm counts the co-occurrence of entities paired with the term "depression" and normalizes the co-occurrence counts into the range between 0 and 1. Next, it utilizes the Word2Vec to compute the cosine similarity between "depression" and each entity. The RAKE algorithm is used to calculate an individual entity and assign a score to it. Now, every entity has three scores that are calculated by the above methods. Based on the total score of the three scores for each entity, top *K* entities with the *K* highest scores are selected. Then, the AEKW algorithm can build a relevant semantic graph. In this semantic graph, the new depressive symptoms may be discovered. Moreover, the important entities related to depression will be used as features in the depression diagnosis and depression self-screening tasks.

6.4 Depression Semantic Graph

The AEKW algorithm calculates the importance score for each entity, then builds a semantic graph to display the relationships between a specific term and its correlated entities. This semantic graph uses the entity "depression" as the central word, the vertices are other relevant entities, and edges represent the importance scores between vertices. In the semantic graph, we may find important entities for depression specifically. Moreover, after expanding the semantic graph, we may discover more unfamiliar and useful depression symptoms. Figure 6.4 displays a partial semantic graph for the central term "depression".

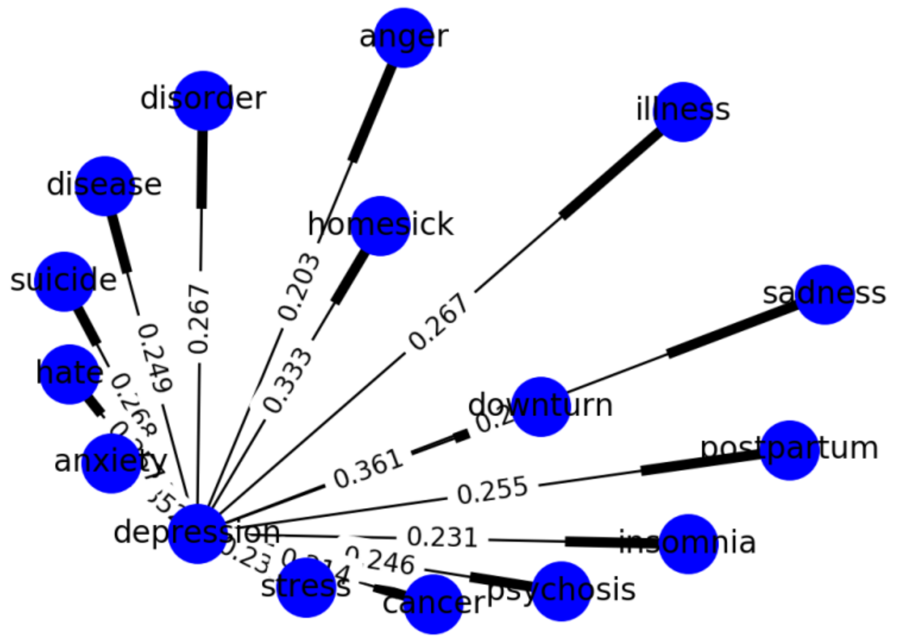

Figure 6.4 Semantic graph with the center vertex "depression" and the weights represented as the importance score between two vertices

6.5 Conclusion

Clinic depression is a serious mental illness which negatively affects human's health. Unfortunately, in contrast to other mental disorder, the depression is persistent. Lots of researches contribute to depression detecting and treatment in diverse scientific fields. However, it is still a challenge to confirm human's depression symptoms from their behaviors via clinic records and depression tests. As social media is growing rapidly, it provides a way of sharing any kinds of feelings and knowledge in the community.

Because of clinical data privacy in hospitals, our new method extracts useful depression symptoms from public tweets generated by depression patients, medical doctors, and other users. Considering Twitter and Web Blog are valuable data resources; we gathered data large quantity of data from them. Although there are some studies on detecting the depression symptoms via the social networks or biomedical literature, our work uses the advanced strategies such as web scraping and word embedding, to process and analyze the social media data. To extract more useful depression symptoms, a new algorithm AEKW is designed by integrating the statistical analysis and the NLP technology. Depression symptoms extracted from the AEKW algorithm are used to construct a depression semantic graph that will be used for the further depression diagnosis.

