

https://dspace.org/

Investigating unsupervised feature learning for email spam classification

Diale, Melvin

2017

https://hdl.handle.net/10539/24027

Diale, Melvin (2017) Investigating unsupervised feature learning for email spam classification, University of the Witwatersrand, Johannesburg, <http://hdl.handle.net/10539/24027>

Downloaded from a DSpace repository. https://dspace.org/

Investigating Unsupervised Feature Learning for Email Spam Classification



UNIVERSITY OF THE
WITWATERSRAND,
JOHANNESBURG

A dissertation submitted in partial fulfillment of the requirements for the degree Master of Science.

Author: Diale Melvin

Principal Supervisor: Prof. Turgay Celik
Co-Supervisor: Dr. Christiaan Van Der Walt

School of Computer Science and Applied Mathematics,
Faculty of Science,
University of the Witwatersrand, Johannesburg.

October 2017

Declaration

I, Melvin Diale, declare that this research project is my own work, and all information sources have been referenced. Additionally, this work has never been submitted to any other university for any other degree.

Signed: _____

Date: _____

Abstract

In the cyberspace, spam emails are used as a way to divulge sensitive information of victims through social engineering. There are various classification systems that have been employed previously to identify spam emails. The primary objective of email spam classification systems is to classify incoming email as either legitimate (non-spam) or spam emails. The spam classification task can thus be regarded as a two-class classification problem. This kind of a problem involves the use of various classifiers such as Decision Trees (DTs) and Support Vector Machines (SVMs). DTs and SVMs have been shown to perform well on email spam classification tasks. Several studies have failed to mention how these classifiers were optimized in terms of their hyperparameters. As a result, poor performance was encountered with complex datasets. This is because SVM classifier is dependent on the selection of the kernel function and the optimization of kernel hyperparameters. Additionally, many studies on spam email filtering task use words and characters to compute Term-Frequency (TF) based feature space. However, TF based feature space leads to sparse representation due to the continuous vocabulary growth. This problem is linked with the curse of dimensionality. Overcoming dimensionality issues involves the use of feature reduction techniques. Traditional feature reduction techniques, for instance, Information Gain (IG) may cause feature representations to lose important features for identifying spam emails. This proposed study demonstrates the use of Distributed Memory (DM), Distributed Bag of Words (DBOW), Cosine Similarity (CS) and Autoencoder for feature representation to retain a better class separability. Generated features enable classifiers to identify spam emails in a lower dimension feature space. The use of the Autoencoder for feature reduction led to improved classification performance. Furthermore, a comparison of kernel functions and CS measure is taken into consideration to evaluate their impacts on classifiers when employed for feature transformation. The study further shows that removal of more frequent words, which have been regarded as noisy words and stemming process, may negatively affect the performance of the classifiers when word order is taken into consideration. In addition, this study investigates the performance of DTs and SVM classifiers on the publicly available datasets. This study makes a further investigation on the selection of optimal kernel function and optimization of kernel hyperparameters for each feature representation. It is further investigated whether the use of Stacked Autoencoder as a pre-processing step for multilayer perceptron (MLP) will lead to improved classification results.

Contents

Declaration i

Abstract ii

Contents iii

List of Figures v

List of Tables vii

Dedication ix

Acknowledgements x

Publications xi

1 Introduction 1
- 1.1 Background . . . 1
- 1.2 Problem Definition . . . 5
- 1.3 Research Questions . . . 6
- 1.4 Research Objectives . . . 6
- 1.5 Delineations and Limitations . . . 7
- 1.6 Research significance . . . 8
- 1.7 Overview . . . 8

2 Literature Review 10
- 2.1 Callback Verification . . . 10
- 2.2 Background Checks . . . 11
- 2.3 Honeypots . . . 11
- 2.4 Vector Space Models . . . 12
  - 2.4.1 Feature Extraction . . . 12
    - 2.4.1.1 Bag of Words . . . 12
    - 2.4.1.2 Term Frequency - Inverse Document Frequency . . . 14
    - 2.4.1.3 N-gram . . . 15
  - 2.4.2 Feature Reduction . . . 16
    - 2.4.2.1 Information Gain . . . 16
    - 2.4.2.2 Chi Square . . . 17

2.4.2.3 Singular Value Decomposition . . . 18
2.4.2.4 Latent Semantic Analysis . . . 21
2.4.2.5 ReliefF . . . 23
2.5 Binary Classification . . . 24
2.6 Autoencoder . . . 36
2.7 Stacked Autoecoder . . . 37
2.8 Summary . . . 37

3 Word Count Based Feature Construction 39
3.1 Introduction . . . 39
3.2 Methodology . . . 41
3.3 Experimental Setup . . . 42
3.4 Results and Analysis . . . 44
3.5 Conclusion . . . 48

4 Neural Network Based Feature Construction 49
4.1 Introduction . . . 49
4.2 Distributed Memory and Distributed Bag of Words . . . 50
4.3 Cosine Similarity . . . 51
4.4 Experimental setup of Neural Network based approach for feature reduction 51
4.5 Experimental setup of supervised methods for feature reduction . . . 54
4.6 Experimental setup of SVD based method for feature reduction . . . 55
4.7 Data . . . 56
4.8 Performance Metrics . . . 57
4.9 Performance of classifiers on features extracted using Neural Network based methods . . . 57
4.10 Performance of classifiers on features extracted using SVD related methods 68
4.11 Summary . . . 78
4.12 Conclusions . . . 78

5 Neural Network for Email Spam Detection 80
5.1 Introduction . . . 80
5.2 Experiment . . . 80
5.3 Performance Metrics . . . 81
5.4 Results Analysis . . . 82
5.5 Conclusions . . . 85

6 Conclusion 87

Bibliography 90

List of Figures

2.1 An example of the Autoencoder with a single hidden layer. 36
2.2 An example of the Stacked Autoencoder with multiple hidden layers. . . . 37

3.1 Feature Selection and Hyperparameter Optimization 43
3.2 The performance of the classifiers on the feature representation generated using IG and BoW with different feature sizes on Enron dataset. 46
3.3 The performance of the classifiers on the feature representation generated using IG and TF-IDF with different feature sizes on Enron dataset. 46
3.4 The performance of the classifiers on the feature representation generated using χ^2 and BoW with different feature sizes on Enron dataset. 47
3.5 The performance of the classifiers on the feature representation generated using χ^2 and TF-IDF with different feature sizes on Enron dataset. . . . 47

4.1 Autoencoder for Feature Reduction 51
4.2 Autoencoder and Stacked Autoencoder for Feature Reduction 52
4.3 Supervised Feature Reduction 55
4.4 Unsupervised Feature Reduction 56
4.5 BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the SVM performance. 61
4.6 BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the RF performance. 61
4.7 BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the C4.5 performance. 62
4.8 Machine learning classifiers performance on features generated by the BoW + IG approach with varying feature sizes. 66
4.9 Machine learning classifiers performance on features generated by the TF-IDF + IG approach with varying feature sizes. 67
4.10 Machine learning classifiers performance on features generated by the BoW + χ^2 approach with varying feature sizes. 67
4.11 Machine learning classifiers performance on features generated by the TF-IDF + χ^2 approach with varying feature sizes. 68
4.12 Performance of the learning classifiers based on the Enron data using the BoW + PCA approach for feature construction. 70
4.13 Machine learning classifiers based on the Trec07 data using the BoW + PCA approach for feature construction. 71

4.14 Machine learning classifiers based on the Enron data using the BoW + LSA approach for feature construction. . . . 71
4.15 Machine learning classifiers based on the Trec07 data using the BoW + LSA approach for feature construction. . . . 72
4.16 Machine learning classifiers based on the Enron data using the TF-IDF + PCA approach for feature construction. . . . 72
4.17 Machine learning classifiers based on the Trec07 data using the TF-IDF + PCA approach for feature construction. . . . 73
4.18 Machine learning classifiers based on the Enron data using the TF-IDF + LSA approach for feature construction. . . . 73
4.19 Machine learning classifiers based on the Trec07 data using the TF-IDF + LSA approach for feature construction. . . . 73
4.20 Classifiers performance based on ROC curve with features generated using TF-IDF and LSA. . . . 74
4.21 Classifiers performance based on ROC curve with features generated using TF-IDF and PCA. . . . 74
4.22 Classifiers performance based on ROC curve with features generated using BoW and LSA. . . . 75
4.23 Classifiers performance based on ROC curve with features generated using BoW and PCA. . . . 75
4.24 Classifiers performance based on ROC curve with features generated using DM + DBOW (Le & Mikolov 2014). . . . 76
4.25 Classifiers performance based on ROC curve with features generated using DM + DBOW and CS + Autoencoder. . . . 76
4.26 RF and C4.5 performances based on ROC curve with various feature representations using Trec07 dataset for performance evaluation. . . . 77
4.27 SVM performance based on ROC curve with various feature representations using Trec07 dataset for performance evaluation. . . . 77

5.1 Autoencoder and Stacked Autoencoder for Feature Reduction . . . 81
5.2 MLP performance based on ROC curve with varying feature representations. 83
5.3 Comparison of the traditional feature representations and the proposed feature representations on MLP. . . . 84
5.4 Comparison of the traditional feature representations and the proposed feature representations on MLP. . . . 84

List of Tables

3.1 SVM performance on Enron dataset reported in (Mi et al. 2016) 44
3.2 The performance of the classifiers on the feature representation generated using IG and BoW with binary values on the Enron dataset reported in (Mi et al. 2015) 44
3.3 The performance of the classifiers on the feature representation generated using IG and BoW with the term count values on the Enron dataset. . . . 45
3.4 The performance of the classifiers on the feature representation generated using IG and BoW with binary values for feature representation on Enron dataset based on the approach proposed in this Chapter. 45

4.1 Classifiers performance on DM + DBOW and BoW + IG features 58
4.2 Impact of IMDB and Noise removal on classifiers performance based on the features generated using the DM + DBOW approach introduced by Le & Mikolov (2014). 59
4.3 Classifiers performance on features generated by DM + DBOW and CS + Autoencoder on Enron dataset. 59
4.4 Classifiers performance on Trec07 data. 60
4.5 Classifiers performance when DM and DBOW + CS and Autoencoder approach employed on Trec07 60
4.6 RBF kernel with Autoencoder 62
4.7 Laplacian Kernel with Autoencoder 63
4.8 Sigmoid Kernel with with Autoencoder 63
4.9 Cosine Similarity with Stacked Autoencoder 64
4.10 SVM performance with binary features on Trec07 64
4.11 SVM performance with DM + DBOW features on Trec07 64
4.12 SVM performance with DM + DBOW and CS + Autoencoder features on Trec07 65
4.13 C4.5 performance with binary features on Trec07 65
4.14 C4.5 performance with DM + DBOW features on Trec07 65
4.15 C4.5 performance with DM + DBOW and CS + Autoencoder features on Trec07 65
4.16 RF performance with binary features on Trec07 66
4.17 RF performance with DM + DBOW features on Trec07 66
4.18 RF performance with DM + DBOW and CS + Autoencoder features on Trec07 66
4.19 BoW + LSA for feature construction on Enron data 68
4.20 BoW + LSA for feature construction on Trec07 data 69
4.21 BoW + PCA for feature construction on Enron data 69
4.22 BoW + PCA for feature construction on Trec07 data 69

4.23 TF-IDF + LSA for feature construction on Enron data 69
4.24 TF-IDF + LSA for feature construction on Trec07 data 70
4.25 TF-IDF + PCA for feature construction on Enron data 70
4.26 TF-IDF + PCA for feature construction on Trec07 data 70

5.1 MLP performance on varying feature representations 83

Dedication

This work is dedicated to my grandparents (Johannes Makhaya Tladi, Selina Mamoneng Tladi, Linah Letang Diale, and the late Makgale Paulus Diale, *May his soul rest in peace.*).

Acknowledgements

To the living God whom I serve and my forefathers did serve, thank you.

Prof. Turgay Celik, thank you for your availability because you shared many ideas with me. You have also built confidence within me. And to Dr. Christiaan van der Walt thank you for suggesting the research area and allowing me to be in charge throughout my research. I would also like to extend my gratitude to CSIR for giving me the opportunity to conduct my research with them. To the people of Modeling and Digital Science, thank you for your valuable inputs. Wits Intelligence System team has always made valuable comments to my work and I am grateful for that. I will not forget Dr. Vukosi Marivate Data Mining team leader for he made sure we get access to all resources we needed. Abiodun Modupe, in our meeting in 2014, made me realize how important it is for me to do my Masters studies. Your encouragement was important to my research.

To my parents, Dolly Makganyane Diale and Abrahm Sogo Diale, thank you for your support from my early age till Today for you have always believed in me. At times it was difficult to come up with solutions, you were there with words of wisdom. The unconditional love and encouragement I got from my siblings, Jeanlena, Nicole, Brunette, Lutricia and Adelia, is priceless. They always bring joy in my life.

To all my friends, thanks for your support and encouragement, more especially Precious Letjoba Mochai, Kgaria Thapelo Motubatse and Mamosebo Alfred you played a huge role throughout my studies.

It was my grandparents whom have showed me the importance of education. They have always told me how life is not going to be easy in future without education. I am thankful of such wise words.

Publications

Part of this work has been presented and submitted for possible publication as listed below:

1. M. Diale, C. Van Der Walt, T. Celik and A. Modupe, "Feature selection and support vector machine hyper-parameter optimisation for spam detection," Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), 2016, pp. 1-7.

2. M. Diale, T. Celik and C. Van Der Walt, "Investigating Unsupervised Feature Learning for Email Spam Classification," Elsevier's "Computers and Electrical Engineering", submitted, 2017.

Chapter 1

Introduction

An Email allows individuals to send and receive digital messages through a system that interconnects computer networks over the Internet. A *Computer network* enables computing devices to share information. People use email for communication because it is efficient and cost effective (Sterne & Priore 2000). As a result, there are more people using emails as one of their primary sources for communication and sharing information (Bo Yu 2008). However, spammers take advantage of this by distributing unwanted emails to targeted email users.

1.1 Background

Spam emails are known as the unsolicited bulk of emails or junk emails (Bo Yu 2008). They are usually sent to various email users at the same time. Email users' addresses are collected from different websites and chat-rooms. Even though a lot of studies have been conducted to try to mitigate the spam email problem, spammers keep on evolving to overcome introduced spam email filters. Sharma & Sahni (2011) indicated that there was an increase of spam emails sent to various email users. Furthermore, in July 2016, *Internet Security Report* [1] also showed that there has been an increase of global spam rate from 52.7% to 53.2%.

Automated spam email identification is a challenging task because emails differ with intentions. Some of the spam emails consist of links leading to websites that host malicious softwares which can be used to collect sensitive information such as credit card details (Dhinakaran et al. 2007, Grimes et al. 2007, Tseng & Chen 2009) and user credentials

[1] https://www.symantec.com/security-center/threat-report

(Patidar et al. 2011) while other spam emails are used for advertisement purposes (Hayati & Potdar 2008). There are legitimate emails that may contain links that are needed by email users. Additionally, users may also share few advertisements for good reasons.

Spam emails are usually sent to many email users. As a result, they consume a large amount of memory on the email servers (Tak & Tapaswi 2010). They also increase Internet traffic congestion and exhaust lot of time for email users who go through them (Li & Shen 2011). Additionally, email users are vulnerable to viruses that may be contained in a spam email.

Initially, to conduct a spam detection problem email domains and IP (Internet Protocol) addresses were used to block emails from unofficial domains (Hayati & Potdar 2008). Due to an unbounded growth rate of SMTP (Simple Mail Transfer Protocol) servers and boundless use of IP addresses by spammers, it became tricky to sustain the spam blacklist (Xie et al. 2008, Hayati & Potdar 2008). SMTP is the Internet protocol for exchanging information through email and the blacklist consists of IP addresses that are flagged as spam. Researchers have therefore decided to make use of email content for detecting spam emails. Huge vocabulary size in various languages is one of the main difficulties using email content for identifying spam emails.

Regardless of whether or not the same language is used to communicate, email users have different ways of writing messages. Messages of the same meaning can be written with different words. If machine learning (ML) techniques can identify such kind of emails, it will be easier for ML techniques to recognize the difference between spam and non-spam emails. Enabling ML techniques to understand characteristics of a spam email is one of the major difficulty we face in this area of study. This is because spammers always evolve their styles of writing messages. As a result, it becomes difficult for classifiers to categorize emails as either non-spam or spam.

There are plenty of studies that have been conducted on natural language processing (NLP) problems for enabling ML techniques to understand the language used by human beings. Unfortunately, not all of the techniques are good for feature extraction in every problem that involves human language. Feature extraction is a dimensionality related process in which a better representation of the raw data is investigated to avoid redundancy and noise issues (Agarwal et al. 2007, Youn & McLeod 2007, Cheng et al. 2008).

Most techniques that have been used for feature extraction from previous studies such as in (Halder et al. 2011, Mi et al. 2015, Mohamad 2015) related to spam email detection problem rely on the counting of unique terms for feature representation. Utilization of unique terms by email users and spammers for writing email messages leads to a huge

vocabulary size. Hence, we have an issue regarding dimensionality. This is one of the reasons classifiers perform poorly in many studies such as in (Shams & Mercer 2013, Trivedi & Dey 2013, Mi et al. 2015) on Enron[2] dataset (Klimt & Yang 2004). Enron dataset has many unique terms that may lead to sparse feature representation.

One of the most used traditional feature extraction approach based on word count is known as the Bag of Words (BoW). Using the BoW for feature representation is difficult to capture document meaning reason being word order is ignored. In addition, the evolving noisy words increase the feature size. As a result, the generated feature representation slightly loses the main content and the dimensionality issues unfold (Xu & Yu 2010).

There are various approaches that have been utilized to handle the issue of dimensionality. Those approaches include removal of the stopwords (*e.g.* "for", "a", "the", "are", etc.) and stemming (the process of converting a word into its root format, e.g., "walking", "walks" to "walk"). The stopwords removal and Stemming reduce feature space size and improve the performance of ML classifiers (Basavaraju & Prabhakar 2010, Mi et al. 2015). But once they are employed email structure and its main content is being lost (Liu et al. 2005). Hence, it becomes difficult for ML techniques to differentiate between the patterns of spam and non-spam emails.

Furthermore, stemming and removal of the stopwords do not overcome the issue of dimensionality due to the use of noisy words by spammers and email users, for example the word "love" can be written as "luv". This kind of words differs with individuals. Usually, people use the previous and coming terms in a sentence to understand noisy words. This clearly shows that stopwords might play an important role in understanding email content with noisy words.

Due to the use of noisy words by email users and spammers we now have a repeatedly increasing vocabulary size. As a result, we face issues regarding the large dimensional feature space, which is a burden to classifiers. These issues can be handled by making use of the feature selection techniques that retain important features (Cheng et al. 2008, Jinzhu et al. 2008).

Feature extraction and feature selection, are the processes that play a significant role before training the classifiers. The use of these processes involves generating a good feature space for training classifiers (Jinzhu et al. 2008, Guzella & Caminhas 2009). They further handle issues regarding the high dimensional feature space. During the feature selection process, features which are more informative are being identified.

[2]http://www.aueb.gr/users/ion/data/enron-spam/

The issue of higher dimensionality is not only limited to NLP based problems like email spam detection. Most studies that make use of real life data face the same problem. Data collected from real world problems in most cases is found to consist of a large number of dimensions. Some of the features in this type of data are found to have inadequate information for learning a model. Regrettably, making use of those features for training a model may affect the performance of a learning model. Therefore the introduction of dimensional reduction methods in such cases is needed.

Principal component analysis (PCA) (Abdi & Williams 2010) is one of the techniques used for dimensional reduction. It is however known that PCA has some drawbacks since it is unable to maintain complex structure in real-world input data *e.g.* in image recognition when traditional PCA is utilized to reduce the number of dimensions in an image, important information is being lost (Belhumeur et al. 1997). Hence, it is a requirement to employ feature reduction techniques that can handle any kind of numeric feature representation.

Autoencoder (Baldi 2012) is an ML approach regarded as the artificial neural network (ANN), which is usually employed for reducing the feature space dimension. Unlike PCA, Autoencoder can handle both linear and non-linear mapping which means in most real life problems Autoencoder will generate better feature representation as compared to PCA.

A lower dimensional feature space with good class separability most of the time ensures a good generalization in classifiers. Furthermore, classifiers learn at a quicker rate and computational memory issues may be avoided.

There are various classifiers that have been employed for spam email identification tasks previously. Binary classification is one of the approaches that has been used in most studies for email spam identification, it makes use of two classes from a labeled dataset to train the classifier. This kind of a classification approach is regarded as a supervised learning.

From the previous studies, there have been several supervised models that were considered for classifying emails. In their work, Gansterer & Pölz (2009) and Youn & McLeod (2007) employed support vector machine (SVM) and a decision tree J48 classifiers. It was found that SVM and J48 are good classifiers for email spam classification problem. SVM learns a hyperplane with maximum margin to delineate positive examples from negative examples and it is advantageous because of its ability in handling high dimensional feature space and noisy data (Li et al. 2010). On the other hand, a decision tree J48 is a java implementation of C4.5 which was initially introduced by Quinlan (2014).

C4.5 utilizes information gain (IG) for measuring an extent to which each feature delineates instances with respect to their labels (Quinlan 2014). C4.5 is advantageous due to its ability to handle data with missing values (Patil & Sherekar 2013).

Most studies such as in (Youn & McLeod 2007, Gansterer & Pölz 2009, Kumar et al. 2012, Mi et al. 2015) shows that support vector machine (SVM) is promising in accurately classifying emails. However, they do not discuss how kernel functions were selected and how they were optimized in terms of hyperparameters. The performance of a classifier such as SVM is dependent on the selection of the kernel function and the choice of the hyperparameters for each feature representation. Taking hyperparameter optimization and optimal selection of kernel function into consideration may improve SVM performance.

1.2 Problem Definition

One of the challenges with feature extraction process is ensuring that the features extracted from an email have meaningful information about the emails. The feature selection techniques determine features that enable classifiers to differentiate between spam emails and non-spam emails. Additionally, feature selection techniques produce feature representation of the original raw data in a lower feature space to ensure that classifiers learn at a quicker rate while not losing important information.

Word count based approaches such as BoW which are mostly used for feature extraction have several drawbacks. They do not capture email structure such as word sequence. In addition, they make classifiers suffer from sparsity. It should be noted that the stopwords are usually being removed in spam detection problems. They are regarded as non-informative due to their appearance in each and every email document. However, the removal of the stopwords brings many problems because it is difficult for even a human being to make sense out of random words without some sequential order. Hence, it is expected that it will be even worse when stemming procedure is employed to interpret words without sequential order.