7 CONCLUSION

We propose and develop a multi-label text classification framework with feature selection strategies for text mining and text annotation tasks. Because of a variety of big data, we investigate different approaches and strategies to analyze data and build a classifier. The main contribution of the proposed framework is to extract features and reduce the feature dimension for data mining and text annotation.

For the good quality of labeled data, we designed bagging schemes in the MIMLfast algorithm [10] to perform the multi-label classification. Besides, we applied the feature reduction method such as PCA [21] and the count-based feature selection method to work with the MIMLfast algorithm, but the overall performance was not improved significantly as our expected. Thus, we developed a new hybrid feature selection method. Based on current feature selection techniques, we build the sequential and ensemble approaches to reduce the feature dimension. The experiment results showed that our methods could increase the accuracy in all label dimensions. To evaluate our method on another type of dataset, we collected the artificial data (GeneBase dataset [58]) to test our feature selection methods, the multi-label text classification performance was improved dramatically.

The above ensemble feature selection methods performed well in the small dataset, even with the relatively large feature dimension. Then we proposed another feature reduction algorithm to process the big textual data by using the neural network language model called Word2Vec. Through transforming the document representation from Bag of Words (BoW) into Bag of Concepts (BoC), the classification performance was improved and the computing cost was decreased as well. In addition, our method could generate the good representation of documents, thus these document vectors can be used in document summarization and document clustering tasks [18].

In the first two projects, feature reduction and feature selection could help the multi-label text classification on the labeled data. Moreover, we built a depression diagnosis system from the unsupervised perspective. Because the clinical data is extremely expensive and inaccessible, we collect data from public social media. The social media provide a public community where users can share their feelings and knowledge for public users, such as Facebook, Twitter, Weibo and Web Blogs. The biggest challenge so far is to discard the noisy data and discover the really important words that related to the depression. Therefore, we built the AEKW algorithm to define the importance of an entity and build a semantic graph to display the most important words related to depression. In this method, many of extracted depression symptoms were common symptoms that were already known; while, our approach could find many useful but unfamiliar depression symptoms which were very often being discussed in the social network. These depression symptoms are good references for a doctor to detect and confirm patients' depression.

Our present research goal is to build an AI-based depression diagnosis system. In the next, we will use the semantic graph to generate major depression symptoms from Twitter, and then apply the association rule mining [23] to find mutual relations among top symptoms without labeled data. These major strong association rules can be used for the early depression diagnosis.

In the future, we will design a grammar-based approach to identifying the depressive users on Twitter. Through combining the depression symptoms and depression users' tweets, a classifier will be built for an accurate depression diagnosis. An intelligent depression diagnosis system for medical doctors and a convenient depression self-screening software system for ordinary people will be developed. In addition, our framework is able to select different kinds of entities for the sentiment analysis [64, 67].

REFERENCES

[1] M. D. Turner, C. Chakrabarti, T. B. Jones, J. F. Xu, P. T. Fox, G. F. Luger, A. R. Laird, and J. A. Turner, “Automated annotation of functional imaging experiments via multi-label classification,” Frontiers in neuroscience, vol. 7, 2013.

[2] C. C. Aggarwal, and C. Zhai, “Mining text data,” Springer Science & Business Media, 2012.

[3] R. Feldman, and J. Sanger, “The text mining handbook: advanced approaches in analyzing unstructured data,” Cambridge University Press, 2007.

[4] S. M. Weiss, N. Indurkhya, and T. Zhang, “Fundamentals of predictive text mining,” Springer Science & Business Media, 2010.

[5] E. Candes, and T. Tao, “The Dantzig selector: Statistical estimation when p is much larger than n,” Annals of Statistics, vol. 35, no. 6, pp. 2313-2351, Dec, 2007.

[6] C.-T. Nguyen, D.-C. Zhan, and Z.-H. Zhou, “Multi-Modal Image Annotation with Multi-Instance Multi-Label LDA,” in Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI'13), Beijing, China, 2013.

[7] S.-J. Huang, and Z.-H. Zhou, “Fast Multi-Instance Multi-Label Learning,” arXiv: 1310.2049, 2013.

[8] S.-J. Huang, and Z.-H. Zhou, “Active query driven by uncertainty and diversity for incremental multi-label learning,” in 2013 IEEE 13th International Conference on Data Mining (ICDM), 2013, pp. 1079-1084.

[9] Z. H. Zhou, and M. L. Zhang, “Solving multi-instance problems with classifier ensemble based on constructive clustering,” Knowledge and Information Systems, vol. 11, no. 2, pp. 155-170, Feb, 2007.

[10] Z.-H. Zhou, M.-L. Zhang, S.-J. Huang, and Y.-F. Li, “Multi-instance multi-label learning,” Artificial Intelligence, vol. 176, no. 1, pp. 2291-2320, 2012.

[11] R. Caruana, N. Karampatziakis, and A. Yessenalina, “An empirical evaluation of supervised learning in high dimensions,” in Proceedings of the 25th international conference on Machine learning, 2008, pp. 96-103.

[12] J. Fan, and Y. Fan, “High dimensional classification using features annealed independence rules,” Annals of statistics, vol. 36, no. 6, pp. 2605, 2008.

[13] Z. Wang, L. Ma, Y.-Q Zhang, “A Hybrid Machine Learning Method for Finding Depression Related Publications by Eliminating Outlier Publications,” IEEE Society for Information Reuse and Integration, San Diego, USA, August 2017

[14] J. Fürnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker, “Multilabel classification via calibrated label ranking,” Machine learning, vol. 73, no. 2, pp. 133-153, 2008.

[15] S. Godbole, and S. Sarawagi, "Discriminative methods for multi-labeled classification," Advances in Knowledge Discovery and Data Mining, pp. 22-30: Springer, 2004.

[16] G. Tsoumakas, and I. Katakis, “Multi-label classification: An overview,” Dept. of Informatics, Aristotle University of Thessaloniki, Greece, 2006.

[17] Z. Wang, L. Ma, and Y.-Q Zhang, "A Hybrid Document Feature Extraction Method Using Latent Dirichlet Allocation and Word2Vec," 2016 1st IEEE International Conference on Data Science in Cyberspace, Changsha, Chine, June 13-16, 2016

[18] Zamir, Oren, and Oren Etzioni. "Web document clustering: A feasibility demonstration," Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. ACM, 1998.

[19] L. Ma, Z. Wang, and Y-Q Zhang, "Extracting Depression Symptoms from Social Networks and Web Blogs via Text Mining", 13th International Symposium on Bioinformatics Research and Applications (ISBRA), Honolulu, Hawaii, May, 2017

[20] "Natural Language Tool Kit (NLTK) 3.0 documentation," November, 2014; www.nltk.org/api/nltk.stem.html.

[21] S. Wold, K. Esbensen, and P. Geladi, "Principal component analysis," Chemometrics and intelligent laboratory systems, vol. 2, no. 1, pp. 37-52, 1987.

[22] K. Sparck Jones, "A statistical interpretation of term specificity and its application in retrieval," Journal of documentation, vol. 28, no. 1, pp. 11-21, 1972.

[23] Ma, Bing Liu Wynne Hsu Yiming, and Bing Liu, "Integrating classification and association rule mining," Proceedings of the 4th. 1998.

[24] A. Özgür, L. Özgür, and T. Güngör, "Text categorization with class-based and corpus-based keyword selection," Computer and Information Sciences-ISCIS 2005, pp. 606-615: Springer, 2005.