Removing the stopwords and not considering the word sequence we end up with random words which we cannot explain their meaning. It is important to transform feature representation of an email content from a higher feature space to a lower feature space without losing important information. Taking into consideration the sequential structure of sentences, keeping the stopwords which connect meaningful words and not employing stemming will ensure that a meaningful feature representation about the emails is generated.

Hyper-parameter optimization is an important step in training the optimal classifier but is often overlooked. Although this is a challenging task, it needs to be addressed for each feature representation to boost classification accuracy of SVMs.

The main problem in this study is generating feature representation that retains significant features in a lower dimensional feature space for identifying spam emails.

1.3 Research Questions

- Feature extraction is one of the pre-training phases which have influence over the performance of classifiers. Previously, stopwords have been regarded as non-informative words in determining the patterns of spam and non-spam emails. The sequential structure of sentences was not taken into consideration in most studies when generating feature representation for identifying spam emails. Our first research question is as follow:
  - By using a technique that takes word order into consideration without applying stemming and stopwords removal, can we generate more robust features for spam detection problem from unstructured data? But also ensuring that features extracted capture semantic relationship of words, that is, words of similar meaning should be closer to each other in terms of numeric vector representation.
- Feature reduction is also an important pre-training phase for classifiers. PCA is one of the unsupervised feature reduction techniques that have been used most previously. However, Autoencoder is a preferred feature reduction technique since it is not limited to a linear map like PCA but it can handle both a nonlinear and linear map. For this reason, our second research question is as follows:
  - Does the use of Autoencoder for feature reduction improve the performance of classifiers on email spam detection problem?
- Between DTs and SVM which one is optimal for email spam classification?
- Between the proposed feature representation approach and the other state of the art feature representation approaches, which one captures more robust features for identifying spam emails?

1.4 Research Objectives

In this research project, we aim to:

- investigate the performance of SVM with different kernel functions and their optimal kernel parameters.
- compare the performance of SVMs and decision trees for email spam classification.
- investigate how Autoencoder and Stacked Autoencoder affect the performance of classifiers when used for feature learning.
- investigate the use of Stacked Autoencoder and Multilayer perceptron for email spam identification.
- investigate the impact of Cosine Similarity (CS) and distance based kernel functions for feature transformation on classifiers employed for email spam detection.

1.5 Delineations and Limitations

This section covers limitations and delineations encountered in this study. Regarding hyperparameter optimization, we only use one approach known as grid search algorithm with k-fold cross validation. We will define an interval for identifying optimal parameters.

Kernel functions that are being investigated are distance based kernel functions. We only focus on Radial Basis Function (RBF), Linear, Polynomial and Sigmoid kernel functions. Other distance based kernel functions that are not mentioned will not form part of this study. String kernel functions which were employed in other studies from the literature will not form part of this study as well.

This research will only use the email content approach for identifying spam emails. There are other approaches for identifying spam email beside using email content. Those approaches will not be considered in this study for performance comparison. We will use publicly available dataset for evaluating email spam filtering models. The Enron data and Trec07 data, which will be discussed in detail in Chapter 3 and Chapter 4 respectively, are the datasets to be utilized in this research project.

It is worth noting that the proposed approach is not only limited to spam email problem. It can also be employed to other natural language processing problems. However, in this study, we will only focus on spam email detection problem. Furthermore, there are many natural language processing techniques which will not be employed for comparison. We only cover few traditional techniques which have been utilized in recent studies for comparison to our proposed approach.

The proposed approach is not limited to one language, however, we make an assumption that English is the only formal language used for writing emails. Hence, we will not evaluate the model on other email datasets which are written in any other language. Thus, every word which is not found in English vocabulary is regarded as a noisy word.

1.6 Research significance

Many NLP algorithms have been employed such as Term Frequency - Inverse Document Frequency (TF-IDF) and BoW for feature extraction on spam email detection problems where email content is used for feature extraction. These approaches suffer from sparsity, word order is not captured and they also do not capture semantic relation of words. In this study, the first contribution is on using neural network approach in which semantic relation of words is captured as well as word sequence. The Distributed Memory (DM) and Distributed Bag of Words (DBOW) approach that handles most issues encountered in TF-IDF and BoW will be utilized for feature representation.

Most studies apply stopwords removal and stemming where every single word is converted to its root format. We realize that with word count based approaches for feature extraction, stopwords increase sparsity of feature representation. To ensure that the main message from the words which are not considered stopwords is not lost, we will not apply stemming and stopwords removal. We will introduce feature learning approaches such as Autoencoder and Stacked Autoencoder to evaluate their impact on the performance of classifiers.

Furthermore, we investigate optimal hyperparameter of kernel functions for SVMs on various feature extraction approaches unlike in previous studies were SVM was considered for spam email classification but not clearly indicated how they selected kernel functions and optimized the kernel functions in terms of hyperparameters (Youn & McLeod 2007, Gansterer & Pölz 2009, Kumar et al. 2012, Mi et al. 2015).

1.7 Overview

In this dissertation, we will investigate unsupervised feature learning approaches which ensure class separability in a lower dimensional feature space. Chapter 2 covers the literature review. In Chapter 3, we investigate the optimal number of features for training classifiers and also investigates optimal hyperparameters on various traditional feature representation approaches. In Chapter 4, we introduce unsupervised feature learning approach as a preprocessing phase to learn models for identifying spam emails.

The proposed approach is compared to other traditional supervised and unsupervised feature construction approaches. In Chapter 5, we investigate Stacked Autoencoder for feature learning and employ MLP on various feature representations to assess its performance. Finally, the conclusion of the study and future work discussion are made in Chapter 6.

Chapter 2

Literature Review

There are various approaches that have been considered to filter or block spam emails from previous studies. Other methods are network based taking connectivity and network distance into consideration (Song et al. 2011). Additionally, there is Blacklist approach that makes use of IP addresses to identify spam emails. Email content based approaches have been studied a lot lately, which make use of the email message content to identify spam emails. In this chapter, we review previous work related to the spam detection problem.

2.1 Callback Verification

According to the Internet Protocol (IP), it is a requirement for any email or data being sent through the Internet to have an IP address of the sender. For example, when an email is being sent the header takes in both the source IP address and the destination address. This is one of the reasons methods like the callback verification were introduced for identifying spam emails. Callback verification approach makes use of the source IP address to determine if the email sender is a spammer. However, there are some disadvantages that have been encountered regarding callback verification approach. As much as it is effective in terms of real-time identification, spammers can always forge the source IP addresses. In that case, an IP address of a legitimate user can be used for spamming purposes. Hence, more people might be reported as spammers although they are not (Heron 2009).

2.2 Background Checks

Spammers usually make programs that can automatically open new email accounts. This makes it easier for the spammers to dynamically change their identities if they were blocked in the past. CAPTCHA (Von Ahn et al. 2003) is one of the programs that has been introduced trying to overcome the issue of email accounts being opened using computer programs. The main aim of CAPTCHA is to determine if the new email user is the real person or just a computer program. CAPTCHA uses distorted text asking users to retype them. This is difficult for machines to identify them. However, spammers overcome CAPTCHA by only using email services that do not make use of CAPTCHA for background checking (Hayati & Potdar 2008). Furthermore, CAPTCHA may only reduce the number of email accounts being opened automatically. But CAPTCHA does not prevent the spammers from opening new email accounts because they can do that manually.

2.3 Honeypots

In contrast to the background checks methods, the honeypots must first wait for the spam activities occur. Honeypots detect spam emails by capturing information regarding malware activities. They can disclose important information about spammers, such as their IP addresses and methods utilized by them. Spamtrap is another type of Honeypots, its main aim is to receive spam emails. Those types of email addresses can only be accessed by spammers since they usually use automated programs to collect email addresses from various web pages. For spammers to overcome Honeypots, they need to detect them (Krawetz 2004, Lee, Caverlee & Webb 2010).

One of the drawbacks regarding the honeypot is that if it is avoided then there are no malicious activities that will be detected (Obied 2006). Furthermore, since usually, spammers use botnets (Barford & Yegneswaran 2007) to sent spam emails, it then means legitimate email users whom their accounts were compromised by spammers will suffer from honeypots. This is because their email addresses will be used to send the malicious emails or spam emails. Hence, there is a need of models which will be able to identify spam emails without any background check of the user.

2.4 Vector Space Models

Spam filtering approaches based on the use of the Vector Space Models (VSMs) are kind of the filtering approaches that require no background check. They only need to analyze the characteristics of the email content. Spam email filtering techniques which are based on VSM usually require feature construction techniques and feature reduction techniques to generate VSMs. Feature construction is a procedure in which the original content of emails is transformed into a numeric format. Feature reduction is mostly utilized to identify more robust features for delineating spam emails from legitimate emails. There are various feature construction and feature reduction approaches which were proposed previously for spam filtering purposes.

2.4.1 Feature Extraction

The primary purpose of categorizing text documents is to ensure that every document is well grouped into the topic of interest (Günal 2012). Naturally, text categorization models need a feature extraction process that well summarizes the raw data. This is because the efficiency of classification methods rely on how features were extracted.

Most previous approaches for email spam filtering involve the use of BoW (Liu et al. 2005), which is one of the most famous techniques used for feature representation. This feature representation approach determines counts of every unique term appearing in a specific document. Many studies from the literature based on spam filtering problem such as (Clark et al. 2003, Ergin & Isik 2014, Mi et al. 2015) employed BoW for feature construction. However, there are limitations that have been encountered regarding the use of BoW. The BoW can overfit classifiers due to a large vocabulary that leads to a huge sparse feature space (Calderbank et al. 2009, Ferreira & Figueiredo 2011).

2.4.1.1 Bag of Words

Recent studies have investigated drawbacks of BoW and introduced new approaches which can overcome some of the BoW issues. For example, text categorization problems such as in (Le & Mikolov 2014) addresses the issues encountered with BoW. However, in many recent studies, BoW has been utilized for feature representation on spam filtering problems. For example, Mi et al. (2015) employed BoW for feature construction and classifiers being utilized still show better performance on spam detection problems.

The BoW approach is based on term count (*e.i.,* term frequency) where the feature values can be in word frequency or binary representation. As for binary feature representation,

all features are represented as 1 or 0. When term frequency based features are employed, then in each email document every word is used as a feature. This means that all words or characters from all emails are used as features. The feature value, in this case, is the frequency value of that term in a specific email.

Given a set E of email documents $E_i \in E$ and the term $t_j \in E_i$ then the total count of t_j in E_i denoted by $f(t_j)$ is regarded as a feature value. Whereas in terms of binary representation, the value of $f(t_j)$ is 1 only if t_j is present in E_i, otherwise $f(t_j)$ is 0.

There are various limitations which have been encountered working with BoW for text categorization problems. Firstly, BoW ignores word order in the document as well as word semantic is not taken into consideration (Le & Mikolov 2014). As a result, documents with many similar words but different meaning end up being treated as documents of the same class. Things which usually make documents of different meaning end up looking similar when represented as BoW are stemming and stopwords removal. For example, if the following two sentences

- John loves their music.
- John does not love their music.

are represented as BoW together with stopwords removal and stemming being employed, we end up with the following BoW for both of them:

- John, love, music

This means we cannot even recognize which one is negative or positive. This clearly shows that by removing stop words we can lose meaning of the sentence. Similarly with stemming, the word *Introduction* and *Introducing* usually after applying stemming on them they get transformed into *introduce* or *intro* (Jivani et al. 2011). This means they end up having the same meaning because it is not considered where and how they are being utilized.

As much as there are some limitations regarding the use of stopwords removal and stemming, it is important to apply them when making use of BoW. Because they reduce sparsity which can result in overfitting for learning models (Silva & Ribeiro 2003). Furthermore, these noisy words known as stopwords when they are not removed before training the learning classifier, they affect ML classifiers because they appear in almost every document. This means in some cases they are not meaningful for text classification purposes (Zhang & Li 2007).

2.4.1.2 Term Frequency - Inverse Document Frequency

The use of BoW approach in most cases makes frequent terms to appear more significant for classification. However, is not always the case that frequent terms are more important for identifying spam emails. Term Frequency-Inverse Document Frequency (TF-IDF) is an intensified feature representation of BoW. Unlike BoW that may only look for a term availability or evaluate its frequency in an email, TF-IDF goes beyond term frequency by ensuring that more frequent terms do not dominate terms that rarely appear in the email documents. For avoiding the dominance of frequent terms over rare terms, TF-IDF incorporates Inverse Document Frequency (IDF) approach with the Term Frequency (TF) approach.

In contrast to using term frequency, the logarithmic frequency can be utilized, which is given by:

$$\log(1 + f(t_j)). \tag{2.1}$$

More additionally, instead of logarithmic term frequencies, inverse document frequencies can be used which are expressed as:

$$1 - \frac{\alpha}{f(t_j + \alpha)}, \alpha > 0. \tag{2.2}$$

Both the logarithmic frequency and inverse frequency were introduced in (Leopold & Kindermann 2002). IDF determines the significance measure of the term t_j by further considering the total count of that term in the set E of email documents. IDF makes an assumption that terms not appearing more often in E, are more significant for classification purposes. The IDF of term t_j in email $E_i \in E$ is given by:

$$\log(\frac{|E|}{df(t_j)}), \tag{2.3}$$

with $df(t_j)$ denoting the total count of email documents containing term t_j and $|E|$ denoting total count of email documents. As a result, we have TF-IDF given by:

$$f(t_j) \times \log(\frac{|E|}{df(t_j)}). \tag{2.4}$$

TF-IDF differs with BoW because TF-IDF takes the importance of the term $t_j \in E_i$ into consideration. However, BoW representation is only based on term frequency. This makes TF-IDF to have a better representation as compared to BoW. The major disadvantage of both TF-IDF and BoW is that they all loose sentence or document structure. They are all dependent on term frequencies. That might make most of the documents loose actual content meaning.

2.4.1.3 N-gram

Since it has been discovered that loss of the document structure might negatively affect feature representation of the text documents, various language models have been employed for various Natural Language Processing (NLP) tasks to capture word order. It is important to note that taking word order into account is not only good for generating a better feature representation. However, the consideration of word order might be helpful for understanding noisy words used by email users and spammers. Hence, language models may play a significant part to overcome various issues in text mining problems.

N-gram is a language model approach which has been employed previously such as in (Fusilier et al. 2015) on spam detection problem. It was employed for feature extraction. N-gram captures term order by giving a certain term co-occurrence sequence a probability value. For instance, given a sequence $t_1 t_2 \dots t_{N-1}$, N-gram determines the probability value of having t_N as the next term.

Overcoming the noise issues in a text data Fusilier et al. (2015) applied N-gram at the character level. Their approach has shown promising performance for identifying spam activities from a noisy data. N-gram makes use of maximum likelihood estimate (MLE) to estimate $P(t_N|t_{N-1} \dots t_2 t_1)$ the probability of t_N given the history $t_{N-1} \dots t_2 t_1$, which makes an assumption that the probability of every term (*i.e.*, word or character) is dependent on the previous sequence of terms. This concept is known as the Markov assumption (Mikolov et al. 2011). The value of $P(t_N|t_{N-1} \dots t_2 t_1)$ is given by:

$$\frac{f(t_N t_{N-1} \dots t_2 t_1)}{f(t_{N-1} \dots t_2 t_1)}, \tag{2.5}$$

where $f(x)$ denotes the number of times the sequence x appears in E a set of emails or any training corpus. Although it has been shown from previous studies that N-gram may lead to a better feature representation by considering term sequence, Mikolov et al. (2011) argued that since usually two previous terms are used to predict the current term in a sequence, it becomes difficult to capture enough semantics. Furthermore, due to the large vocabulary, the N-gram approach generates sparse representation. As a result, we will have classifiers overfitting the training data.

Feature construction techniques leading to sparse feature representations can be avoided by using distributed feature representation models. These kinds of representations are based on neural language models. One of their major advantages is capturing more semantic properties (Le & Mikolov 2014, Turian et al. 2010). In addition to that, they

handle some of the high dimensional feature space issues by generating more dense feature representations.

2.4.2 Feature Reduction

Many studies on spam email filtering problem overcame the high dimensional feature space issue encountered in feature construction approaches such as the BoW and TF-IDF by making use of the feature reduction techniques. The main purpose of the feature reduction techniques is to retain features that ensure class separability in a lower dimensional feature space. This is because most feature values generated may not be important for prediction or classification. Feature reduction process plays an important part by making use of the feature selection techniques to determine more robust features for training and testing the classifiers. This section presents various feature reduction techniques that have been employed previously in studies related to email spam filtering.

2.4.2.1 Information Gain

One of the most used feature reduction technique in text categorization problems is Information Gain (IG). Particularly in email spam filtering studies, IG has shown to have a positive impact on the performance of classifiers. Usually, in most studies IG is introduced to select robust features for learning the classifier from the feature representation generated using the BoW. In (Mi et al. 2015) when a deep learning approach known as stacked autoencoder was introduced for email spam filtering, BoW with binary features was employed together with IG for feature representation. The study has shown that BoW and IG can still feed good features to deep learning techniques when employed for spam filtering.

IG examines the importance of each term with respect to the categorization label of a data instance. Let $c_l \in \{+1, -1\}$ be a class label or instance category. IG makes use of Kullback-Leibler (KL) distance (Kullback 1997) to evaluate the importance of t_j or its impact in determining the possibility of E_i belonging to the category c_l. This is given by:

$$\sum_{l} P(c_l) \log \frac{1}{P(c_l)} - \sum_{l} P(c_l|t_j) \log \frac{1}{P(c_l|t_j)}. \tag{2.6}$$

IG is regarded as a supervised feature reduction technique because it requires class labels to evaluate the importance of the feature variables.

2.4.2.2 Chi Square

A popular statistical method that has been employed many times for feature selection is known as chi-square (χ^2). Similarly to IG, χ^2 requires class labels to evaluate the robustness of each feature variable. The main objective of χ^2 is to determine the independence of two given variables or their correlation.

For feature selection χ^2 evaluates the co-occurrence of the feature variable $t_j \in E_i$ and the corresponding class label c_l of the instance E_i. This means that the value of χ^2 will be 0 when t_j and c_l are not correlated. In such a case, t_j will not be significant to train the learning classifier.

Given $t_j \in E_i$ and the corresponding class label c_l of $E_i \in E$, then χ^2 is given by:

$$\frac{n(\alpha\delta - \zeta\beta)^2}{(\alpha+\zeta)(\beta+\delta)(\alpha+\beta)(\zeta+\delta)}, \tag{2.7}$$

where α denotes the total co-occurrence of c_l and t_j, ζ is the total occurrence of c_l in the absence of t_j, δ is the total count of the times neither c_l nor t_j being present, β is the total occurrence of t_j in the absence of c_l and n is the total count of instances in E.

In most studies based on text categorization such as in (Moh'd A Mesleh 2007), χ^2 and IG are found to be more effective for feature selection. This was observed after learning classifiers were employed for classification task and showed good performance. These feature selection techniques, χ^2 and IG, were applied on the numeric representation generated by TF-IDF from a text data. χ^2 and IG where compared to other well-known feature selection methods such as document frequency thresholding (Yang & Pedersen 1997), Term Strength (Yang & Wilbur 1996) and Mutual Information (Yang & Pedersen 1997).

It is disadvantageous for reducing feature dimensional space using χ^2 and IG because all terms or feature variables are treated independently. Hence, it is a challenging task to understand the actual content of the message. From one word, it is not easy to conclude the overall meaning in the message. To have a clear meaning of the message, grammar needs to be taken into consideration.

In addition, these techniques, χ^2 and IG, for them to evaluate the importance of the feature variable t_j they only consider the co-occurrence of t_j and c_l. Meaning that keywords or characters individually are important for training the learning classifier to be able to identify spam emails. Insignificant terms (*i.e.*, keywords and characters) are discarded from the training data to avoid overfitting issues.

2.4.2.3 Singular Value Decomposition

In contrast to the χ^2 and IG that rely on evaluating term importance, Singular Value Decomposition (SVD) (Wall et al. 2003) ensures that the data variance of the higher dimensional feature space is preserved in the lower feature dimensional space. Furthermore, SVD utilizes the unsupervised approach that does not require any knowledge about categories of the instances.

For instance, with the BoW approach for feature representation, we might have a sparse representation consisting of d dimensions. Using the SVD approach we can generate $\tilde{X}$ the approximation of the original feature space X such that $\tilde{X}$ has $\tilde{d}$ feature variables, where $\tilde{d} < d$. One of the most important phases before employing the SVD approach is to go through the pre-processing phase which involves data translation and data scaling.

Data translation is a procedure known as mean centering because it translates the data center to the origin. For a given set of feature vectors $X = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\}$, where $\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \ldots, x_d^{(i)}] \in \mathbb{R}^d$, data translation of each feature vector $\mathbf{x}^{(i)}$ is performed as follows

$$\mathbf{x}^{(i)} \leftarrow \mathbf{x}^{(i)} - \boldsymbol{\mu}, \tag{2.8}$$

where

$$\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}^{(i)}. \tag{2.9}$$

Data scaling, also known as *variance scaling*, normalizes each feature variable $x_j^{(i)}$ of a feature vector $\mathbf{x}^{(i)}$ to ensure that all feature variables have the same variance. This is done by dividing each feature variable by its variance σ_j^2 calculated over the data set, *i.e.*,

$$x_j^{(i)} \leftarrow \frac{x_j^{(i)}}{\sigma_j^2}, \tag{2.10}$$

where

$$\sigma_j^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_j^{(i)})^2. \tag{2.11}$$

The normalization is more important when feature variables have varying scales. For instance, in the text processing problem where the BoW is employed for feature representation, the value of each feature variable $x_j^{(i)}$ corresponds to the total counts of a term t_j in a document E_i. This can result in a case where all the feature variables have a different scaling. Hence, the data pre-processing phase plays an important role.

One of the advantages of SVD is that the feature vector $\mathbf{x}^{(i)} \in \mathbb{R}^d$ can be transformed into a lower dimensional feature vector $\tilde{\mathbf{x}}^{(i)} \in \mathbb{R}^{\tilde{d}}$, where $\tilde{d} \leq d$, ensuring that the most

important information that explain $\mathbf{x}^{(i)}$ is preserved in $\tilde{\mathbf{x}}^{(i)}$. This is done by initially decomposing X as:

$$X = U\Lambda V^T, \tag{2.12}$$

where the symbol U represents a left singular vector of X, which is given by

$$U = \left[\mathbf{u}^{(1)}, \mathbf{u}^{(2)}, \ldots, \mathbf{u}^{(n)}\right] \in \mathbb{R}^{n \times n}.$$

The symbol V is given by

$$V = \left[\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots, \mathbf{v}^{(d)}\right] \in \mathbb{R}^{d \times d}$$

and it is known as a right singular vector belonging to X. $\mathbf{u}^{(i)}$ and $\mathbf{v}^{(i)}$ are the unit orthonormal vectors. Λ is an $n \times d$ diagonal matrix consisting of the singular values.