[25] J. A. Turner, and A. R. Laird, "The cognitive paradigm ontology: design and application," Neuroinformatics, vol. 10, no. 1, pp. 57-66, 2012.

[26] G. Tsoumakas, I. Katakis, and I. Vlahavas. "Mining multi-label data," In O. Maimon and L. Rokach, editors, Data Mining and Knowledge Discovery Handbook. 2nd edition, Springer, 2010.

[27] D. Ren, L. Ma, Y. Zhang, R. Sunderraman, P. Fox, A. Laird, J. Turner, and M. Turner, "Online Biomedical Publication Classification Using Multi-Instance Multi-Label Algorithms with Feature Reduction,", 14th IEEE International Conference on Cognitive Informatics and Cognitive Computing (ICCI*CC 2015), 2015.

[28] T. Simon, and D. Koller, "Support vector machine active learning with applications to text classification," The Journal of Machine Learning Research 2, pp.45-66, 2002.

[29] S. Johan, and J. Vandewalle, "Least squares support vector machine classifiers," Neural processing letters 9.3 pp. 293-300, 1999.

[30] B. Justin, C. Sanderson, and A. Kowalczyk, "An efficient alternative to svm based recursive feature elimination with applications in natural language processing and bioinformatics," AI 2006: Advances in Artificial Intelligence, Springer Berlin Heidelberg, pp. 170-180, 2006.

[31] A. Alexandre, F. Pedregosa, M. Eickenberg, P. Gervais, A. Mueller, J. Kossaifi, A. Gramfort, B. Thirion, and G. Varoquaux, "Machine learning for neuroimaging with scikit-learn," Frontiers in neuroinformatics 8, 2014.

[32] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[33] K. A. Markus, “Principles and Practice of Structural Equation Modeling by Rex B. Kline,” Structural Equation Modeling: A Multidisciplinary Journal 19.3, pp. 509-512, 2012.

[34] L. Andy and M. Wiener, “Classification and regression by Random Forest,” R news 2.3, pp. 18-22, 2002.

[35] G.Robin, J. Poggi, and C. Tuleau-Malot, “Variable selection using random forests,” Pattern Recognition Letters 31.14 p. 2225-2236, 2010

[36] B. Leo, “Random forests–random features,” UC Berkeley, Statistics Department, Technical Report 567, 1999.

[37] Y. Li, D.F. Hsu, and S.M. Chung, “Combining Multiple Feature Selection Methods for Text Categorization by Using Rank-Score Characteristics,” Tools with Artificial Intelligence, ICTAI '09. 21st International Conference, vol. 508, no. 517, pp. 2-4, 2009.

[38] R. Neumayer, R. Mayer, and K. Nørvåg, “Combination of Feature Selection Methods for Text Categorisation,” Advances in Information Retrieval Lecture Notes in Computer Science, vol. 6611, pp. 763-766, 2011

[39] S. Godbole and S. Sarawagi, “Discriminative methods for multi-labeled classification,” PAKDD ’04: 8th Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 22–30, 2004.

[40] J. Fu ̈rnkranz, E. Hüllermeier, E. L. Mencía, and K. Brinker. “Multilabel classification via calibrated label ranking,” Machine Learning, vol. 73(2), pp. 133– 153, 2008.

[41] J. Read, B. Pfahringer, G. Holmes and E. Frank, “Classifier Chains for Multi-label Classification,” Machine Learning Journal, Springer. Vol. 85(3), 2011.

[42] E. Montañes, R. Senge, J. Barranquero, J. R. Quevedo, J. J. del Coz, and E Hüllermeier, “Dependent binary relevance models for multi-label classification,” Pattern Recognition, 47(3), pp.1494-1508, 2014.

[43] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A library for large linear classification,” Journal of Machine Learning Research 9, pp. 1871-1874, 2008.

[44] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” Machine Learning, vol. 46, pp. 389–422, 2002.

[45] M. Viktor and K. Cukier, “Big data: A revolution that will transform how we live, work, and think,” Houghton Mifflin Harcourt, 2013.

[46] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient Estimation of Word Representations in Vector Space,” Proc. Workshop at ICLR, 2013.

[47] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Distributed Representations of Words and Phrases and their Compositionality,” Proc. NIPS, 2013.

[48] T. Mikolov, W.Yih, and G. Zweig, “Linguistic Regularities in Continuous Space Word Representations,” Proc. NAACL HLT, 2013.

[49] J. A Hartigan and M. A. Wong, "Algorithm AS 136: A k-means clustering algorithm," Applied statistics, 1979, pp.100-108.

[50] K. Lang, "20 newsgroup data set", Available at: qwone.com/~jason/20Newsgroups/. [Accessed 30-Sep-2015]

[51] R. Řehůřek and P. Sojka, Proceedings of the LREC 2010 Workshop on New Challenges for NLP Framework. ELRA, Valletta, Malta, 2010, pp.45-50.

[52] D. Ravichandran, P. Pantel, and E. Hovy, "Randomized algorithms and nlp: using locality sensitive hash function for high speed noun clustering," Proc. The 43rd Annual Meeting on Association for Computational Linguistics, 2005, pp.622-629.

[53] G. Allen and R. M. Gray, "Vector quantization and signal compression," Springer Science & Business Media, vol. 159, 2012.

[54] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler, "The Hadoop Distributed File System," In Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST) (MSST '10). IEEE Computer Society, Washington, DC, USA, 1-10

[55] Matei Zaharia, Mosharaf Chowdhury, Michael J. Franklin, Scott Shenker, and Ion Stoica, "Spark: cluster computing with working sets," In Proceedings of the 2nd USENIX conference on Hot topics in cloud computing (HotCloud'10). USENIX Association, Berkeley, CA, USA, 10-10.

[56] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in neural information processing systems. 2012.

[57] Lipton, Zachary C., "A critical review of recurrent neural networks for sequence learning," arXiv preprint arXiv:1506.00019 (2015).

[58] S. Diplaris, G. Tsoumakas, P. Mitkas and I. Vlahavas, "Protein Classification with Multiple Algorithms," Proc. 10th Panhellenic Conference on Informatics (PCI 2005), pp. 448-456, Volos, Greece, November 2005.

[59] Cline, Alan Kaylor, and Inderjit S. Dhillon, "Computation of the singular value decomposition," Handbook of linear algebra (2006): 45-1.

[60] Yeo, Chee Shin, et al., "Cluster computing: high-performance, high-availability, and high-throughput processing on a network of computers," Handbook of nature-inspired and innovative computing. Springer US, 2006. 521-551.

[61] L. Ma, Y. Zhang, R. Sunderraman, P. Fox, A. Laird, J. Turner, and M. Turner, "Hybrid Feature Selection Methods for Online Biomedical Publication Classification," 2015 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology, Canada, 2015.

[62] Long Ma and Yanqing Zhang, "Using Word2Vec to process big text data," 2015 IEEE International Conference on Big Data, San Jose, USA, 2015.

[63] Najafabadi, Maryam M., et al., "Deep learning applications and challenges in big data analytics," Journal of Big Data 2.1 (2015): 1-21.

[64] Asghar, M. Zubair, et al., “A Review of Feature Extraction in Sentiment Analysis,” Journal of Basic and Applied Scientific Research 4.3 (2014): 181-186.

[65] Steinbach, Michael, George Karypis, and Vipin Kumar, “A comparison of document clustering techniques,” KDD workshop on text mining. Vol. 400. No. 1. 2000.

[66] Brown, Peter F., et al., “Class-based n-gram models of natural language,” Computational linguistics 18.4 pp. 467-479, 1992.