The $\tilde{X}$ can be generated by assigning all low ranked singular values in the the singular matrix Λ to zero values. This means we have

$$\tilde{X} \approx X, \tag{2.13}$$

where all the low ranked columns in $\tilde{X}$ consist of zero values. If we discard all zero columns, then $\tilde{X}$ will be decomposed as:

$$\tilde{X} = U_{\tilde{d}}\Lambda_{\tilde{d}}V_{\tilde{d}}^T, \tag{2.14}$$

where $\tilde{d} < d$.

One of the mostly used multivariate statistical technique in the text processing problems which is associated with the SVD is known as the Principal Component Analysis (PCA) (Abdi & Williams 2010). The main aim of this technique, PCA, is to transform the feature representation X into the new space of uncorrelated variables. Those uncorrelated variables are known as principal components.

The SVD approach can be employed to calculate those principal components (Xie et al. 2017). PCA is one of the feature reduction technique that has been employed previously to ensure that hidden patterns about feature representations generated are revealed. That has let to acceptable performance of classifiers for identifying spam emails and other malicious activities throughout the Internet.

Masud et al. (2007) employed PCA for reducing feature size of the vector space model which was constructed for email worms detection. The email worm is a code used for malicious purposes by infecting a computer device and also distributing its copy throughout the Internet. In this study (Masud et al. 2007) authors make use of PCA to extract meaningful features which will be helpful for identifying email worms.

Features being used includes availability of the

- HTML tags;
- images to avoid attacks through buggy image processor;
- hyper-links leading to the infected sites;
- binary attachments.

Masud et al. (2007) further explained that PCA can be used for finding hidden patterns of the data. In their study, Masud et al. (2007), a decision tree C4.5 was employed for feature reduction and it is found to lead to better performance in identifying email worms compared to when PCA alone is used for feature reduction. Classifiers which were taken into consideration includes SVM and Naive Bayes (NB).

Goodman et al. (2015) in contrast to Masud et al. (2007) employed PCA for feature reduction. The objective of this study (Goodman et al. 2015) was to tackle two cyber-crime problems, which are spam detection in Short Message Service (SMS) and malicious movements detection in the network. This study (Goodman et al. 2015) is related to email spam detection because in both email spam detection problem and SMS spam detection problem, message content can be used for feature extraction. The generated features can be used to train learning models that will be able to identify patterns of spam related messages and legitimate messages.

In contrast to Masud et al. (2007), for feature representation Goodman et al. (2015) used anomaly scores. Use of anomaly scores has resulted in an improved Receiver Operating Characteristic curve (ROC) (Wright 2005) when employed on spam detection problem. One of the major disadvantages in this study (Goodman et al. 2015) is that N-gram approach was used for feature extraction where N was set to 2 and 3. The major problem is that SMS users these days make use of slangs and many noisy words. This can result in a case where we have a very sparse representation. Furthermore, words which are more related to each other with meaning although are different with character co-occurrence may not be captured. This becomes a challenge for feature reduction techniques to determine more robust features with the evolving vocabulary in the SMS platform.

2.4.2.4 Latent Semantic Analysis

The main goal regarding PCA is to project the feature space X into the lower dimensional feature space. In contrast to PCA, LSA aims to grasp semantic relations regarding the terms found in the text document (Turney et al. 2010, Chen & Lin 2014). LSA was introduced to reduce the number of dimensions by capturing hidden patterns within documents (Qian et al. 2010). An LSA technique employs the SVD approach to compute feature representation. The generated SVD returns highly ranked features across all documents.

In this case the rows of the VSM are the term vectors $\mathbf{t}_j$ and the columns are the document vectors $\mathbf{x}^{(i)}$. This means we have $\mathbf{t}_j$ given by:

$$\mathbf{t}_j = \begin{bmatrix} x_1^{(i)} & x_2^{(i)} & x_3^{(i)} & \dots & x_n^{(i)} \end{bmatrix}$$

and $\mathbf{x}^{(i)}$ give by:

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_j^{(1)} \\ x_j^{(2)} \\ \vdots \\ x_j^{(d)} \end{bmatrix}$$

The mutual relationship between these two vectors, $\mathbf{x}^{(i-1)}$ and $\mathbf{x}^{(i)}$, can be evaluated by calculating their dot product which is given as

$$\mathbf{x}^{(i-1)} \cdot \mathbf{x}^{(i)} = \mathbf{x}^{(i-1)^T} \mathbf{x}^{(i)}. \tag{2.15}$$

Hence,

$$\mathbf{x}^{(i-1)} \cdot \mathbf{x}^{(i)} = \sum_{j=1}^{m} x_j^{(i-1)} x_j^{(i)}. \tag{2.16}$$

Similarly for determining the correlation between two terms, say $\mathbf{t}_1$ and $\mathbf{t}_2$ will be defined as

$$\mathbf{t}_1^T \mathbf{t}_2 = \sum_{i=1}^{n} x_1^{(i)} x_2^{(i)}. \tag{2.17}$$

This means we can have X^TX which will give the correlation measures between all $\mathbf{x}^{(i)} \in X$. XX^T will also give the correlation measures between all $\mathbf{t}_j \in X$. If all documents in X are not correlated, then X^TX will be a matrix of zero elements.

According to the SVD property X can be expressed as

$$X = U\Lambda V^T. \tag{2.18}$$

Using SVD, then

$$X^TX = (U\Lambda V^T)^T(U\Lambda V^T). \tag{2.19}$$

As a result, we have X^TX

$$X^TX = (V\Lambda^T U^T)(U\Lambda V^T). \tag{2.20}$$

Now, because Λ is a diagonal matrix we then have X^TX expressed as

$$X^TX = (V\Lambda U^T)(U\Lambda V^T). \tag{2.21}$$

Hence,

$$X^TX = V\Lambda U^T U\Lambda V^T. \tag{2.22}$$

But U is an orthonormal vector, this means

$$X^TX = V\Lambda^2 V^T. \tag{2.23}$$

Therefore V consists of X^TX eigenvectors. Similarly to XX^T using SVD can be decomposed as

$$XX^T = (U\Lambda V^T)(U\Lambda V^T)^T = (U\Lambda V^T)(V\Lambda^T U^T). \tag{2.24}$$

Since V is an orthonormal vector and Λ is a diagonal matrix, then

$$XX^T = U\Lambda^2 U^T. \tag{2.25}$$

As a result, we have U consisting of eigenvectors of XX^T. This means we can have low ranked approximation of both XX^T and X^TX given by

$$XX^T = U\Lambda^2 U \approx U_{\tilde{d}}\Lambda^2_{\tilde{d}}U^T_{\tilde{d}} = X_{\tilde{d}}X^T_{\tilde{d}} \tag{2.26}$$

and

$$X^TX = V\Lambda^2 V^T \approx V_{\tilde{d}}\Lambda^2_{\tilde{d}}V^T_{\tilde{d}} = X^T_{\tilde{d}}X_{\tilde{d}}. \tag{2.27}$$

There are various studies in which LSA has been utilized on a spam email filtering related problem such as in (Wan et al. 2015). LSA has shown to be a promising feature representation technique for identifying email messages which are spam. However, Cai et al. (2012) have shown that since LSA is a co-occurrence based technique, it will have difficulties in a noisy data. Currently, spammers use noisy words to trig most spam filters. As a result, we have an increasing vocabulary. For LSA this is a serious challenge, because LSA is mainly based on term co-occurrence and does not take grammar into consideration. This result in a situation where the introduction of twisted words make data representation to be noisy which negatively affects LSA feature representation.

2.4.2.5 ReliefF

SVD based feature reduction techniques, LSA and PCA, do not take into account the information regarding the class labels of instances. Hence, the class separability may not be retained in some cases. ReliefF in contrast to LSA and PCA takes into account discriminatory information (Agre & Dzhondzhorov 2016). χ^2 and IG also take into account discriminatory information. However, χ^2 and IG are based on an assumption that there is no dependence among all feature variables (Kononenko et al. 1996).

ReliefF is a feature reduction method that can be used for regression and classification (Xu et al. 2016). However, in (Kumar et al. 2012) it is utilized for feature reduction on a spambase data from UCI repository before employing ML classifiers for identifying spam emails. In cases where ReliefF is employed for feature reduction, its main objective is to evaluate discriminative features (Jacob & Ramani 2011).

ReliefF originates from the Relief feature selection algorithm (Kira & Rendell 1992) which could only handle binary class tasks (Arauzo-Azofra et al. 2004). Suppose we are given $\mathbf{x}^{(i)} \in X$, where $\mathbf{x}^{(i)}$ is a binary vector of size d. Relief determines nearest neighbors of every data point $\mathbf{x}^{(i)}$ from k selected instances. Those k instances are randomly selected initially and the value of the constant k is determined by the user.

For each $\mathbf{x}^{(i)}$, the euclidean distance measure is being utilized to determine how close or how far is a data point $\mathbf{x}^{(i)}$ with reference to all other data instances in different class categories. A weight vector $\mathbf{v}$ is initialized to zero. We let $\mathbf{v}_H$ and $\mathbf{v}_M$ be a near hit and a near miss respectively.

Since at every iterative step a feature vector $\mathbf{x}^{(i)}$ is selected, we then let $\mathbf{v}_H$ be a vector of the same class as $\mathbf{x}^{(i)}$ with minimum euclidean distance. $\mathbf{v}_M$ is a vector of minimum euclidean distance from data point $\mathbf{x}^{(i)}$ belonging to a different class. Then $\mathbf{v}$ at every iterative step is updated by

$$\mathbf{v}_i = \mathbf{v}_i - (\mathbf{x}^{(i)} - \mathbf{v}_H^{(i)})^2 + (\mathbf{x}^{(i)} - \mathbf{v}_M^{(i)})^2. \tag{2.28}$$

At every l^{th} iteration, each feature variable is divided by l. A threshold value τ is determined by inspection so that feature weights with a value of more than τ are selected (Dash & Liu 1997).

Some of the limitations to Relief is that it cannot handle data which is noisy and have redundant features (Kononenko et al. 1997). Kononenko (1994) introduced ReliefF to overcome mentioned drawbacks regarding Relief. However, with ReliefF we still have some difficulties again. Since ReliefF is based on the relevance score for identifying more robust features to classify a data point, in some cases all feature variables may be given a score of high rank. Hence, it may be a challenge to filter out insignificant feature variables (Agre & Dzhondzhorov 2016).

2.5 Binary Classification

Feature extraction and feature reduction are the pre-processing phases which are usually employed before using a learning classifier to automatically identify spam emails. Lee, Kim, Kim & Park (2010) explained that receiving a small amount of spam is easier to identify and delete them from your mail inbox. However, previous studies showed that there has been a steady increase in spam emails. This means it will be time-consuming for an email user to always delete junk emails. Hence, we realize that there is a need of tools that can be used to automatically detect or classify spam emails.

Past studies have utilized different machine learning techniques to tackle the problem. An approach that was mostly used in many studies is known as binary/two-class classification. This approach requires two classes, the negative and the positive class, for training a model. One of the reasons two-class classification was applied in most cases is because it is found to outperform one-class classification approach in some previous

studies (Wu et al. 2005). Furthermore, it was indicated in (Manevitz & Yousef 2002) that use of positive samples only for training a model might not have good performance as compared to when both negative and positive examples are present.

Analyses of a Spambase dataset, one of the bench-marking datasets for assessing the reliability of classification models, was considered for analysis by Kumar et al. (2012) using Tanagra. Tanagra is a tool used in data mining for learning about productive email spam classifiers. Important features were extracted using feature extraction approaches such as Fisher filtering, Stepwise Discriminative Analysis (STEPDISC) (Costanza & Afifi 1979) and ReliefF.

A supervised feature reduction technique called the Fisher filtering, which makes use of Fisher's ANOVA ranking to determine features which lead to high degree of separability in a classification problem was employed and led to an acceptable classification performance (Kumar et al. 2012, Desai et al. 2016).

Beside the Fisher filtering technique that led to the best performance, STEPDISC filtering technique is one of the feature reduction techniques that were employed for a further comparison in (Kumar et al. 2012). These techniques, STEPDISC and Fisher filtering, played a significant part before classification phase. Because usually when there are more feature variables compared to the training instances, classifiers misclassify many data points.

Feature representations that were generated by Kumar et al. (2012) using varying feature reduction techniques on the Spambase dataset from UCI Machine Learning Repository, were used to asses various classification algorithms to delineate legitimate emails from spam emails. 15 classifiers were evaluated by applying them on the generated feature representations. The efficiency of each classifier was evaluated based on the error rate, recall, and precision. Classification algorithms that were evaluated include:

- C4.5
- C-RT & CS-CRT
- ID3
- k-NN
- LDA
- Log Regression TRIRLS
- Multilayer Perceptron (MLP)

- PLS-DA & PLS-LDA
- Random Forest (RF)
- SVM

With LDA (Linear Discriminant Analysis) given at least two class categories, the main objective is to find linear transformation of the features which delineate instances of varying class categories. For pattern recognition, LDA has mostly been employed for data classification and feature reduction. In contrast to PCA, for feature reduction LDA takes class label into account to determine the linear transformation of features which are more significant for understanding patterns of the data. In a classification problem, the main objective of LDA is to learn function f which determines the linear transformation of $\mathbf{x}^{(i)} \in X$ where $\mathbf{x}^{(i)} \in \mathbb{R}^d$ into the corresponding label $y^{(i)} \in \mathbf{y}$.

Unlike LDA that is based on an assumption that all data representations follow a normal distribution, SVM is not based on any probability distribution assumption. SVM is a binary classification technique that has been applied to various linear tasks.

The main objective with the SVM technique is to generate a decision hyperplane:

$$f(\mathbf{x}) = sign(\omega^T \mathbf{x}^{(i)} + b), \tag{2.29}$$

that will be able to classify an unknown instance $\mathbf{x}^{(i)} \in \mathbb{R}^d$. The ω can be expressed as:

$$\omega = \sum_{i=1}^{n} \alpha^{(i)} y^{(i)} \mathbf{x}^{(i)}. \tag{2.30}$$

During the training process the main objective is to learn $\omega \in \mathbb{R}^d$ and $b \in \mathbb{R}$ parameters.

SVM is not only limited to solving linear problems but is also able to solve non-linear problems. For non-linear data, it makes use of the kernel functions to transform the data into the high space hoping that the data will be linearly separable in that space. Let K be a kernel function being employed, then the learning function f will be expressed as:

$$f(\mathbf{x}) = \sum_{i=1}^{M} \alpha^{(i)} K(\mathbf{x}, \mathbf{x}^{(i)}) + b, \tag{2.31}$$

with M denoting the total count of support vectors.

A positively definite kernel function K has a feature map ϕ which transforms $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ such that

$$K(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \phi(\mathbf{x}^{(i)}) \cdot \phi(\mathbf{x}^{(j)}). \tag{2.32}$$

There are different kernel functions such as a Sigmoid, Polynomial, Radial Basis Function (RBF) and Linear. These mentioned kernel functions are given as:

- Sigmoid: $K(\mathbf{z}, \mathbf{x}) = \tanh(\gamma \cdot \mathbf{z}^T\mathbf{x} + \mathbf{x})$
- Polynomial: $K(\mathbf{z}, \mathbf{x}) = (\gamma \cdot \mathbf{z}^T\mathbf{x} + \mathbf{x})^d$
- RBF: $K(\mathbf{z}, \mathbf{x}) = \exp(-\frac{||\mathbf{z}-\mathbf{x}||^2}{2\sigma^2})$
- Linear: $K(\mathbf{z}, \mathbf{x}) = \mathbf{z}^T\mathbf{x}$

where γ, r, d and σ are kernel parameters that need to be well tuned. Optimization of kernel hyperparameters is a requirement when using kernel functions (Hsu et al. 2003). These processes, hyperparameter tuning and Optimal selection of a kernel, are some of the major challenges regarding SVM.

An ensemble method called Random Forest (RF) does not look for maximum-margin hyperplane like SVM, however, RF classifies data points using a large number of decision trees. Given $\mathbf{x}^{(i)} \in X$ where $\mathbf{x}^{(i)}$ is a d-dimensional vector, RF randomly picks samples from the original data, and construct multiple decision trees based on random selection of features. Due to a random selection of subsets and features for constructing multiple trees, this results in a random forest. When classifying an instance, each decision tree predicts the class of an instance. Because they might not all give the same output, then the majority vote strategy is used to decide the class label of that instance.

C4.5 in contrast to the RF, applies the concept of IG (*i.e.*, difference in entropy) to generate the decision tree. Normalized IG for each attribute a_j is generated, then an attribute with maximum IG value will then be chosen as a decision node that splits input data into sub-samples so that we end up being able to classify an instance into a correct class.

Suppose that a set X contains training instances $\mathbf{x}^{(i)}$, where every $\mathbf{x}^{(i)} \in X$ is a d dimensional vector. The corresponding class labels for each training instance are denoted by $y^{(i)} \in \mathbf{y}$, where $|\mathbf{y}| = C$ and C denoting the total count of the categories in the training data. Let X_s denotes a subset of X where $a = s$ and a denotes any arbitrary feature variable, then the information gain for a subset X_s will be expressed as:

$$IG = info(X) - \sum_{s \in A} \frac{|X_s|}{|X|} info(X_s) \tag{2.33}$$

where: A consists of all possible values of s and

$$info(X) = -\sum_{i=1}^{C} P(y^{(i)}, X) \log_2(P(y^{(i)}, X)), \tag{2.34}$$

where $P(y^{(i)}, X)$ is the fraction of instances belonging to a category $y^{(i)}$ in X. A C4.5 decision tree is advantageous over other decision trees like Classification and Regressing Trees (C-RT) because it is susceptible on outliers. However, C-RT is able to handle outliers (Singh & Gupta 2014).

C-RT is usually abbreviated as CART in most studies. CART is a decision tree which was firstly introduced by Breiman et al. (1984). this decision tree can be used on classification or regression tasks. CART construct the binary decision tree starting at a root note.

The root note split into branches leading to two child nodes. The child nodes also split into their child nodes which are grand children of the root note. This process continues until the split process is no longer possible. When a stopping criterion has been reached, the tree is then pruned by removing leaves which are least informative to the learning of the model.

Suppose we are given X consisting of $\mathbf{x}^{(i)}$'s and $\mathbf{x}^{(i)} \in \mathbb{R}^d$. Let $\mathbf{y}$ denotes a set of the corresponding labels for each $\mathbf{x}^{(i)} \in X$. $k > 0$ is the total count of the unique labels in $\mathbf{y}$. We denote the parent node as N_p. The left and right child are denoted as N_l and N_r respectively.

In every note N_t splition is done using impurity function $f(N_t)$. The most utilized impurity function is known as Gini index which is defined as

$$f(N_t) = \sum_{i=1}^{k} P(y^{(i)}|N_t) P(y^{(j)}|N_t), j \neq i \tag{2.35}$$

where $P(y^{(i)}|N_t)$ is the proportion of instances labeled as $y^{(i)}$ in X. $f(N_t)$ can also be defined as

$$f(N_t) = 1 - \sum_{i=1} k P^2(y^{(i)}|N_t) \tag{2.36}$$

Chi-Square Automatic Interaction Detection (CHAID) is one of the decision trees different to CART. CHAID which is abbreviated as CS-CRT in Tanagra is a chi-squared test based decision tree for classification tasks. It was initially introduced by Gordon V Kass in 1980. CHAID in contrast to CART that makes use of binary splits, it uses

multi-level splits. As a result, CHAID needs more training instances as compared to CART (Karthikeyani et al. 2012). With CHAID a note can be split into at least two branches (Pradhan 2013). Ture et al. (2009) presented more details regarding CHAID.

Some of the decision trees like RF have similar behavior as the non-parametric methods like k-Nearest Neighbors (k-NN) since during the testing phase they use the training instances for determining the class categories of the new instances. A pattern recognition technique known as k-NN can be used for classification tasks.

Consider a binary classification problem (*i.e.*, A case where we have class 1 and class 2 as labels of instances) and $k > 0$. k-NN identifies k data points that are nearest neighbors of a data point $\mathbf{x}^{(i)}$ from a training sample. Thus the class label of $\mathbf{x}^{(i)}$ will be class 1 if it has the majority votes as compared to class 2, otherwise it will be categorized into class 2. To avoid clashes it is important to ensure that $k \neq 2N$, with N denoting total count of classes. For a case where $k = 1$ an instance is given a label of the closest point.

The k-NN technique has been employed in various studies like (Pandey & Chakravarty 2010) for identifying unwanted text documents, where decision trees such as C4.5, ID3, and CART were employed for comparison. A decision tree C4.5 is a descendant of the Interactive Dichotomizer version 3 well known as ID3. Unlike CART which makes use of the Gini impurity for evaluating a node that well split the data, instead both ID3 and C4.5 utilize Information gain. ID3 make use of the top-down greedy search approach to construct a decision tree (Jin et al. 2009). For every instance to determine more significant attributes, Information Gain is employed. The Information Gain, in this case, plays an important role in determining the significance of each feature variable for determining the class category of a data instance.

According to the results reported in (Kumar et al. 2012), ID3 in most cases showed to have a poor performance compared to other classifiers such as CART, C4.5, k-NN, and RF. However, there were other classifiers like Partial Least Square-Discriminant Analysis (PLS-DA) that were outperformed by ID3 most of the time. This classifier, PLS-DA, delineates categories by determining the differences in the characteristics of the class categories.

Given the training instances $\mathbf{x}^{(i)} \in X$, and their corresponding class labels $y^{(i)} \in \mathbf{y}$, the main objective of PLS-DA is to find maximum covariance between X and $\mathbf{y}$ (Gromski et al. 2015). PLS-DA originates from a multivariate regression model known as partial least square regression (PLS-R) which generates the regression model between X and $\mathbf{y}$ (Brereton & Lloyd 2014). The regression model is determined by transforming X and $\mathbf{y}$ into the new spaces. PLS-R employs PCA to generate principal components of X. As a result, we then have X being decomposed using SVD.