[67] Maas, Andrew L., et al., “Learning word vectors for sentiment analysis,” Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1. Association for Computational Linguistics, 2011.

[68] Menczer, Filippo, Melania Degeratu, and W. Nick Street, “Efficient and scalable pareto optimization by evolutionary local selection algorithms,” Evolutionary computation 8.2 pp. 223-247, 2000.

[69] Bengio, Y., Ducharme, R., Vincent, and P., Jauvin, C., “A neural probabilistic language model,” Journal of machine learning research, vol. 3 (Feb), pp. 113--1155, 2003

[70] Wang, S. H., Ding, Y., Zhao, W., Huang, Y. H., Perkins, R., Zou, W., and Chen, J. J, “Text mining for identifying topics in the literatures about adolescent substance use and depression,” BMC public health, vol. 16(1), pp. 279. (2016)

[71] Blei, D. M., Ng, A. Y., Jordan, and M. I, “Latent dirichlet allocation,” Journal of machine Learning research, vol. 3(Jan), pp. 93--1022. (2003)

[72] Nadeem, Moin, “Identifying depression on Twitter,” arXiv preprint arXiv:1607.07384, 2016.

[73] Tsugawa, Sho, Yusuke Kikuchi, Fumio Kishino, Kosuke Nakajima, Yuichi Itoh, and Hiroyuki Ohsaki, “Recognizing depression from twitter activity,” In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, pp. 3187-3196. ACM, 2015.

[74] De Choudhury, Munmun, Michael Gamon, Scott Counts, and Eric Horvitz, “Predicting Depression via Social Media,” In ICWSM, pp. 2, 2013.

[75] Witten, I. H., Frank, E., Hall, M. A., Pal, and C. J., “Data Mining: Practical machine learning tools and techniques,” Morgan Kaufmann. (2016)

[76] Griffin, J. M., Fuhrer, R., Stansfeld, S. A., and Marmot, M., “The importance of low control at work and home on depression and anxiety: do these effects vary by gender and social class?,” Social science and medicine, vol. 54(5), pp. 783--798. (2002)

[77] He, W., Zha, S., and Li, L., “Social media competitive analysis and text mining: A case study in the pizza industry,” International Journal of Information Management, vol. 33(3), pp. 464--472. (2013)

[78] Berry, Michael W. and Jacob Kogan, eds, “Text mining: applications and theory,” John Wiley and Sons, 2010.

[79] Landauer, T. K., “Latent semantic analysis,” In: John Wiley Sons, Ltd, 2006

[80] Larose, Daniel T., “k-Nearest Neighbor Algorithm,” Discovering Knowledge in Data: An Introduction to Data Mining (2005): 90-106.

[81] Schalkoff, Robert J., “Artificial neural networks,” Vol. 1. New York: McGraw-Hill, 1997.

[82] Dunne, Rob A., and Norm A. Campbell, “On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function,” Proc. 8th Aust. Conf. on the Neural Networks, Melbourne, 181. Vol. 185. 1997.

[83] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” Proceedings of COMPSTAT' 2010, pp. 177-186: Springer, 2010.

[84] Zhipeng Cai, Randy Goebel, Mohammad Salavatipour, and Guohui Lin, “Selecting dissimilar genes for multi-class classification, an application in cancer subtyping,” BMC Bioinformatics, 8(2007), 206, 2007.

[85] Hadi Sabaa, Zhipeng Cai, Yining Wang, Randy Goebel, Steve Moore, and Guohui Lin, “Whole Genome Identity-by-descent Determination,” Journal of Bioinformatics and Computational Biology. 11(2), 1350002, 2013.

[86] Zhipeng Cai, Lizhe Xu, Yi Shi, Mohammad Salavatipour, Randy Goebel, and Guohui Lin, “Using Gene Clustering to Identify Discriminatory Genes with Higher Classification Accuracy,” The IEEE 6th Symposium on Bioinformatics and Bioengineering (BIBE 2006), 2016.