PLS-DA differs with PLS-R because with PLS-R $\mathbf{y}$ is a set of continuous values while with PLS-DA $\mathbf{y}$ is a set of discrete variables which usually represent the groups or categories within X. Generally, the linear model is given by

$$\mathbf{y} = XB + \epsilon, \tag{2.37}$$

with the symbol B representing the corresponding matrix and the symbol ϵ regarded as the residual. The main objective is to generate the covariance matrix W of X so that T known as factor score matrix is defined as

$$T = XW \tag{2.38}$$

Equation 2.38 expresses the relationship between T and W (Zeng et al. 2007). As a result, X can be decomposed as

$$X = TP^T + \epsilon_x, \tag{2.39}$$

with P denoting a loading matrix of X. ϵ_x is the residual of X. Similarly $\mathbf{y}$ can be decomposed as

$$\mathbf{y} = TQ^T + \epsilon_y \tag{2.40}$$

where Q is the loading matrix of $\mathbf{y}$ and ϵ_y denoting $\mathbf{y}$ residual. Since $T = XW$, and $X = TP^T + \epsilon_x$ then we have B defined as

$$B = WQ^T \tag{2.41}$$

since $\mathbf{y} = XB + \epsilon_y$, then we have

$$\mathbf{y} = XWQ^T + \epsilon_y \tag{2.42}$$

Another classifier that make use of the PLS, Partial least square - linear discriminant analysis (PLS-LDA), is a classification technique which combines partial least squares (PLS) and linear discriminant analysis models. This is because PLS plays an important role on LDA by doing feature reduction using PLS feature transformation (Beleites et al. 2013).

Given $X \in \mathbb{R}^{n \times d}$, with n representing total count of the instances in X and the value of d representing the total count of the feature variables, We denote a set containing the corresponding labels of the instances in X by a symbol $\mathbf{y}$. PLS is iteratively being employed so that the maximum covariance in the X space and the $\mathbf{y}$ space is modeled to strengthen the relations of X and $\mathbf{y}$ (Karthikeyani et al. 2012).

Determining the best classifier, Kumar et al. (2012) investigated the feature selection algorithm that leads a classifier to a high accuracy for most classifiers and as well the classifier with the highest accuracy compared to all other classifiers. From the given results it is found that RF performs better than all classifiers when using fisher filtering for feature selection.

In data mining, a fisher filtering technique is known as a supervised algorithm used for selecting important features required to train classifiers. Fisher filtering does not take repetition of input features into consideration, however, the selection of feature does not depend on the classifiers (Nancy & Ramani 2011).

This study (Kumar et al. 2012) uses a data with a lot of limitations. The data is already in the pre-processed format making assumptions that every email document should have those features. Furthermore, features generated are frequency based. In addition to that, some of the words are used as informative terms for identifying spam emails. With the noise and slangs being introduced by spammers, it will make it difficult for the approach proposed by Kumar et al. (2012) to identify spam emails.

In contrast to this study (Kumar et al. 2012) in which the pre-processed data was used for assessing learning classifiers, Bo Yu (2008) used a raw data that was in the text format to generate feature representation from it through these processes, feature selection and feature extraction. In addition, the BoW technique was employed for feature extraction. The performances of the following four machine learning algorithms:

- Naive Bayes (NB)
- Neural Networks (NN)
- Relevance Vector Machine (RVM)
- SVM

were assessed based on the generated features.

Unlike, SVM which tries to find a hyperplane with maximum margin for delineating negative data examples from positive data examples, Bayesian classifiers use probability

distribution to classify instances. Consider $y^{(i)} \in \mathbf{y}$ with $\mathbf{y}$ symbolizing a set consisting of the class labels. Let $\mathbf{x}^{(i)} \in X$ be a training instance in X with n denoting the total number of instances. We suppose that each training instance consists of d features. Then the probability $P(y^{(i)}|\mathbf{x}^{(i)})$ is given by

$$P(y^{(i)}|\mathbf{x}^{(i)}) = \frac{P(\mathbf{x}^{(i)}|y^{(i)})P(y^{(i)})}{P(\mathbf{x}^{(i)})} \tag{2.43}$$

Thus $\mathbf{x}^{(i)}$ belongs to the class with maximum probability. NB assumes that all features are independent of each other. Meaning that each feature has its own contribution in determining to which class category an instance belongs. With $x_j^{(i)}$ denoting feature value of $\mathbf{x}^{(i)}$, the probability $P(\mathbf{x}^{(i)}|y^{(i)})$ is generated by $P(x_j^{(i)}|y^{(i)})$ (Sahami et al. 1998).

RVM in contrast to SVM employs the Bayesian approach for implementing probabilistic classifier. Given a training set X containing $\mathbf{x}^{(i)}$'s instances and their corresponding $y^{(i)} \in \{-1, +1\}$ labels of categories. With SVM the main aim is to find a good generalization by maximizing the hyperplane margin while minimizing the training error (Tipping 2003). Some of the limitations regarding SVM approach is its requirements for tuning the regularization parameter C which in most cases requires the cross-validation technique. This is computationally expensive. Furthermore, the number of support vectors increase with the number of training instances (Tzikas et al. 2006).

Tipping (2003) introduced RVM to overcome some of the limitations regarding SVM. RVM revolve around the assumption that the conditional probability $P(y^{(i)}|\mathbf{x})$ is equivalent to the Gaussian $N(y^{(i)}|f(\mathbf{x}), \sigma^2)$, where σ^2 is a variance. As a result, we have the likelihood of dataset given by

$$P(y^{(i)}|\omega, \sigma^2) = (2\pi\sigma^2)^{\frac{n}{2}} \exp\left(-\frac{1}{2\sigma^2||y^{(i)} - \mathbf{\Phi}\omega||^2}\right) \tag{2.44}$$

where $\mathbf{\Phi} = K(\mathbf{x}, \mathbf{x}^{(i)})$.

Similarly to RVM and SVM, the Artificial Neural Network (ANN) model well known as the Neural Network (NN) is also a supervised technique, however, NN interconnects artificial neurons. The way in which ANN solves problems is inspired by the human brain. The main goal of this model, NN, is not to implement human brain but to build a model that can solve many tasks in various studies such as speech recognition, anomaly detection, image processing, signal processing, pattern recognition and natural language processing.

One of the advantages regarding the ANN is its capability to adjust and become adaptive with respect to the information being processed. Weight vectors interconnecting neurons

are the ones which are being adjusted to make the ANN be adaptive. An example of the ANN model may be made of three layers namely:

- input layer;
- hidden layer;
- output layer.

However, ANN model can be more complex adding more multiple layers. Each layer has at least one neuron.

The basic ANN has one neuron, such kind of the model is known as perceptron which consist of at least one input, one output and a single processor. For a perceptron, given input $\mathbf{x} \in \mathbb{R}^d$ and an interconnection weight $\omega \in \mathbb{R}^d$, then the output $h(\mathbf{x})$ is given by

$$h(\mathbf{x}) = \sum_{j=1}^{d} \omega_j x_j + b \tag{2.45}$$

where b is known as the bias. Hence, we have $h(\mathbf{x})$ represented by

$$h(\mathbf{x}) = \omega^T \mathbf{x} \tag{2.46}$$

This is because $\mathbf{x}$ is always augmented by $x_0 = +1$, thus we have

$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$$

and ω given by

$$\omega = \begin{bmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \\ \vdots \\ \omega_d \end{bmatrix}$$

where ω_0 is the intercept. Equation 2.46 can be used as the decision hyperplane for classifying an instance as a negative data point or a positive data point by defining a threshold function (Alpaydin 2014). There are various types of ANNs which include the Recurrent Neural Network (Mikolov et al. 2010), Convolutional Neural Network (Ciresan et al. 2011), and Feed-forward Neural Network (Bebis & Georgiopoulos 1994).

MLP is a Feed-forward ANN consisting of multiple layers. The basic MLP consists of a single hidden layer. From each layer there are neurons which takes in the output from the previous layer as input and generate the output using an activation function. Backpropagation algorithm is used to train MLP.

Given a finite set $\{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(n)}, y^{(n)})\}$ containing n instances with their corresponding categories. The main objective is to minimize the cost function $J(\omega, b, \mathbf{x}, \mathbf{y})$ for a training instance $(\mathbf{x}, \mathbf{y})$ which is given by

$$J(\omega, b, \mathbf{x}, \mathbf{y}) = \frac{||h(\mathbf{x}) - \mathbf{y}||^2}{2}, \tag{2.47}$$

where $h(\mathbf{x})$ described as

$$h(\mathbf{x}) = \hat{f}(\omega^T \mathbf{x}) \tag{2.48}$$

is the hypothesis with parameters ω and b. A function $\hat{f}$ in equation 2.48 is known as the activation function. There is a plenty of them, for example, we may have $\hat{f}$ described as

$$\hat{f}(\mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{x})}. \tag{2.49}$$

This kind of the activation function in equation 2.49 is called the sigmoid.

These weight parameters, ω and b, enable $h(\mathbf{x})$ to fit the data. They need to be initialized in every layer before the learning process. In each layer, ω and b, are described as ω_{ij}^l and b_i^l subsequently. They are initiated with respect to $Normal(0, \epsilon^2)$ for a very small positive value of ϵ. ω_{ij}^l is an interconnection weight between the j^{th} neuron in the l^{th} layer and the i^{th} neuron in $(l-1)$ layer, with $l = 1, 2, \ldots, N+1$. The value of $N+1$ denotes the total count of layers in the MLP. Usually stochastic gradient descent algorithm is employed for updating learning parameters.

Bo Yu (2008) reported that NN produce a poor performance compared to other classifiers like NB, RVM, and SVM. However, the study fail to clearly show how feature

selection or feature reduction was conducted. Additionally, it is not clear which technique was used for feature reduction or feature selection. These processes if are not well conducted most of the time lead to a poor performance depending on the complexity of the dataset. Regarding SVM the study indicated that as the total count of training instances increases, required number of support vectors also increases. As a result, basis functions are used for no good reason. Additionally, cross validation in most cases when SVM is being trained is required which is computationally expensive.

However, Webb et al. (2005) showed that even though RVM requires fewer vectors as compared to SVM, it takes a long time for training. This was observed when a Corpora with the size of more than 300000 features in (Webb et al. 2005) was utilized for assessing the classifiers performance. The study further showed that Classifiers like SVM performs poorly when such kind of data with a huge number of features is being employed. This happens when some of the irrelevant features are used for feature representation. Webb et al. (2005) solved the problem by utilizing IG to select important features that can be used to train classifiers. Both SVM and RVM always outperformed NB in (Webb et al. 2005).

Amazingly, Song et al. (2009) showed that NB with Inverse Document Frequency (IDF) used for feature selection outperforms SVM. However, they have very close results. These results are based on data collected from Hotmail. One of the things that might have made SVM to not outperform NB, in this case, is the selection of the optimal kernel function and optimization of kernel hyper-parameters. This is because it is not clear how SVM was trained in terms of the hyper-parameters. In addition to that, NLP problems can be complex to an extent that it becomes difficult for non-parametric learning models like SVM to grasp important information. Furthermore, the performance of SVM can be influenced by how features were generated. In most cases more feature engineering will be required for a good performance.

Mi et al. (2015) introduced Stacked Autoencoder for classifying an email as either spam or non-spam to overcome some of the issues that may be encountered with non-parametric techniques like SVM. Stacked Autoencoder is regarded as a deep learning technique. One of the advantages of the deep learning models is their capability of handling more complex datasets from various fields of studies like speech recognition and Image processing. Unlike non-parametric techniques, deep learning techniques need a less feature engineering.

Both Autoencoder and Stacked Autoencoder have been employed for feature reduction in computer vision problems. However, for feature extraction and feature reduction, Mi et al. (2015) used BoW and IG respectively. The proposed approach was compared to other traditional methods. Based on the reported results in (Mi et al. 2015) Stacked

Autoncoder for spam email identification outperformed all traditional classifiers that were taken into consideration for comparison. The major limitation of this study is in how features were generated. BoW and IG techniques which encounter a lot of limitations for NLP problems were used.

2.6 Autoencoder

Autoencoder is an unsupervised learning algorithm that compresses a huge feature space into the corresponding low feature space. Autoencoder transforms data to a lower feature space in a way that we can reconstruct into the original input data. Because it is unsupervised it requires unlabeled data to learn the model. A simple Autoencoder is made up of the three consecutive layers, an input layer, hidden layer and output layer.

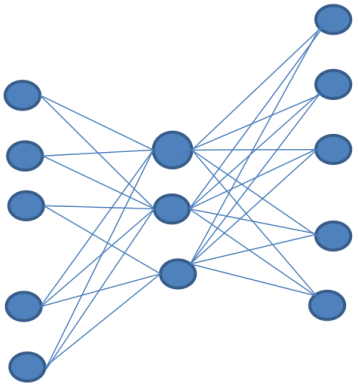


FIGURE 2.1: An example of the Autoencoder with a single hidden layer.

An example of the Autoencoder is presented in Figure 2.1. Starting from the left-hand side is the input layer, to the right-hand side the input layer is followed by the hidden layer, then lastly we have the output layer. These layers, input, hidden and output they subsequently follow each other. Each layer consists of neurons. The left half is known as encoding part and the other half is known as decoding part. The middle layer is known as the bottleneck.

Given a set of unlabeled training instances $\mathbf{x}^{(i)} \in X$, where $\mathbf{x}^{(i)} \in \mathbb{R}^d$, and $d, \hat{d} \in \mathbb{Z}^+$. We suppose that $d > \hat{d}$. Autoencoder uses the encoding part to transform $\mathbf{x}^{(i)}$ into $Z^{(i)} \in \mathbb{R}^{\hat{d}}$. However, the decoding part transforms $Z^{(i)}$ into $\tilde{\mathbf{x}}^{(i)} \in \mathbb{R}^d$, where $\mathbf{x}^{(i)} \approx \tilde{\mathbf{x}}^{(i)}$. As a result, we have

$$Z^{(i)} = \omega^{(1)}\mathbf{x}^{(i)} + b^{(1)} \tag{2.50}$$

and

$$\tilde{\mathbf{x}}^{(i)} = \omega^{(2)} Z^{(i)} + b^{(2)}. \tag{2.51}$$

During the training process of the Autoencoder the main objective is to minimize function J given by

$$J(b^{(1)}, b^{(2)}, \omega^{(1)}, \omega^{(2)}) = \sum_{i=1}^{n} (\tilde{\mathbf{x}}^{(i)} - \mathbf{x}^{(i)})^2 \tag{2.52}$$

by making use of stochastic gradient descent algorithm.

2.7 Stacked Autoecoder

Stacked Autoencoder (SAE) is regarded as a deep network. In contrast to the basic Autoencoder, SAE consists of multiple layers of the Autoencoders. Figure 2.2 is an example of the SAE architect. With the SAE the input data is encoded through successive layers in the encoding part of the SAE through the middle layer known as a bottleneck. From there it is decoded again through successive layers in the decoding part of the SAE. The encoding part of the SAE is executed by executing each layer in the encoding part.

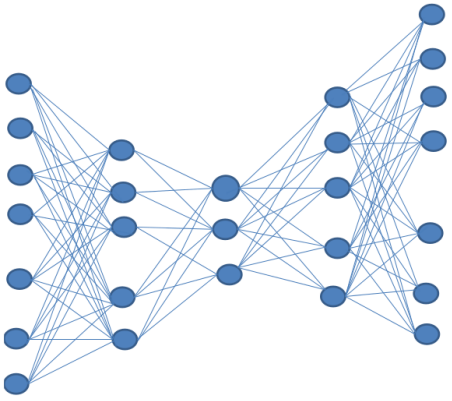


FIGURE 2.2: An example of the Stacked Autoencoder with multiple hidden layers.

2.8 Summary

From previous studies we realize that there are various techniques that can be used for email spam classification. We also noticed that classification algorithms categorize an instance based on knowledge acquired from other instances used for training. Training of a learning algorithm requires attribute/feature values of the instances selected for training, as well as their predefined category/class labels. Furthermore, feature representation also plays a huge role towards the performance of the classifiers. Frequency

based features have been employed mostly for identifying spam emails. However, one of the major challenges is dealing with the curse of dimensionality issue.

Chapter 3

Word Count Based Feature Construction

3.1 Introduction

BoW and TF-IDF are most commonly used feature representation methods in spam email classification (Mi et al. 2015). However, both feature representation methods suffer from the curse of dimensionality. Although as alternative to BoW and TF-IDF, several feature extraction methods have been proposed like term space partition based ensemble feature construction (ETSP) (Mi et al. 2016), we hypothesize that BoW and TF-IDF can still be efficient feature representation methods if efficient feature selection methods, such as IG or χ^2 were applied on them to mitigate the curse of dimensionality.

Decision Trees (DTs), such as C4.5 and Random Forest (RF), and Support Vector Machines (SVMs) have shown promising results in spam detection. Although C4.5 and RF are non-parametric classifiers, the performance of SVMs is highly dependent on the selected kernel function and its hyperparameters. Further, feature representation and selection plays an important role for the performances of both DTs and SVMs. This chapter investigates the impact of feature representation and selection techniques on the classifiers' performances. This chapter also investigates the effect of different kernel functions and their parameters on SVMs' performance in spam detection.

The research work regarding feature extraction and feature selection processes is still a work in progress in the natural language processing (NLP) domain. Usually, words which make feature representations to be more sparse and found to be not informative are removed before training classifiers by using feature reduction methods (Kumar et al. 2012).

Many studies which work on problems related to NLP classification tasks make use of hybrid approaches (Uğuz 2011, Günal 2012, Uysal & Gunal 2012) which tries to overcome the issue of high dimensionality so as to improve performance accuracy and processing time. Term Strength (TS) (Yang 1995) based techniques such as IG, χ^2, Mutual Information (MI) (Liu et al. 2009) and Document Frequency (DF) (Yang & Pedersen 1997) have mostly been used to evaluate the importance of the features. As a result, features which are regarded as of high importance are then used to train machine learning techniques for identifying non-spam emails from spam emails.

Based on the results reported by previous studies it was clear that feature selection techniques play a phenomenal part before the execution of the classifiers. The introduction of the noisy words by the spammers resulted in the TS based feature selection techniques failing to retain more robust features in many cases.

The TS based feature selection techniques are usually employed on various feature representations that are word count based. These feature extraction techniques, TF-IDF and BoW, refrain from taking into account the grammar and word sequence. TF-IDF technique in most cases still generates feature representation that has the meaning to classification algorithms but suffers from sparsity with the growing vocabulary (Zhang et al. 2011, Amayri & Bouguila 2010, Sebastiani 2002).

Most feature representation techniques have no guarantee regarding linear separability. That may lead to a poor generalization for linear SVM. Hence, SVM employs kernel functions for feature representations that are not linearly separable. This procedure is known to be kernel trick. The aim of the kernel trick procedure is to transform the existing feature space into the high dimensional feature space hoping for linear separability in the new space. Most of the time the kernel functions being used are distance based.

Amayri & Bouguila (2010) made a comparison of String kernels and mostly used kernel functions which are known to be distance based like Radial Basis Function (RBF), Sigmoid and Polynomial. However, Amayri & Bouguila (2010) failed to clearly show how variation of feature sizes affected SVM performance when distance based kernel functions are employed instead of String Kernel functions. This shows some of the limitations encountered in the study because sometimes feature size considered for training a learning classifier may affect the performance of the classification technique.

In addition, this Chapter investigates the optimal hyperparameters and optimal selection of a kernel function for SVM to identify spam emails from non-spam emails. Different

feature extraction and feature selection techniques are employed to improve computational time and performance accuracy. The investigation of the optimal hyperparameters for each feature representation is taken into consideration.

Hyperparameters include kernel parameters and a regularization parameter usually denoted by a constant C. The parameter C determines the margin size between positive data points and negative data points. The flexibility of the hyperplane is controlled by the kernel parameters. Optimal hyperparameters ensures that we always have the decision boundary which gives good generalization, as a result, a learning classifier easily adapts to the new datasets.

This chapter utilizes spam email filtering methods which have been well studied for email spam classification tasks. During the preprocessing phase, feature selection methods evaluate scores to all feature variables. More distinctive features are given a high score and a low score is given to features which are less distinctive. For each feature representation method, the optimal kernel parameters, regularization parameter and an optimal kernel function for the SVM are evaluated. As a result, our proposed approach manage to identify spam emails with a higher value of F-Score and high value of accuracy compared to the approach proposed by Mi et al. (2015) in which binary features were utilized for feature representation using BoW and IG.

3.2 Methodology

In this section, a spam filtering technique is proposed making use of the features generated using the co-occurrence based methods. For feature reduction, techniques being used evaluate the feature relevance and assign scores to the feature variables. Various traditional feature representations are employed to investigate the optimal hyperparameters and kernel functions for each of them.

The hyperparameter optimization process is conducted using a grid search algorithm (Hsu et al. 2003). 10-fold cross validation (CV) (Kohavi et al. 1995) is being employed for performance evaluation when tuning the kernel parameters and the regularization parameter. Linear kernel, Sigmoid kernel, and RBF kernel functions are employed. Since the features extracted from datasets are all normalized, all Kernel parameters fall into a range which is further discritized in grid search algorithm.

The Linear Kernel requires optimization of the regularization parameter C. Values of constant C being considered are 10^M where $M = 0, 1, 2, 3$. The RBF requires adjustments on λ and C, with λ given as $[0.1, 0.01, 0.001, 0.0001]$. For the Sigmoid kernel function, same parameter tuning as with RBF kernel is required. Due to the execution

time that the Polynomial kernel function was taking, we could not consider the polynomial kernel function for performance evaluation. Some of the challenges we face with grid search algorithm include its execution time. Grid search algorithm prolongs the computational time to process the training data.

For performance evaluation, various performance measures are taken into account for comparison. This Chapter makes use of the True Negatives (t_n), False Negatives (f_n), False Positives (f_p), and True Positives (t_p). Note that t_n and t_p are the total counts of the emails well classified, while f_p and f_n are the misclassified email documents. Weka (Garner et al. 1995) the data mining tool is employed for implementing learning models and also for performance evaluation. F-measure, Precision, Recall, and Accuracy measures are determined as follow:

$$F - measure(Fm) = \frac{2t_p}{2t_p + f_n + f_p} \tag{3.1}$$

$$Recall(ReC) = \frac{t_p}{t_p + f_n} \tag{3.2}$$

$$Precision(PrC) = \frac{t_p}{t_p + f_p} \tag{3.3}$$

$$Accuracy(ACC) = \frac{t_p + t_n}{t_p + f_n + f_p + t_n} \tag{3.4}$$

3.3 Experimental Setup

The experimental design of this chapter is presented in Figure 3.1. The proposed approach is evaluated on the Enron dataset (Metsis et al. 2006). The preprocessing phase is firstly employed to ensure that the raw data is in the right format before the feature extraction process. The preprocessing phase involves removal of all null characters, the removal of the stopwords, conversion of all words to their root format (*i.e.,* stemming) and conversion of all characters to their lower case.

For feature representation there are four approaches which are considered. These approaches involve feature extraction techniques and reduction techniques. The feature extraction techniques taken into consideration are BoW and TF-IDF. The feature reduction approaches which are employed are χ^2 and IG. As a result, we have the following feature representation approaches:

- BoW + IG
- BoW + χ^2
- TF-IDF + IG
- TF-IDF + χ^2

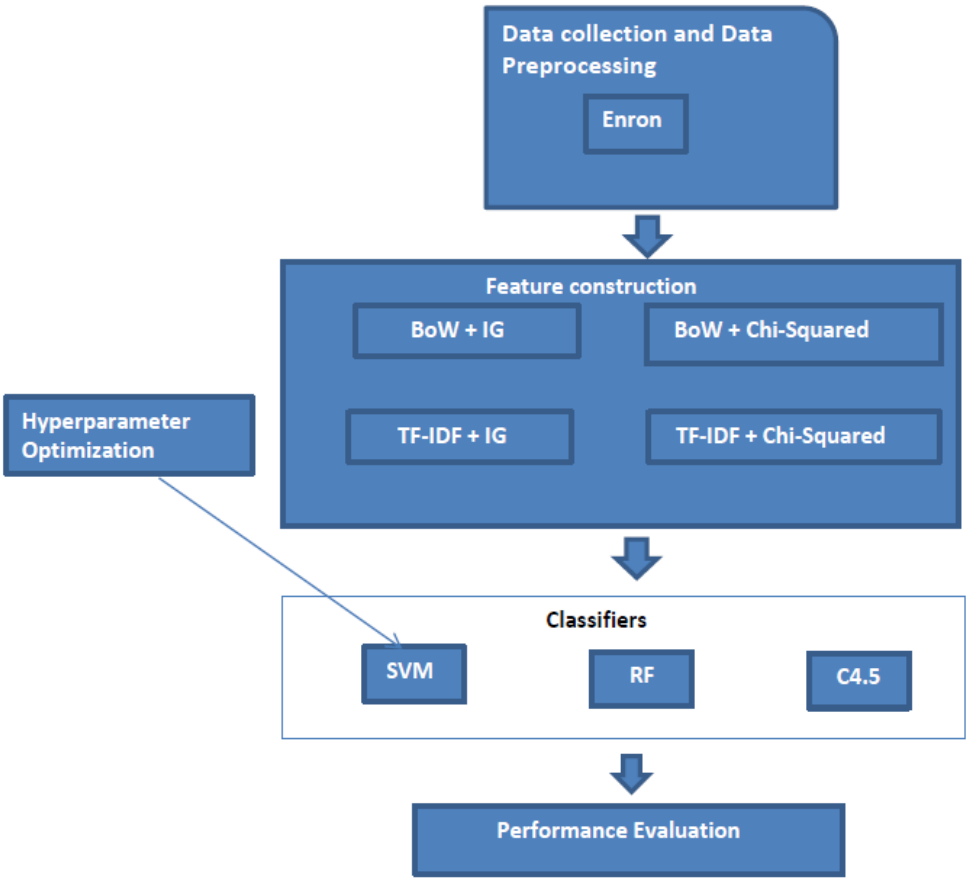


FIGURE 3.1: Feature Selection and Hyperparameter Optimization

For BoW, both frequency feature values and binary feature values are taken into consideration. For SVM, this chapter further considers hyperparameter optimization for each feature representation approach. k-fold CV technique is employed for implementing the classifiers. The performance evaluation is done based on Precision, Recall, and F-Score.

This proposed approach is compared to some of the recent studies such as (Mi et al. 2015, 2016) that have reported the performances of classifiers which include SVM on Enron dataset. However, there are some limitations which were encountered in these studies regarding the performance of classifiers on features generated using BoW. This study takes into account the optimal selection of features and hyperparameter optimization.

In addition, this study further shows that the traditional classifiers can still perform better compared to the results which were reported in (Mi et al. 2015, 2016). For instance, Mi et al. (2016) introduced term space partition based ensemble feature construction (ETSP) and showed that it is more robust for feature construction as compared to BoW. Table 3.1 shows results which were reported by Mi et al. (2016) and a significant improvement on SVM performance for delineating spam emails from non-spam emails was reported.

TABLE 3.1: SVM performance on Enron dataset reported in (Mi et al. 2016)

	PrC (%)	ReC (%)	Fm (%)	ACC (%)
BoW	90.88	98.87	94.62	95.13
ETSP	94.97	98.35	96.57	97.32

3.4 Results and Analysis

Table 3.2 presents the performance results for C4.5, SVM, and RF which were reported by Mi et al. (2015). A 6-fold CV was utilized for training the learning classifiers. For feature representation IG and BoW were used for generating the binary representation. With the binary feature representation, whenever the feature variable occurs within a document its value is denoted by "1" otherwise "0".

TABLE 3.2: The performance of the classifiers on the feature representation generated using IG and BoW with binary values on the Enron dataset reported in (Mi et al. 2015)

	PrC (%)	ReC (%)	Fm (%)	ACC (%)
SVM	89.64	98.74	93.86	94.63
C4.5	82.88	97.07	89.02	90.33
RF	91.46	99.28	96.06	95.11

Using all the tokens in the Eron data for generating the feature space, result in a case where we have more than 120000 features. Some of the tokens are encountered due to the noise issue in the text processing problems. As a result, we face an issue regarding the processing time of a learning classifier. Hence, it is important to keep a certain limit of attributes for constructing the feature space. We decided to keep $\tilde{T}$ top terms for generating the feature space to avoid memory error issues. $\tilde{T}$ was assigned the value of 1500. Those top $\tilde{T}$ terms are the most common terms. However, the number of the most common terms is not limited to the value of $\tilde{T}$ since we might have some ties. Hence, we end up with the feature space consisting of 2119 attributes.

The results from Table 3.2 are compared with the results presented in Table 3.3. In Table 3.3 similarly to Table 3.2 BoW approach was used for feature extraction and IG for feature selection. However, in Table 3.3 instead of binary representation, word

count values are used for feature representation. The number of the features used for training the classifiers in Table 3.3 was 400. The value of 400 was chosen because was leading to the best performance in terms of both ACC and Fm scores. This was done by investigating the number of features between 50 and 400. The number of features were varied with 50.

Furthermore, in Table 3.2 and Table 3.3 for evaluating classification performance 6-fold CV was used. Based on the feature representation employed, RBF kernel was found to be optimal for training SVM. $\lambda = 0.1$ and $C = 10$ were found to produce the best performance for SVM based on the given interval of parameters. All classifiers in Table 3.3 have shown a significant improvement compared to the classifiers performances published in (Mi et al. 2015) with respect to Accuracy and F-Score.

TABLE 3.3: The performance of the classifiers on the feature representation generated using IG and BoW with the term count values on the Enron dataset.

	PrC (%)	ReC (%)	Fm (%)	ACC (%)
SVM	97.90	97.90	97.90	97.89
C4.5	95.70	95.70	95.70	95.66
RF	98.00	97.90	97.90	97.94

For Table 3.4 in contrast to Table 3.3, binary features were employed. Table 3.4 presents results of classifiers performance on the same approach which was employed in Table 3.2. However, in Table 3.4 200 features were used to train all the classifiers. 200 features were used because after the investigation of the optimal feature size we found out that the performance of the classifiers improved with the increase of the feature size. However, feature sizes between 200 and 400 didn't lead classifiers to a significant improvement. Hence, we opted to choose 200 features for training learning classifiers for a faster execution of learning models. Looking at the Fm score from Table 3.4, the classifiers have a significantly better performance compared to the classifiers in Table 3.2 where more features were employed for training classifiers. RBF kernel was found to lead SVM to a better performance when the parameter C is set to 10 and λ set to 0.1.

TABLE 3.4: The performance of the classifiers on the feature representation generated using IG and BoW with binary values for feature representation on Enron dataset based on the approach proposed in this Chapter.

	PrC (%)	ReC (%)	Fm (%)	ACC (%)
SVM	97.40	97.40	97.40	97.37
C4.5	95.70	95.70	95.70	95.69
RF	97.30	97.30	97.30	97.32

The results presented in Figure 3.2 and Figure 3.3 based on the use of the different feature selection and feature extraction techniques. They also cover various classification methods such as C4.5,RF, and SVM. For performance evaluation 10-fold CV is being

used. The results presented in Figure 3.2 explains the performance of all classifiers being evaluated based on the use of IG and BoW for feature representation. The increase in the total count of the features resulted in an improved classification performance. $\lambda = 0.1$ and $C = 10$ are found to be the optimal parameter values for the RBF kernel.

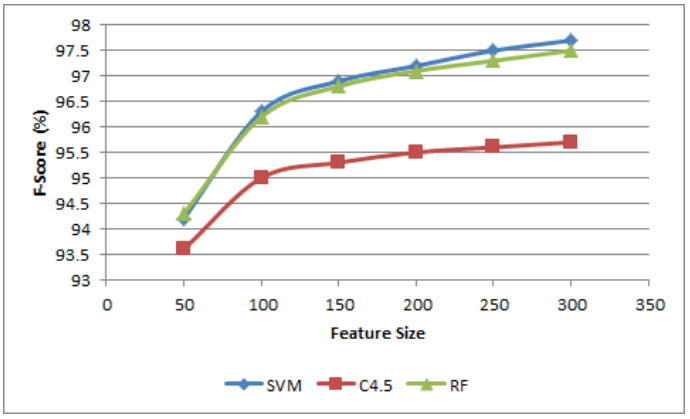


FIGURE 3.2: The performance of the classifiers on the feature representation generated using IG and BoW with different feature sizes on Enron dataset.

Figure 3.3 presents the performance of the classifiers on the feature representations constructed using IG with TF-IDF. In contrast to when the BoW technique is used for feature extraction where RBF was found optimal, now SVM performs well with linear kernel and led to an improved classification accuracy. The SVM with linear kernel was found to achieve the best performance when $C = 1$. The performance is found to slightly degrade with the increasing value of C.

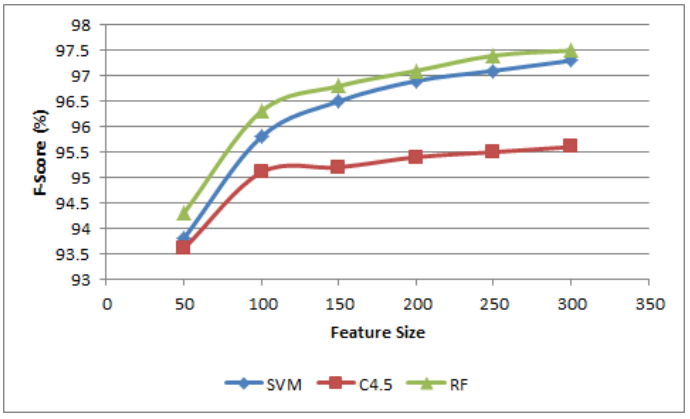


FIGURE 3.3: The performance of the classifiers on the feature representation generated using IG and TF-IDF with different feature sizes on Enron dataset.

In Figure 3.4 BoW was used for feature extraction while χ^2 technique is being used for feature selection. Same kernel functions are being used both in Figure 3.2 and Figure 3.4. The variation of feature selection techniques led to a slight performance difference for classification. This is due to the optimal selection of a kernel function for each feature representation.

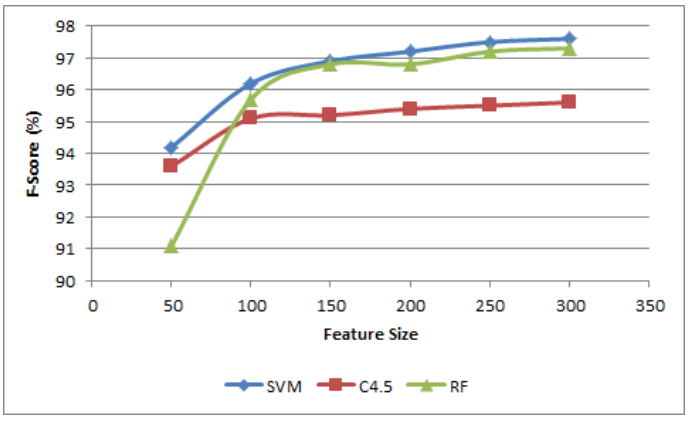


FIGURE 3.4: The performance of the classifiers on the feature representation generated using χ^2 and BoW with different feature sizes on Enron dataset.

For the features generated using the BoW approach, SVM performs well with the RBF kernel. Even if the feature selection technique is different, the SVM classifier is still found to perform well with the RBF kernel function.

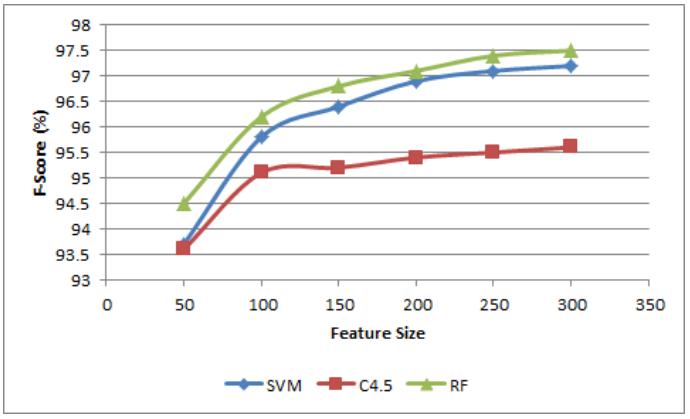


FIGURE 3.5: The performance of the classifiers on the feature representation generated using χ^2 and TF-IDF with different feature sizes on Enron dataset.

Based on the feature sizes considered in Figure 3.5, the classifiers performance improve with the growth of the feature size. The feature extraction technique employed is TF-IDF. The selection of informative features was done using the χ^2 technique. The SVM classifier performed better when the linear kernel function was employed. The parameter $C = 1$ was found to lead to a better performance. Comparing the C4.5, RF and SVM performances, the RF performed better in most cases. The classifier found with the worst performance is the C4.5. The C4.5 showed a significant improvement when the feature size was increased from 50 feature size to 100 feature size. There was an insignificant improvement on the C4.5 classifier when the feature size was increased from 100 feature size.

3.5 Conclusion

The performances of all the classifiers improve with the increase in the total count of the features. The lower feature space representation does not retain significant features, as a result, a learning classifier tends to perform poorly.

In most cases SVM performs better with IG and BoW used for generating feature representation by considering Fm Score and ACC.

BoW + IG feature representation led to an excellent performance compared to feature representation techniques proposed in (Mi et al. 2016). There are various limitations which have been encountered by previous studies, however, BoW with the optimal number of features may lead the classifiers to a better performance. For the kernel based methods, hyperparameter optimization is important to ensure a better performance in most cases. Looking at the outcomes reported in (Mi et al. 2015, 2016) about SVM performance based on the Enron dataset the proposed approach in this chapter led to a significantly improved performance, where IG and BoW are used for feature representation.

Chapter 4

Neural Network Based Feature Construction

4.1 Introduction

The majority of the spam classification literature such as in (Joachims 1996, Mi et al. 2015, Diale et al. 2016, Mi et al. 2016) utilizes term count based feature construction techniques to present meaningful semantics within a document. However, using term count based feature construction techniques such as TF-IDF and BoW we encounter various limitations which includes curse of dimensionality. To ensure a better performance, terms which are not common are discarded. In addition, the removal of less informative terms is taken into consideration by firstly identifying them using ranking methods such as χ^2 and IG as mentioned in Chapter 3. As a result, the generated feature representation loses informative terms for identifying spam emails. This chapter proposes to use a neural network based approach to generate a robust dense feature representation that will lead to an improved classification performance in a lower dimensional feature space.

Dimensionality reduction of the vector space is an important concept that increases the efficiency of a document representation. One of the issues in text categorization is dealing with large sparse matrices. An excessive number of the feature variables is a burden to a learning classifier. Because an excessive number of features may negatively affect the performance of a learning classifier. In addition, the computational time for processing the data during the training process may be prolonged. Therefore, a preprocessing stage that includes feature extraction and feature selection processes in the field of machine learning (ML) is a vital role for speeding up computation and improving classification accuracy.

In this study, unsupervised learning models named distributed bag of words (DBOW) and Distributed Memory (DM) (Le & Mikolov 2014) are utilized to learn a numerical vector of fixed-length for every email. These models capture word ordering and semantic meaning from a text document. Furthermore, the cosine similarity (CS) (Kumaran & Allan 2004) is used to generate a new feature space using CS measures of each email document with all email documents.

Autoencoder (Thompson et al. 2002) with a single hidden layer is then employed for feature reduction. This study clearly shows that Autoencoder can determine more robust features for identifying spam emails. The effectiveness of the proposed feature representation is examined by running several supervised classification methods on them.

The results in relation to the performance of classifiers indicate that the unsupervised learning technique we introduced leads to a better classification accuracy compared to the traditional feature extraction and feature selection methods for email spam identification. Stacked Autoencoder is also employed for feature reduction to compare with Autoencoder. Furthermore, kernel functions are also employed for feature transformation and compared with Cosine Similarity measure.

The problem considered in this study is related to data transformation, prior to machine learning classifiers. Feature extraction and feature selection processes are employed to transform raw data into a numeric representation that preserves class separability with the goal of identifying email documents as either spam or non-spam with the lowest dimensionality as discussed in Section 4.2. This study examines the performance of SVM, RF, and C4.5 for identifying an incoming email as spam or non-spam.

4.2 Distributed Memory and Distributed Bag of Words

DM and DBOW were introduced in (Le & Mikolov 2014) to learn vectors that represent documents of some sequential words. Similarly skip-gram and continuous bag of words were used to generate term/word vectors (Mikolov et al. 2013). More formally, given document vector $\mathbf{D}$ of sequential word vectors $\{\mathbf{t}_0, \mathbf{t}_1, \ldots, \mathbf{t}_{j-1}\} \in \mathbf{W}$, the core objective of DM is to predict next word $\mathbf{t}_j$ using all previous vectors in $\mathbf{W}$ together with $\mathbf{D}$. Stochastic gradient descent (Plagianakos & Magoulas 2013) is used to train both vectors and paragraph vectors. Distributed bag of words (DBOW) considers the case where the document/paragraph vector is given, then the model has to predict all words from the paragraph.

4.3 Cosine Similarity

Cosine similarity computes the angular distance between two document vectors. This measure can be used to determine the similarity of two given documents in terms of content. Given two feature vectors, say $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$, then CS measure is given by:

$$\cos\theta = \frac{\mathbf{x}^{(i)} \cdot \mathbf{x}^{(j)}}{||\mathbf{x}^{(i)}||||\mathbf{x}^{(j)}||}. \tag{4.1}$$

In this study for every email $\mathbf{x}^{(i)} \in X$ after getting its numeric representation using DM + DBOW, we calculate its CS measure against all emails in X. Those measures are utilized as features.

4.4 Experimental setup of Neural Network based approach for feature reduction

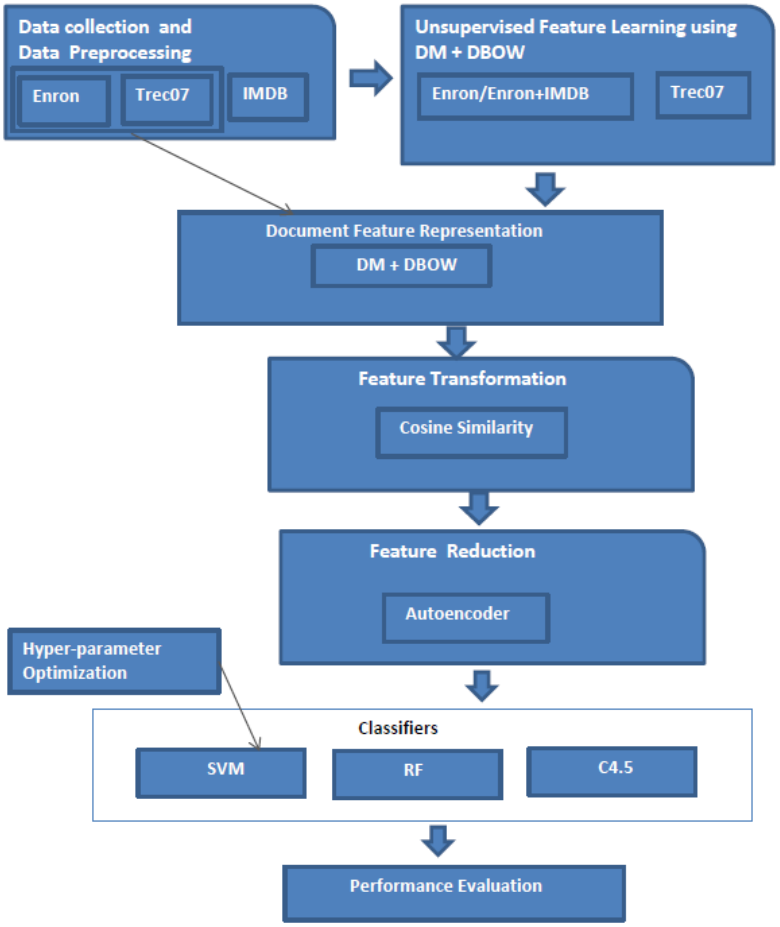


FIGURE 4.1: Autoencoder for Feature Reduction

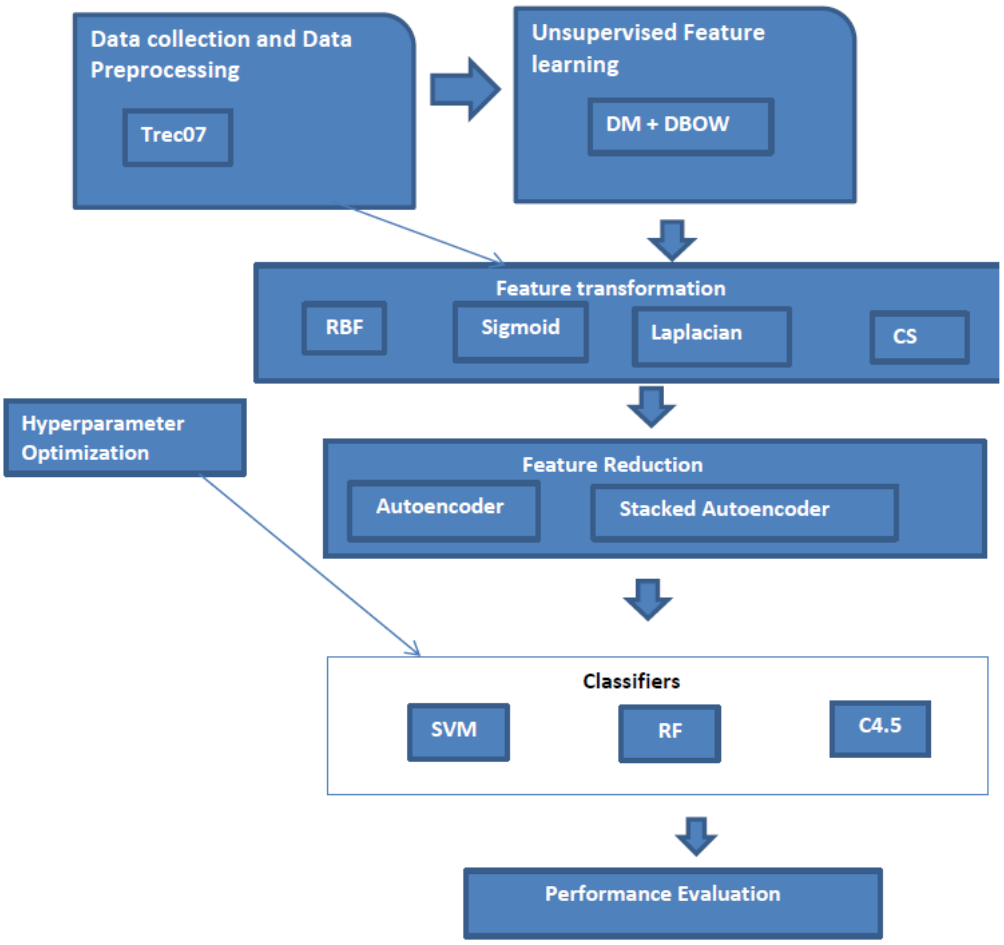


FIGURE 4.2: Autoencoder and Stacked Autoencoder for Feature Reduction

The entire process of identifying spam email from various recipients as proposed in this study uses seven main procedures namely: Data Collection and Data preprocessing, Unsupervised Feature Learning, Document Feature representation, Feature transformation, Feature Reduction, Hyperparameter Optimization, and classification. The entire process is shown explicitly in Figure 4.1.

The *Data preprocessing* phase extracts meta-information (e.g. message) from the email document. This preprocessing phase involves removal of punctuations and null characters, conversion of all characters and words to lower case.

For the *Unsupervised Feature Learning and Document Feature representation* DM + DBOW in an unsupervised manner generate and learn continuously distributed vectors that can represent meta-information of the preprocessed email files (Le & Mikolov 2014). The Enron dataset together with the unlabeled datasets from IMDB (Maas et al. 2011) are combined to learn DM + DBOW for feature construction. We only combine Enron and IMDB datasets whenever Enron dataset is being utilized to make a detailed inspection regarding the performance of the model we introduced. However, for Trec07

(Cormack 2007) data only 70000 unlabeled email messages are used to train DM + DBOW model.

During the *Feature transformation* phase, the CS measure is then employed to generate features which ensure that the information regarding the closeness of similar emails in terms of content is captured.

Autoencoder is then employed to reduce the feature space into more robust feature representation during the *Feature Reduction* phase. This is to ensure that the information regarding the closeness of email contents is still captured in the lower dimensional feature space. The relu (Maas et al. 2013) activation function was employed in every layer of the Autoencoder. This feature reduction approach ensured that there was an improved classification accuracy.

In this study, the classification algorithms were implemented using Weka[4] which consists of learning algorithms that have the capability to distinguish labeled data. This classification phase included *Hyperparameter Optimization* which was important for training SVM. RBF kernel function with $C = 100$ and $\lambda = 0.001$ was found to lead to a better performance for SVM. Optimization of hyperparameters was done using grid search algorithm with 10-fold CV.

The experimental design in Figure 4.2 investigates the performance of RF, C4.5 and SVM on various feature representation which include.

- DM + DBOW and Sigmoid + Autoencoder
- DM + DBOW and Laplacian + Autoencoder
- DM + DBOW and RBF + Autoencoder
- DM + DBOW and RBF + Autoencoder
- DM + DBOW and CS + Stacked Autoencoder

These proposed feature representations differ with the one in Figure 4.1 because the RBF, Sigmoid and Laplacian are considered for feature transformation. These feature transformations are then compared to CS approach. Stacked Autoencoder is also introduced for feature reduction.

[4]Waikato Environment for Knowledge Analysis

4.5 Experimental setup of supervised methods for feature reduction

Figure 4.3 presents approaches which have been utilized in many previous studies for identifying spam emails. These traditional feature representation will be employed on Trec07 data and Enron data for comparison with the approach proposed in Figure 4.1. There is a huge difference between the approaches which are taken into consideration in this section compared to the approach which was introduced in Section 4.4. The methods of this section for comparison are supervised for feature reduction. The methods being employed for feature reduction are IG and χ^2. All these methods make use of class category to evaluate the significance of a feature variable. However, Autoencoder in contrast to IG and χ^2 for feature reduction does not require any knowledge regarding the class category of the training instance.

For feature extraction in this section BoW with frequency values, BoW with binary values and TF-IDF are employed for feature extraction. These methods differ with DM + DBOW because they do not take word sequence into consideration. Every word is treated independently. Hence, the words with similar meaning but different character combination can not be identified. As a result, we end up with the stopwords not being important for delineating the spam emails from the legitimate emails.

During the *Preprocessing* phase all the stopwords and null characters are removed. All the words and characters are converted to their lower case. Stemming is also employed to convert all the words into their root format. During the *Feature Extraction* phase all the feature extraction methods are utilized to generate numeric representation of each email document. We end up having the following five traditional feature representations:

- BoW + IG
- BoW + χ^2
- BoW (Binary values) + IG
- TF-IDF + IG
- TF-IDF + χ^2

On all of the mentioned feature representations, we employ C4.5, RF, and SVM for classification. For SVM we further investigate optimal hyperparameters for each feature representation. For performance evaluation, F-measure, Precision, and Recall are being utilized.

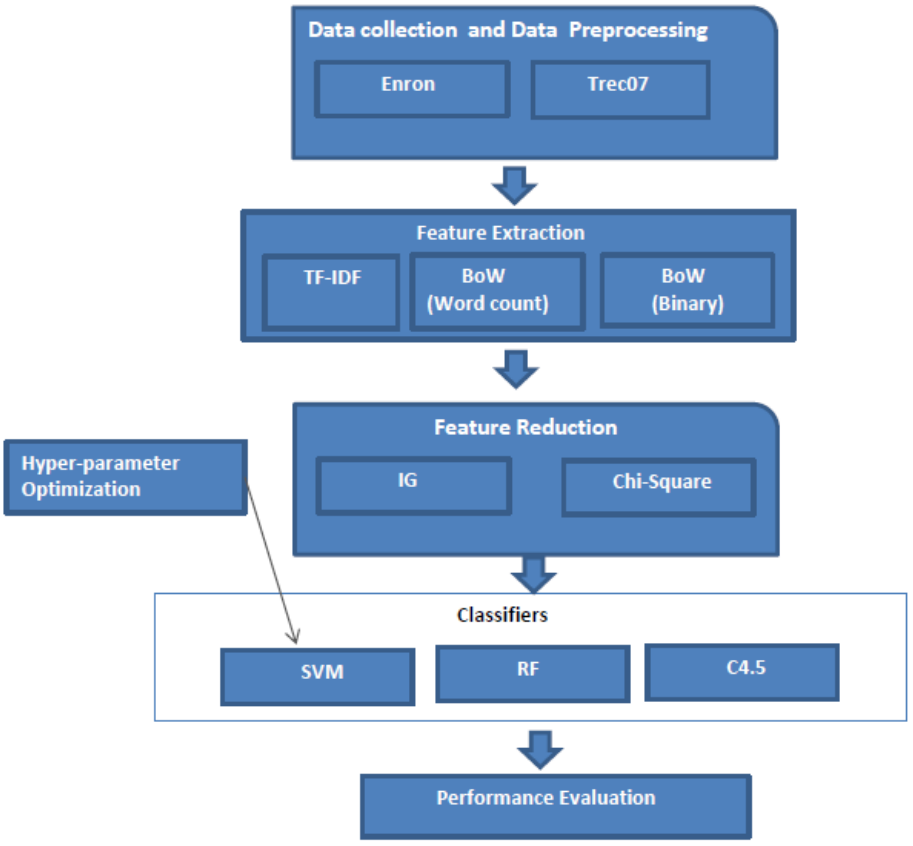


FIGURE 4.3: Supervised Feature Reduction

4.6 Experimental setup of SVD based method for feature reduction

In this section, we employ the unsupervised feature reduction approaches which have also been used mostly from previous studies for feature reduction in problems related to spam email detection. The two feature reduction approaches which are taken into consideration for further comparison are latent semantic analysis (LSA) (Qian et al. 2010) and principal component analysis (PCA) (Abdi & Williams 2010). These models make use of SVD to move the created feature representation to a lower dimensional space.

For feature extraction, only TF-IDF and BoW (with frequency values) are being used. As a result, we end up with the following feature representation:

- BoW + PCA
- BoW + LSA

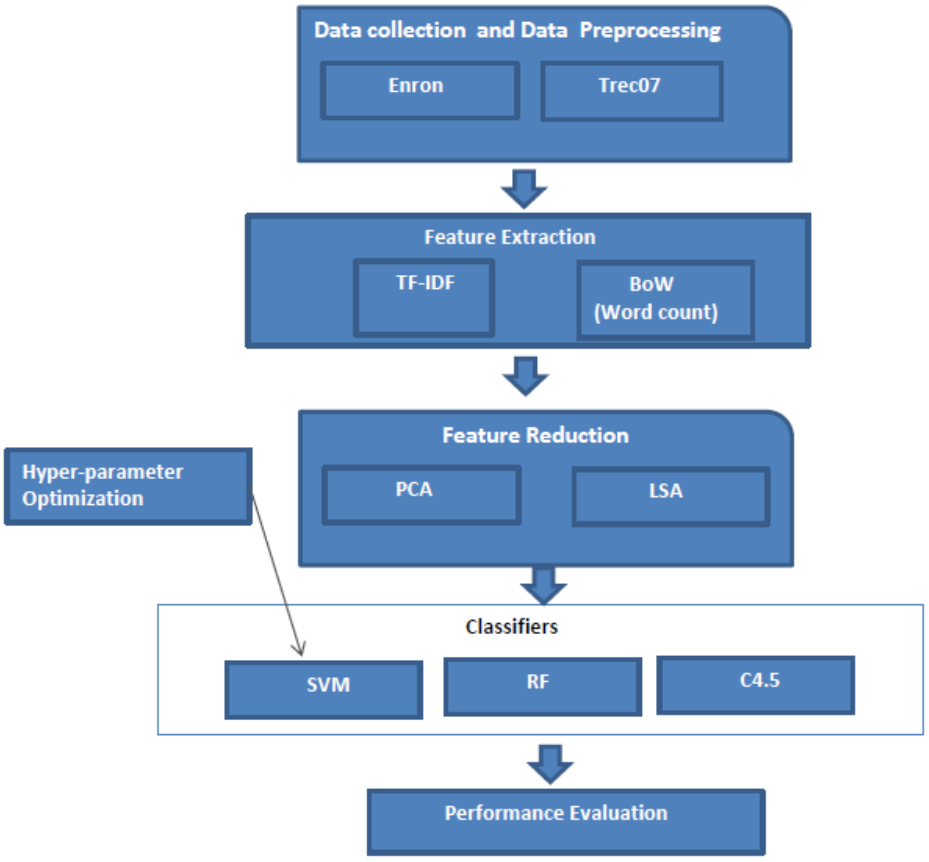


FIGURE 4.4: Unsupervised Feature Reduction

- TF-IDF + PCA
- TF-IDF + LSA

These classifiers, C4.5, RF, and SVM, are assessed using F-measure, Recall, and Precision.

4.7 Data

The effectiveness of each machine learning algorithm[6] is evaluated on the Enron dataset (Metsis et al. 2006). However, for training DM + DBOW model Enron data is combined with IMDB data (Maas et al. 2011). The IMDB data is made up of 50000 movie reviews.

The Enron dataset contains a total of 33, 716 email documents, about 17, 171 are labeled as spam. The *preprocessing* of the unstructured email documents was done by discarding

[6]SVM, RF and C4.5

the header information, subject content, attachments, HTML tags and retaining only the messages within the body to be processed by the proposed approach. The Trec 2007 (Trec07) (Cormack 2007) data for evaluating the performance of the classifiers based on the proposed approach was also utilized. Due to the computational memory issues from Trec07 data, only the first 15000 spam emails and 15000 non-spam emails are used for performance evaluation. The Trec07 dataset consists of 75419 email messages, where 50199 of them are spam and the remaining 25220 are non-spam.

4.8 Performance Metrics

Practical evaluations are conducted by employing varying performance measures. Measures taken into consideration are: Accuracy(ACC), Precision (PrE), Recall (ReC) and F-measure (Fm) as discussed in chapter Chapter 3. Some of the important concepts that should not be avoided when evaluating the performance of the classifiers are true positive rate (t_{pr}) and false positive rate (f_{pr}). With spam detection problem we want to ensure that the number of legitimate emails classified as spam is minimized. The ROC curve is usually used for such cases. It is a curve that explains the performance of a learning classifier with varying threshold function. The plot of the ROC curve has f_{pr} on the horizontal axis and t_{pr} on the vertical axis. t_{pr} is given by

$$t_{pr} = \frac{t_p}{t_p + f_n} \tag{4.2}$$

and f_{pr} is given by

$$f_{pr} = \frac{f_p}{f_p + t_n} \tag{4.3}$$

4.9 Performance of classifiers on features extracted using Neural Network based methods

This section will be showing that the introduction of CS measure and the Autoencoder lead to a better classification performance. This proposed approach, DM + DBOW and CS + Autoencoder, will be compared to the DM + DBOW approach which was introduced by Le & Mikolov (2014). A further comparison between traditional feature construction techniques and the proposed approach (*i.e.*, DM + DBOW and CS + Autoencoder) will be taken into consideration.

For instance, looking at the results presented in Table 4.1b where only the DM + DBOW approach by Le & Mikolov (2014) is used and comparing them with the results presented in Table 4.3 where DM + DBOW and CS + Autoencoder approach is employed, we observe a significant improvement on the performance of a C4.5 classifier. Similarly, when comparing the results presented in Table 4.4b where only DM + DBOW appraoch is used and Table 4.5 where DM + DBOW and CS + Autoencoder approach is used we also observe a significant improvement on both C4.5 and SVM classifiers.

TABLE 4.1: Classifiers performance on DM + DBOW and BoW + IG features

(A) Classifiers performance on Binary Features (Mi et al. 2015).

	PrE (%)	ReC (%)	Fm (%)
SVM	89.64	98.74	93.86
RF	91.46	99.28	95.11
C4.5	82.88	97.07	89.02

(B) An approach by Le & Mikolov (2014), where the DM + DBOW is trained With IMDB and Enron.

	PrE (%)	ReC (%)	Fm (%)
SVM	97.60	97.60	97.60
RF	97.00	97.00	97.00
C4.5	91.80	91.80	91.80

This section examines the strength in terms of performance for the unsupervised learning approach being introduced as a feature extraction and reconstruction strategy for transforming email documents into feature vectors. Experiments were conducted on Enron and Trec07 datasets, which are some of the widely used datasets for evaluating the reliability of models in detecting spam documents. Table 4.1 - Table 4.3 show comparison of proposed solution with results reported in (Mi et al. 2015) where IG and BoW (*i.e.,* binary representation) were used for feature construction. The results obtained in Table 4.1 - Table 4.3 are based on Enron data. While results in Table 4.4 - Table 4.5 are based on Trec07 data. For performance evaluation, 10-fold cross validation was used.

Table 4.1 - Table 4.5 show performances of classifiers considered based on Precision (PrE), Recall (ReC) and F-measure (Fm) for grouping email documents as non-spam or spam. Table 4.1a presents results based on BoW and IG for feature representation, these results were reported in (Mi et al. 2015). Compared to the results in Table 4.1b in terms of F-measure where DM + DBOW model was trained by generating a dense representation of each email document with 50 continuous feature values, BoW + IG shows to have lower performance on classifiers. Note that when training DM + DBOW and classifiers, stemming and stopwords removal were not applied.

Table 4.2b shows results when stemming and stopwords removal are applied. The performance of classifiers declines. This clearly shows that the removal of stopwords and stemming can negatively affect the performance of classifiers. Therefore, it is important to have stopwords and not apply stemming when training models that take word order into consideration.

TABLE 4.2: Impact of IMDB and Noise removal on classifiers performance based on the features generated using the DM + DBOW approach introduced by Le & Mikolov (2014).

(A) Classifiers performance when DM + DBOW trained without IMDB on Enron.

	PrE (%)	ReC (%)	Fm (%)
SVM	61.40	61.40	61.40
RF	60.70	60.20	60.20
C4.5	58.50	58.50	58.50

(B) Classifiers performance when stemming and stop words removal are employed.

	PrE (%)	ReC (%)	Fm (%)
SVM	95.80	95.80	95.80
RF	96.60	96.60	96.60
C4.5	89.80	89.80	89.80

TABLE 4.3: Classifiers performance on features generated by DM + DBOW and CS + Autoencoder on Enron dataset.

	PrE (%)	ReC (%)	Fm (%)
SVM	97.80	97.80	97.80
RF	97.10	97.10	97.10
C4.5	94.20	94.20	94.20

Table 4.2a visualizes the performance results of classifiers on Enron data when only Enron Data is used to train DM + DBOW model for learning document feature representation vector. From these results, we observe that training DM + DBOW with Enron data only declines the performance of classifiers. However, combining both Enron and IMDB shows a significant improvement on classifiers performance in Table 4.1b. Pay attention to the fact that in Table 4.1b and Table 4.2b Enron and IMDB data were combined to train DM + DBOW. In Table 4.3 CS + Autoencoder are introduced to generate more robust features and they did improve classifiers performance in terms of Fm score.

Table 4.2a presents classifiers performances on the Enron dataset in a situation where only Enron dataset is used for training DM + DBOW model for learning document feature representation. From the reported results we observe that training DM + DBOW with only the Enron data declines the performance of the classifiers. However, combining both Enron and IMDB shows a significant improvement on classifiers performance in Table 4.1b. In Table 4.3 CS + Autoencoder are introduced to generate more robust features and they led classifiers to an improved performance.

Table 4.4 presents classifiers performance based on the BoW + IG approach and the DM + DBOW approach for feature construction on the Trec07 dataset. Similarly as in Table 4.1b stopwords removal and stemming were not applied during the preprocessing phase. However, IMBD data was not combined with Trec07 data for training DM + DBOW model. Table 4.4a shows the results that were obtained when BoW + IG with binary values were used for feature representation. While Table 4.4b shows performance results of classifiers when DM + DBOW model is used for feature representation, DM +

TABLE 4.4: Classifiers performance on Trec07 data.

(A) Classifiers performance when BoW + IG employed on Trec07.

	PrE (%)	ReC (%)	Fm (%)
SVM	80.30	76.30	75.50
RF	79.90	76.70	76.00
C4.5	79.60	76.20	75.50

(B) Classifiers performance when DM + DBOW approach (Le & Mikolov 2014) is employed on Trec07 dataset.

	PrE (%)	ReC (%)	Fm (%)
SVM	96.10	96.10	96.10
RF	97.10	97.10	97.10
C4.5	91.40	91.40	91.40

TABLE 4.5: Classifiers performance when DM and DBOW + CS and Autoencoder approach employed on Trec07

	PrE (%)	ReC (%)	Fm (%)
SVM	98.43	98.40	98.40
RF	97.60	97.60	97.60
C4.5	95.00	95.00	95.00

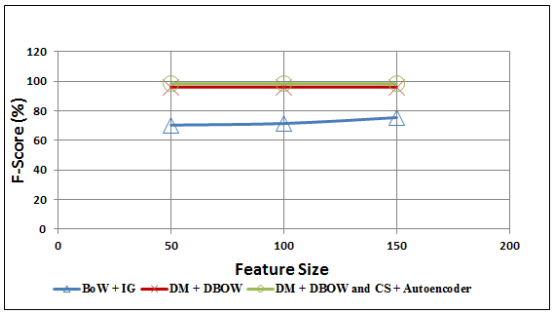
DBOW approach led the classifiers to a significant improvement. Note that a document vector with 50 number of feature was used to train DM + DBOW. It is important to note that the main objective in this chapter is to identify spam emails in a lower dimensional feature space. An optimal feature size was investigated as in chapter 3 by varying feature sizes between 50 and 150 with the difference value of 50.

In Table 4.5 CS + Autoencoder are introduced and a further improvement in all classifiers it is observed. DM + DBOW model was trained similarly as in Table 4.4b.

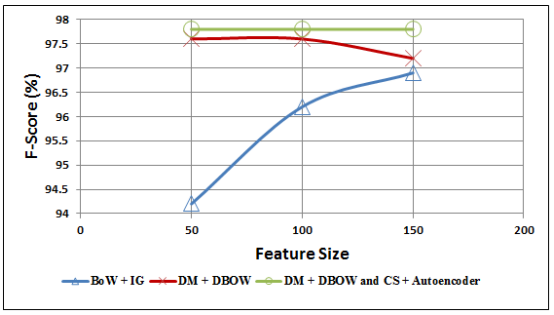
The plots from Figure 4.5 - Figure 4.7 visualize the performance of classifiers (*i.e.*, SVM, RF, C4.5) when BoW + IG, DM + DBOW and, DM + DBOW and CS + Autoencoder are used for feature representation. The number of features is varied into $50, 100, 150$. The performance is based on Fm score.

Figure 4.5 presents the performance of SVM with varying feature representation approaches on Enron and Trec07. Figure 4.5a plot is based on Trec07 while Figure 4.5b plot is based on Enron. Both on Enron and Trec07 data the performance of SVM declined with the decrease of features when BoW + IG approach is used for feature representation. When DM + DBOW approach is employed for feature representation the performance slightly declines with the increase of the feature size on the Enron data. The use of DM + DBOW together with CS + Autoencoder on the SVM performance does not show a significant difference. The Fm score on SVM is always beyond 98% with Trec07 and beyond 97% with Enron data when the DM + DBOW and CS + Autoencoder approach is employed.

RF shows to perform well with DM + DBOW in Figure 4.6a, looking at the Fm score with varying number of features there is a slight difference in performance on Trec07

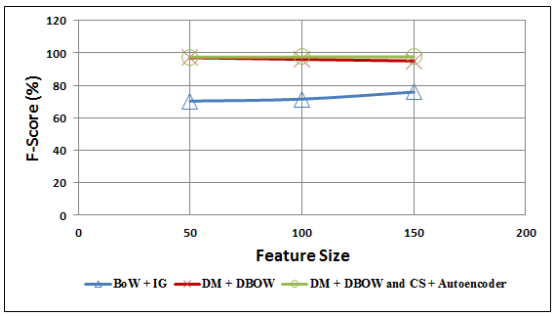


(A) SVM performance on Trec07 dataset

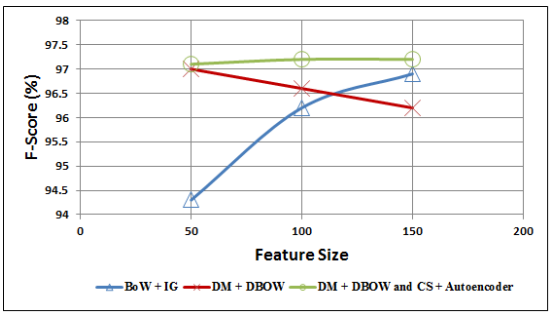


(B) SVM performance on Enron dataset

FIGURE 4.5: BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the SVM performance.



(A) RF performance on Trec07 dataset



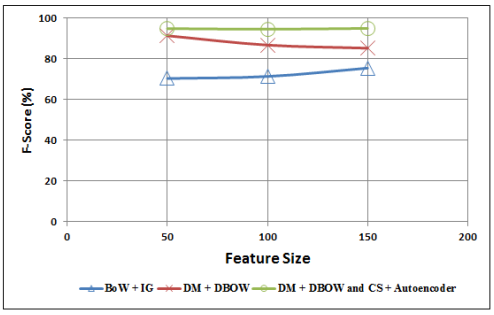
(B) RF performance on Enron dataset

FIGURE 4.6: BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the RF performance.

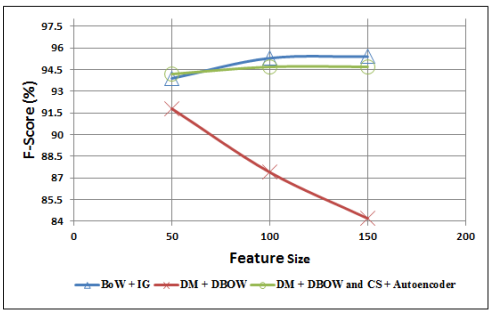
data. Similarly with DM + DBOW together with CS + Autoencoder the performance of RF is still good and there is a slight difference with varying number of features on Trec07 data. However, with BoW + IG there is a major performance decline in terms of the Fm when the feature size decreases on Trec07 dataset.

Figure 4.6b presents RF performance on the Enron dataset. DM + DBOW representation and BoW + IG representation make RF to have a serious inconsistent performance with varying number of features. But DM + DBOW together with CS + Autoencoder in most cases ensures good performance for RF.

In Figure 4.7b it is clear that C4.5 performs well when BoW + IG approach is used for feature representation on Enron data. However, C4.5 underperformed when employed on Trec07 data with BoW + IG for feature representation in Figure 4.7a. In Figure 4.7 both on Enron and Trec07 performance of C4.5 declines with increase in feature size, when DM + DBOW approach is employed for feature representation. DM + DBOW together with CS + Autoencoder ensures that C4.5 on both Enron and Trec07 datasets performs well. Furthermore, in Figure 4.7 it is observed that there is no significant difference in the Fm score with varying number of features when DM + DBOW together with CS + Autoencoder approach is used for feature representation.



(A) C4.5 performance on Trec07 dataset



(B) C4.5 performance on Enron dataset

FIGURE 4.7: BoW + IG, DM + DBOW, and DM + DBOW with CS + Autoencoder for feature representation on Enron and Trec07 datasets to assess the C4.5 performance.

TABLE 4.6: RBF kernel with Autoencoder

	PrE (%)	ReC (%)	Fm (%)
SVM	96.10	96.10	96.10
RF	97.10	97.10	97.10
C4.5	96.00	96.00	96.00

From the plots in Figure 4.5 - Figure 4.7 it is clear that DM + DBOW together with CS + Autoencoder ensures that classifiers performs well with different datasets and varying number of features.

This study makes a further investigation on how kernel functions can affect the performance of classifiers when employed on features generated using DM + DBOW. Kernel functions are employed to transform the generated features to a higher feature space. Autoencoder is then employed for feature reduction.

Table 4.6 presents results of classifiers on features which were firstly generated using DM + DBOW model. Features generated by DM + DBOW were then transformed into the higher feature space using RBF kernel. To overcome overfitting issues, Autoencoder was then employed to reduce the feature space size. SVM, RF and C4.5 were employed to identify spam emails from non-spam emails. Using those features RF has shown a better performance as compared to SVM and C4.5. Between SVM and C4.5 there was no a major difference in their performance based on Fm. They had a difference value of 0.10%. Trec07 data was employed for performance evaluation.

Comparing the performances of all the classifiers in Table 4.6 to their performances in Table 4.5 where DM and DBOW + CS and Autoencoder approach was employed for feature representation, there was a downfall in SVM and RF performances. However, C4.5 improved from 95.00% to 96.00, RF declined from 97.60% to 97.10% and SVM declined from 98.40% to 96.10%.

TABLE 4.7: Laplacian Kernel with Autoencoder

	PrE (%)	ReC (%)	Fm (%)
SVM	94.60	94.60	94.60
RF	96.50	96.50	96.50
C4.5	95.00	95.00	95.00

TABLE 4.8: Sigmoid Kernel with with Autoencoder

	PrE (%)	ReC (%)	Fm (%)
SVM	87.90	87.10	87.50
RF	94.60	94.60	94.60
C4.5	91.90	91.90	91.90

Table 4.7 in contrast to Table 4.6 where the RBF kernel is employed for feature transformation the Laplacian Kernel is utilized instead. The performances of all the classifiers in Table 4.7 have declined in terms of Fm as compared to their performances in Table 4.6. However, the C4.5 has achieved a similar performance in Table 4.7 and Table 4.5. Note that for feature representation Table 4.7 approach differs with Table 4.5 approach because in Table 4.5 CS measure was used for feature transformation instead of the Laplacian kernel.

Table 4.8 makes use of the Sigmoid kernel for feature transformation. SVM shows a poor performance when compared to all the classifiers. Furthermore, compared to the approaches used in Table 4.5, Table 4.6 and Table 4.7 SVM still has the worst performance with the approach used in Table 4.8 where the Sigmoid kernel is employed for feature transformation.

In Table 4.9 the approach used for feature representation differs with the approach used in Table 4.5 because Stacked Autoencoder is introduced for feature reduction instead of Autoencoder with a single hidden layer. The Stacked Autoencoder introduced has five hidden layers including the middle layer known as a bottleneck. The sizes of the layers are as follow: Input layer has 30000 neurons. The second layer has 2000 neurons, the third layer has 1000 neurons and the middle layer has 50 neurons. The first layer following middle layer has 1000 neurons, followed by the layer with 2000 neurons and lastly output layer with the same number of neurons as input layer.

For feature transformation in Table 4.9 Cosine similarity measure was employed. SVM has outperformed all classifiers which are C4.5 and RF. Although in Table 4.5 SVM showed better performance as compared to its performance in Table 4.9, the difference was not significant. Based on the Fm score, SVM declined from 98.40 to 98.30. There might be many factors that affected performance. For instance, it was challenging to investigate optimal number of layers and neurons due to computational memory issues

TABLE 4.9: Cosine Similarity with Stacked Autoencoder

	PrE (%)	ReC (%)	Fm (%)
SVM	98.3	98.3	98.3
RF	96.9	96.9	96.9
C4.5	92.10	92.10	92.10

TABLE 4.10: SVM performance with binary features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	78.10	72.00	70.40	72.04
100	78.00	72.80	71.50	72.79
150	80.30	76.30	75.50	76.34

TABLE 4.11: SVM performance with DM + DBOW features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	96.10	96.10	96.10	96.10
100	96.10	96.10	96.10	96.08
150	96.10	96.10	96.10	96.09

which were encountered. The performance of C4.5 and RF have declined when compared to their performance in Table 4.5.

Table 4.10 - Table 4.12 show the performance of SVM on Trec07 data with the following feature representation:

- BoW + IG
- DM + DBOW
- DM + DBOW and CS + Autoencoder

For each feature representation 50, 100 and 150 features are considered for evaluating how varying sizes might affect the classifiers performance. Table 4.10 presents the results of SVM when BoW and IG are employed for feature construction. Table 4.11 presents the results of SVM when DM + DBOW approach is used to generate dense feature representation. Table 4.12 presents the results of SVM when DM + DBOW and CS + Autoencoder are used for feature representation. Making a comparison of these three feature construction approaches, DM + DBOW and CS + Autoencoder approach shows to generate more robust features for delineating spam emails from non-spam emails on Trec07 data. SVM has the worst performance when BoW + IG approach is used for feature construction. DM + DBOW ensures a better performance on SVM compared to when BoW + IG approach is employed.

Table 4.13 presents results of C4.5 when BoW and IG are employed for feature construction. Table 4.14 presents results of C4.5 when DM + DBOW approach is used

TABLE 4.12: SVM performance with DM + DBOW and CS + Autoencoder features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	98.40	98.40	98.40	98.43
100	98.40	98.40	98.40	98.43
150	98.43	98.40	98.40	98.43

TABLE 4.13: C4.5 performance with binary features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	78.00	72.00	70.40	72.00
100	78.00	72.80	71.40	72.76
150	79.60	76.20	75.50	76.21

TABLE 4.14: C4.5 performance with DM + DBOW features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	96.10	96.10	96.10	96.10
100	96.10	96.10	96.10	96.11
150	95.10	95.10	95.10	95.08

TABLE 4.15: C4.5 performance with DM + DBOW and CS + Autoencoder features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	98.40	98.40	98.40	98.43
100	98.40	98.40	98.40	98.43
150	98.43	98.40	98.40	98.43

to generate features for training classifiers. Table 4.15 presents results of C4.5 when DM + DBOW and CS + Autoencoder are used for feature representation. Making a comparison of these three feature construction approaches, similarly to the performance of SVM, DM + DBOW and CS + Autoencoder approach shows to generate more robust features for delineating spam emails from non-spam emails on Trec07 data. More additionally, C4.5 has the worst performance when BoW + IG approach is used for feature construction. DM + DBOW ensures that C4.5 has a better performance compared to when using binary features for learning the classifiers.

Table 4.16 presents performance results of RF when BoW and IG are employed for feature construction. Table 4.17 presents results of RF performance when DM + DBOW approach is used to generate features for training classifiers. Table 4.18 presents results of Rf when DM + DBOW and CS + Autoencoder are used for feature representation. For analysis these three feature construction approaches, similarly to the performance of RF, DM + DBOW and CS + Autoencoder approach shows to generate more robust features for delineating spam emails from non-spam emails on Trec07 data. More additionally, RF has the worst performance when BoW + IG approach is used for feature construction.

TABLE 4.16: RF performance with binary features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	77.90	72.00	74.95	71.96
100	78.10	72.90	75.50	72.92
150	79.90	76.70	78.30	76.69

TABLE 4.17: RF performance with DM + DBOW features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	97.10	97.10	97.10	97.09
100	96.10	96.10	96.10	96.11
150	95.10	95.10	95.10	95.08

TABLE 4.18: RF performance with DM + DBOW and CS + Autoencoder features on Trec07

Feature Size	PrE(%)	ReC(%)	Fm(%)	ACC(%)
50	97.50	97.50	97.50	97.50
100	97.60	97.60	97.60	97.63
150	97.60	97.60	97.60	97.59

DM + DBOW ensures that RF has a better performance compared to when using binary features for learning the classifiers.

The results presented in Figure 4.8 - Figure 4.9 are based on different feature construction approaches and varying classification methods. During the implementation of every learning classifier, the 10-fold CV model was employed. The results presented in Figure 4.8 correspond to the BoW + IG feature construction approach. All the classifiers performances get better with the increasing number of the feature variables. The RBF kernel was found to be the optimal one for the SVM performance when the BoW + IG approach is employed during the construction of the feature representation. The optimal parameters found were $C = 1$ and $\lambda = 0.1$.

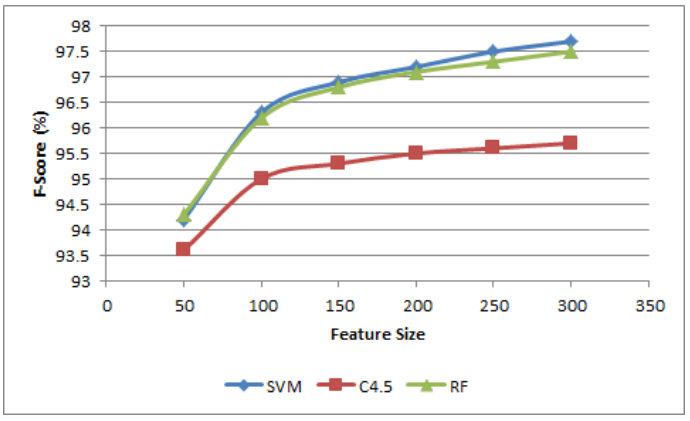


FIGURE 4.8: Machine learning classifiers performance on features generated by the BoW + IG approach with varying feature sizes.

Figure 4.9 presents the results of the ML classifiers based on the feature representation that was constructed using the TF-IDF + IG approach. The SVM model shows a better a better performance with the linear kernel function. The parameter $C = 1$ is found as the optimal value in this case. The SVM performance degraded with the increasing value of C.

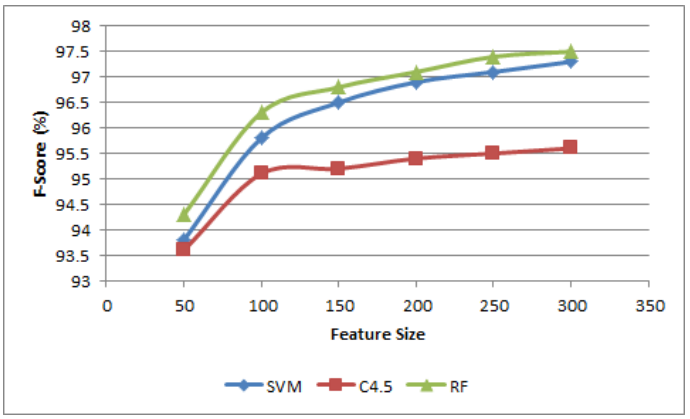


FIGURE 4.9: Machine learning classifiers performance on features generated by the TF-IDF + IG approach with varying feature sizes.

In Figure 4.10 χ^2 is employed to determine feature variables that are more informative for the classification task. The BoW technique was employed for feature construction. The same kernel that has been used with the BoW + IG approach in Figure 4.8 was also found optimal with the BoW + χ^2 approach.

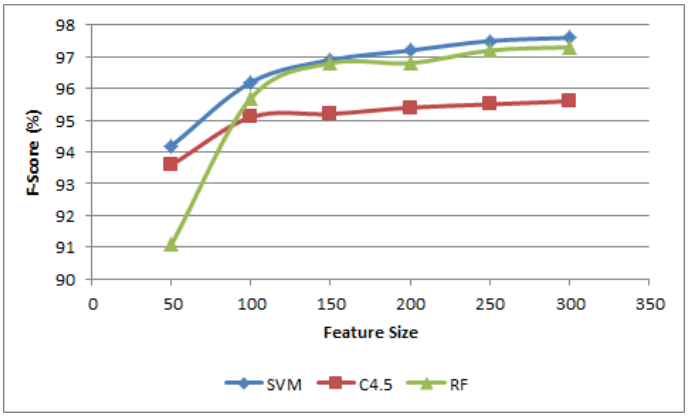


FIGURE 4.10: Machine learning classifiers performance on features generated by the BoW + χ^2 approach with varying feature sizes.

When using the BoW approach for feature construction the RBF kernel function in most cases leads the SVM to a good performance with varying feature selection approaches. Looking at the Figure 4.10 and Figure 4.8 there is no a major difference in an Fm score for the SVM. It is worth noting that in these figures, Figure 4.10 and Figure 4.8, regardless of using different feature selection approaches, the RBF was found as the optimal kernel for the SVM.

Additionally, it is clear that the reduction of the feature space when using term count based feature construction approaches like TF-IDF and BoW we observe a decline in the performance for classifying instances most of the time. This might be due to the fact that features that are low ranked by the feature selection approaches are usually being discarded and not used for training learning classifiers.

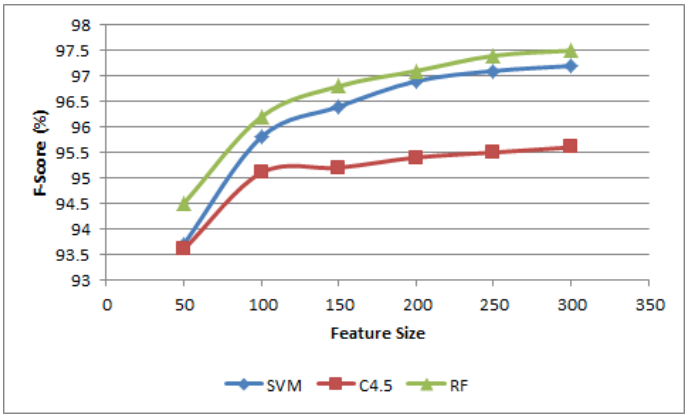


FIGURE 4.11: Machine learning classifiers performance on features generated by the TF-IDF + χ^2 approach with varying feature sizes.

4.10 Performance of classifiers on features extracted using SVD related methods

Table 4.19 - Table 4.26 present performance of classifiers based on features constructed using traditional unsupervised feature construction methods. Table 4.19 and Table 4.20 present the performance of classifiers on Enron and Trec07 data respectively. The feature representation in Table 4.19 and Table 4.20 were generated using BoW and LSA.

Looking at Table 4.19 SVM outperformed all classifiers with the Fm value of 98.10% on Enron dataset. It was followed by RF with Fm value of 97.50%. C4.5 got the worst performance with the Fm value of 95.60%.

TABLE 4.19: BoW + LSA for feature construction on Enron data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	98.10	98.10	98.10	98.11
RF	97.50	97.50	97.50	97.47
C4.5	95.60	95.60	95.60	95.55

Using the same approach on Trec07 data as it was employed on Enron data we observe that classifiers performance degraded significantly. BoW + LSA for feature reduction makes classifiers to be inconstant with different data. Table 4.20 presents the results of classifiers on Trec07 data using BoW + LSA for feature construction. All classifiers

performance based on PrE, ReC, Fm and ACC is bellow 85%. However, with the same classifiers on the same data using CS + Autoencoder and DM+DBOW for feature construction, all classifiers managed to reach at least 95% with PrE, ReC, Fm and ACC.

TABLE 4.20: BoW + LSA for feature construction on Trec07 data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	84.30	84.20	84.20	84.21
RF	85.00	84.70	84.70	84.71
C4.5	82.50	82.20	82.20	82.24

In Table 4.19 and Table 4.20, all the results are based on features extracted using BoW + LSA but different datasets. In Table 4.21 and Table 4.22 for feature construction the BoW + PCA approach is employed for feature construction. Similar situation as in Table 4.19 and Table 4.20 is observed. Classifiers shows to perform well with Enron data and poor with Trec07 dataset.

TABLE 4.21: BoW + PCA for feature construction on Enron data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	98.10	98.10	98.10	98.06
RF	97.60	97.60	97.60	97.59
C4.5	95.60	95.60	95.60	95.55

TABLE 4.22: BoW + PCA for feature construction on Trec07 data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	84.20	83.90	83.80	83.85
RF	84.90	84.60	84.60	84.63
C4.5	82.40	82.20	82.10	82.15

Table 4.23 - Table 4.26 in contrast to Table 4.19 - Table 4.22 for feature construction TF-IDF is employed. However, similar feature reduction techniques as in Table 4.19 - Table 4.22 are employed (*i.e.*, PCA and LSA). Again with the approaches considered for feature construction, classifiers performs well only with Enron data and the performance significantly degrades when Trec07 data is employed for performance evaluation.

TABLE 4.23: TF-IDF + LSA for feature construction on Enron data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	98.10	98.00	98.00	98.05
RF	97.50	97.50	97.50	97.50
C4.5	95.70	95.70	95.70	95.70

Approaches that were employed for feature construction in Table 4.19 - Table 4.26 make classifiers to have inconsistent performance with the variation of the data. However,

TABLE 4.24: TF-IDF + LSA for feature construction on Trec07 data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	83.70	83.70	83.70	83.67
RF	84.70	84.50	84.40	84.40
C4.5	81.80	81.70	81.70	81.69

TABLE 4.25: TF-IDF + PCA for feature construction on Enron data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	98.10	98.10	98.10	98.06
RF	97.60	97.60	97.50	97.55
C4.5	95.60	95.60	95.60	95.62

TABLE 4.26: TF-IDF + PCA for feature construction on Trec07 data

	PrE(%)	ReC(%)	Fm(%)	ACC(%)
SVM	83.60	83.60	83.60	83.59
RF	84.70	84.40	84.30	84.38
C4.5	82.00	81.90	81.80	81.85

they improved the performance of the classifiers compared to when BoW + IG, BoW + χ^2, TF-IDF + IG and TF-IDF + χ^2 are employed for feature construction.

Figure 4.12 - Figure 4.19 shows the performance of classifiers with varying feature sizes for learning classifiers with approaches that were considered in Table 4.19 - Table 4.26 for feature construction. In Figure 4.12 where Enron data is employed for performance evaluation the SVM performance improves with the increase of feature size while RF and C4.5 their performances slightly decline with the increase of the feature size.

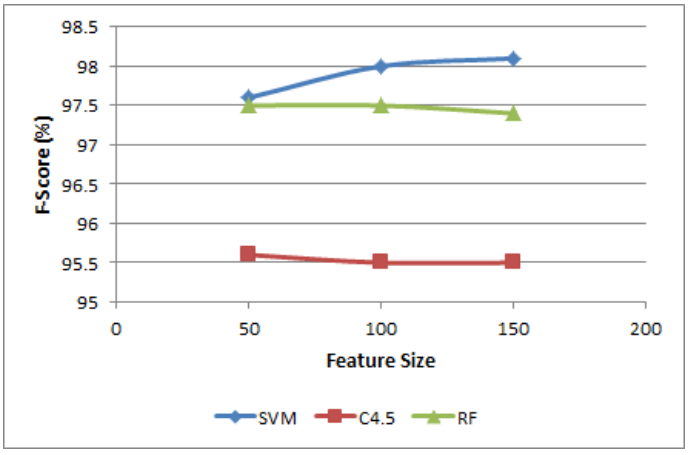


FIGURE 4.12: Performance of the learning classifiers based on the Enron data using the BoW + PCA approach for feature construction.

In Figure 4.13 SVM with the same approach for feature construction as in Figure 4.12 does not have an improving performance with the increase of the feature size. On Trec07 data as it is presented in Figure 4.13 SVM have an inconsistent performance.

However, RF and C4.5 similarly to in Figure 4.12 their performances slightly degrade with increasing feature size.

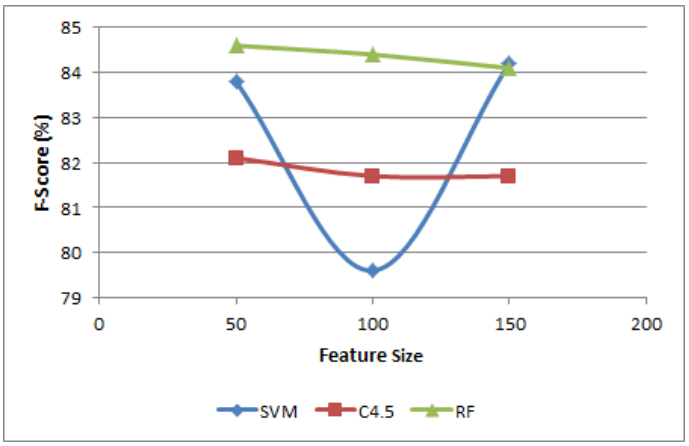


FIGURE 4.13: Machine learning classifiers based on the Trec07 data using the BoW + PCA approach for feature construction.

Figure 4.14 presents the performance of the classifiers when BoW + LSA approach is utilized for feature construction. In this case RF performance improves with increasing feature size. C4.5 has the best performance with small feature size and the worst performance with huge number of features. The SVM performance based on the feature sizes considered, reaches its peak with the 100 feature size used for performance evaluation. When 50 and 150 feature sizes are employed for performance evaluation, the SVM performance degrades and the performance is not much different when 50 and 150 feature sizes are used for performance evaluation.



FIGURE 4.14: Machine learning classifiers based on the Enron data using the BoW + LSA approach for feature construction.

Figure 4.15 presents the classifiers performance on Trec07 data with BoW + LSA approach for feature construction. The feature extraction approach which was used in Figure 4.15 is the same as the one which was utilized in Figure 4.14. In this case Trec07

data is used for performance evaluation, unlike in Figure 4.14 SVM performance improves with increasing number of features. This clearly shows that whenever BoW + LSA approach is employed for feature construction, optimal feature size for training SVM will be investigated with different datasets. This is due the inconsistency BoW + LSA approach brings to SVM.

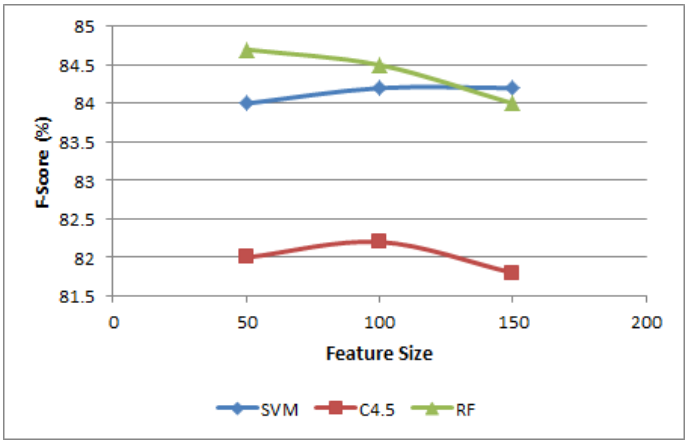


FIGURE 4.15: Machine learning classifiers based on the Trec07 data using the BoW + LSA approach for feature construction.

Figure 4.16 shows the performance of classifiers with varying feature sizes when TF-IDF + PCA approach is being utilized for feature representation. In this case RF on Enron data shows consistent performance with varying feature size. C4.5 performance declines with the increase of feature size while SVM improves with the increase of feature size.

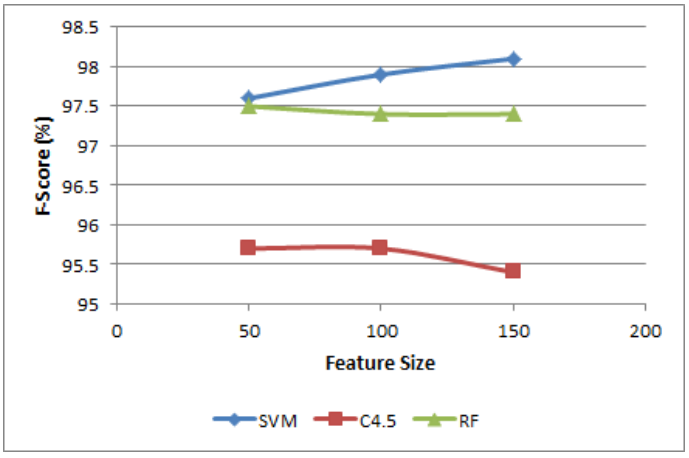


FIGURE 4.16: Machine learning classifiers based on the Enron data using the TF-IDF + PCA approach for feature construction.

In Figure 4.16 RF performance slightly degrades with increase in feature size, but in Figure 4.17 slightly varies with increase of the feature size. Note that in Figure 4.16 the same approach as in Figure 4.17 for feature construction was employed. However, in

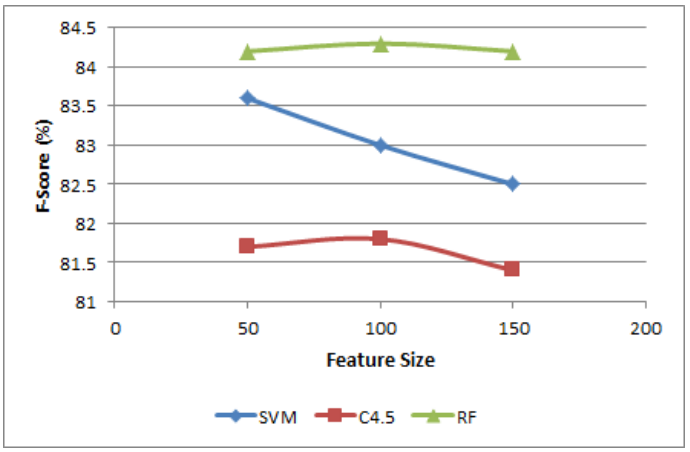


FIGURE 4.17: Machine learning classifiers based on the Trec07 data using the TF-IDF + PCA approach for feature construction.

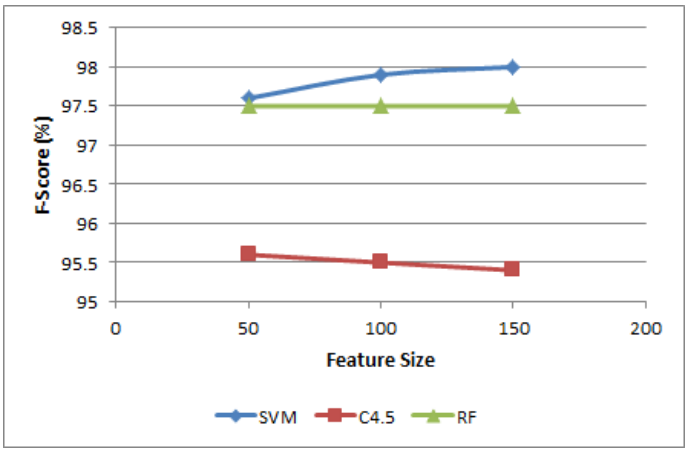


FIGURE 4.18: Machine learning classifiers based on the Enron data using the TF-IDF + LSA approach for feature construction.

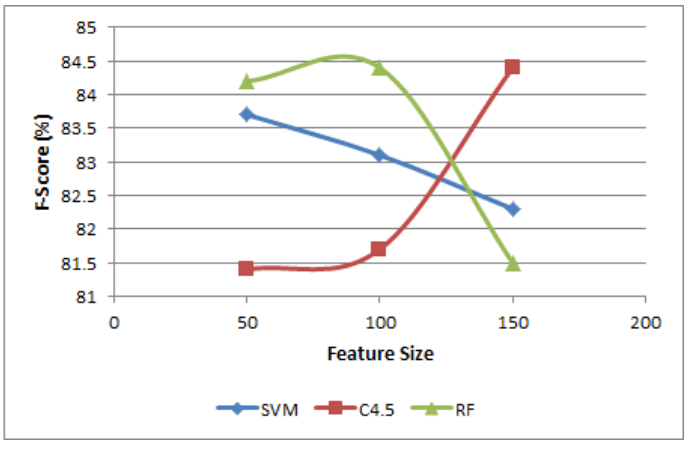
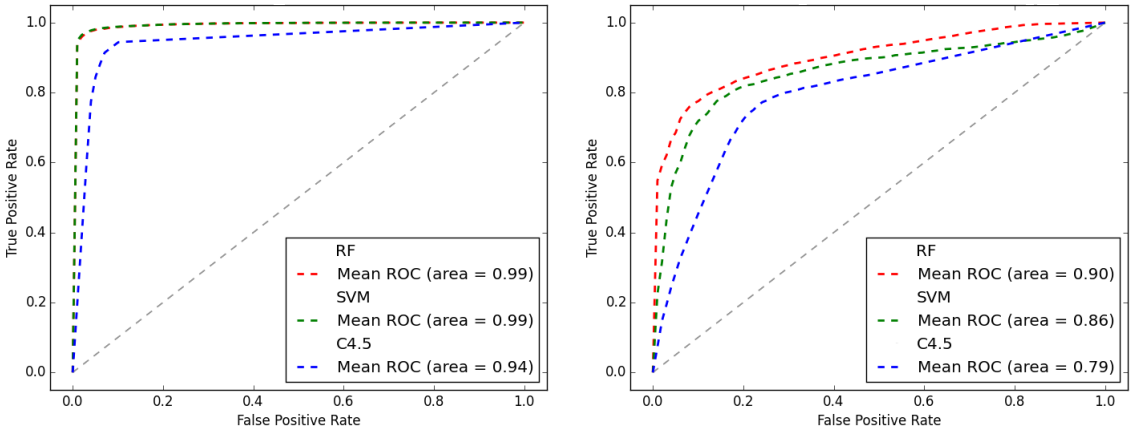


FIGURE 4.19: Machine learning classifiers based on the Trec07 data using the TF-IDF + LSA approach for feature construction.

Figure 4.16 SVM performance improves with increase in feature size. whereas, in Figure 4.17 SVM performance declines with the increase in feature size.

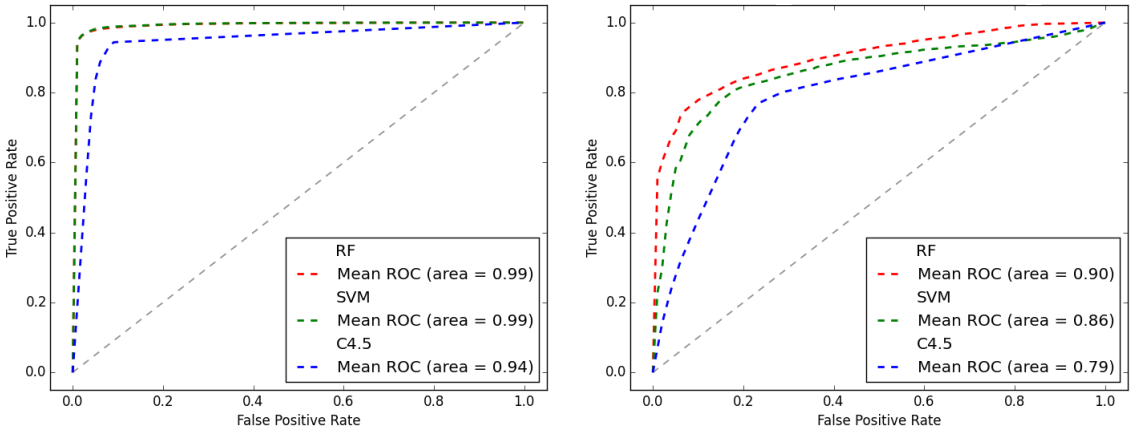
Figure 4.18 and Figure 4.19 respectively present the performance results of classifiers on Enron data and Trec07 data using TF-IDF + LSA for feature construction. With the same approaches but different data, the performance of classifiers was affected differently with varying feature sizes. It becomes difficult to tell if increase in feature size have positive impact towards the performance of classifiers or not. As a result, for determining optimal number of features for training classifier will be very tricky in this case.



(A) ROC curve based on Enron dataset for performance evaluation.

(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.20: Classifiers performance based on ROC curve with features generated using TF-IDF and LSA.

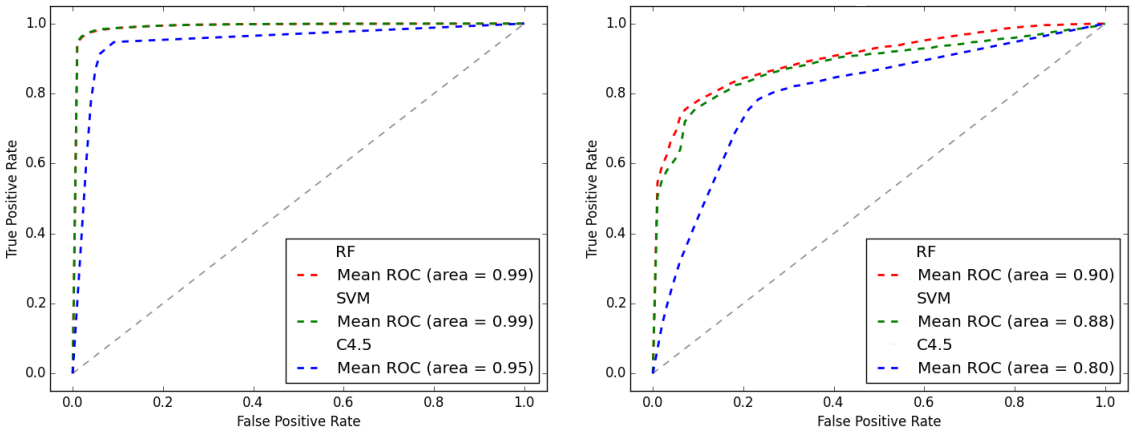


(A) ROC curve based on Enron dataset for performance evaluation.

(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.21: Classifiers performance based on ROC curve with features generated using TF-IDF and PCA.

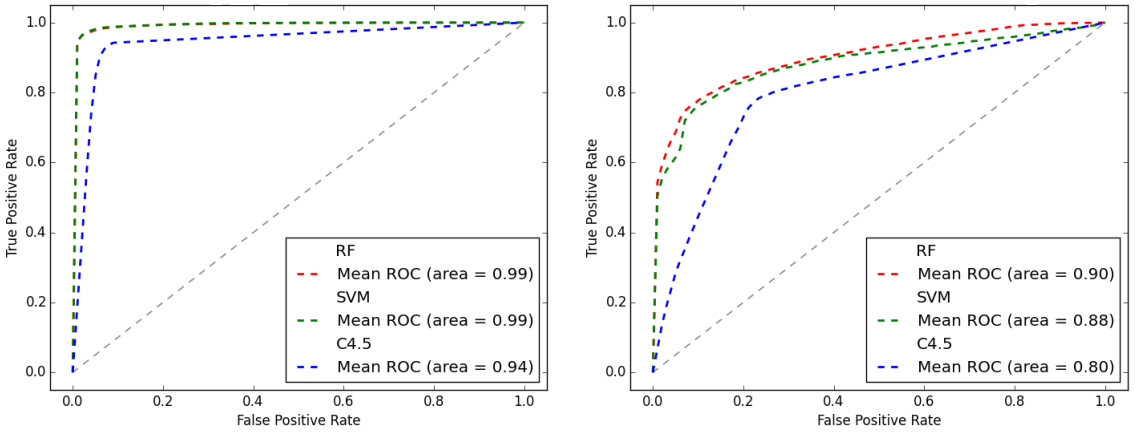
Figure 4.20 presents the performance results regarding the RF, C4.5 and SVM on the features generated using TF-IDF and LSA. The performance in Figure 4.20a is based on the use of Enron dataset for performance evaluation. Figure 4.20b presents the



(A) ROC curve based on Enron dataset for performance evaluation.

(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.22: Classifiers performance based on ROC curve with features generated using BoW and LSA.



(A) ROC curve based on Enron dataset for performance evaluation.

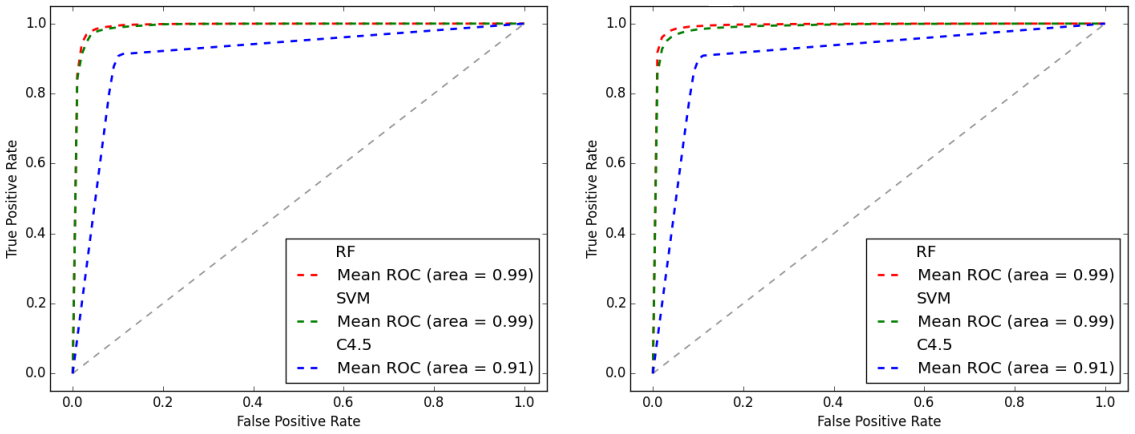
(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.23: Classifiers performance based on ROC curve with features generated using BoW and PCA.

performance results of the same classifiers as in Figure 4.20a, however, Trec07 dataset is employed for the performance evaluation. In Figure 4.20a, SVM and RF based on variation of the threshold function have the similar performance.

In Figure 4.20b the performance of all classifiers degraded, in this case note that Trec07 data is employed for performance evaluation. The performance ranking ranging from the best to the worst is as follows: RF, SVM then followed by C4.5.

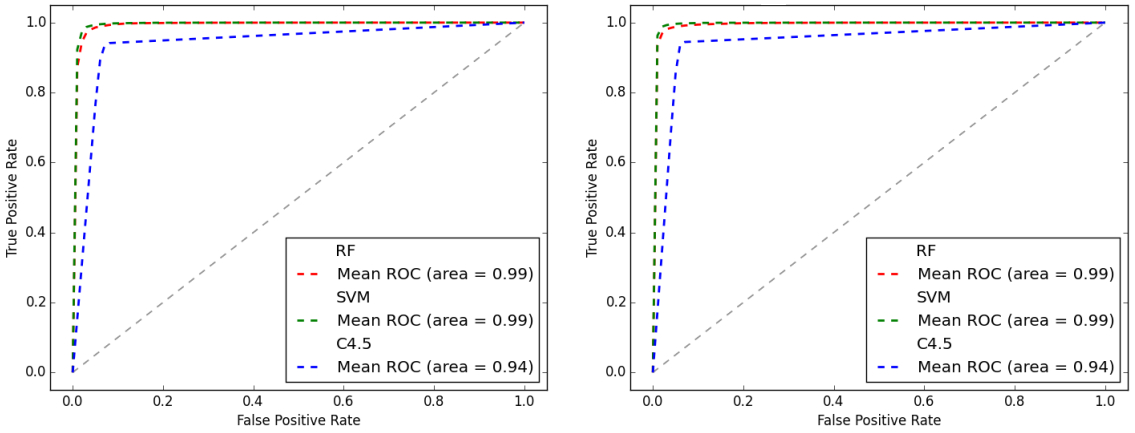
In Figure 4.21 similar datasets and classifiers as in Figure 4.20 are considered. In contrast to Figure 4.20, the performance of the classifiers in Figure 4.21 is based on the use of



(A) ROC curve based on Enron dataset for performance evaluation.

(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.24: Classifiers performance based on ROC curve with features generated using DM + DBOW (Le & Mikolov 2014).



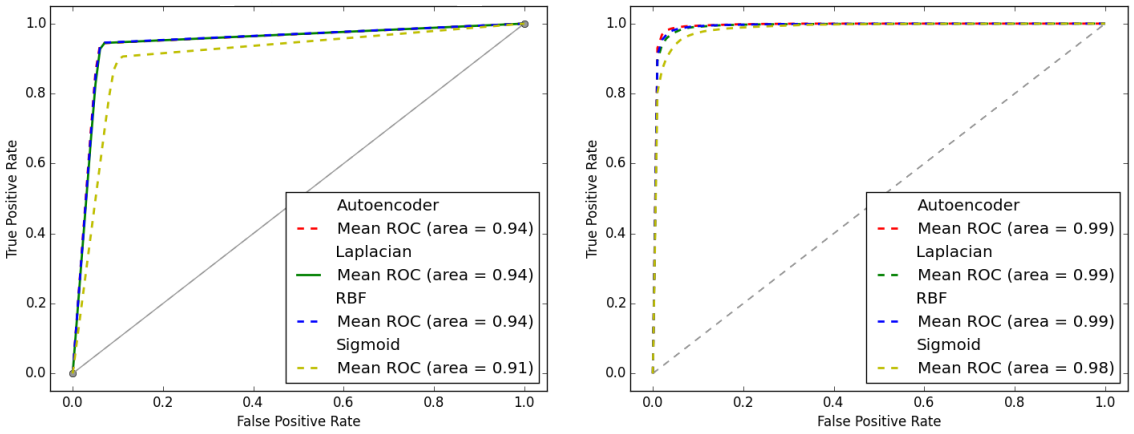
(A) ROC curve based on Enron dataset for performance evaluation.

(B) ROC curve based on Trec07 dataset for performance evaluation.

FIGURE 4.25: Classifiers performance based on ROC curve with features generated using DM + DBOW and CS + Autoencoder.

TF-IDF and PCA for feature representation. Comparing the performance of classifiers both in Figure 4.21a and Figure 4.21b there is a major difference. Again in Figure 4.22 and Figure 4.23, BoW + LSA and BoW + PCA respectively are used for feature representation. The reported performance in Figure 4.20 - Figure 4.23 is almost similar.

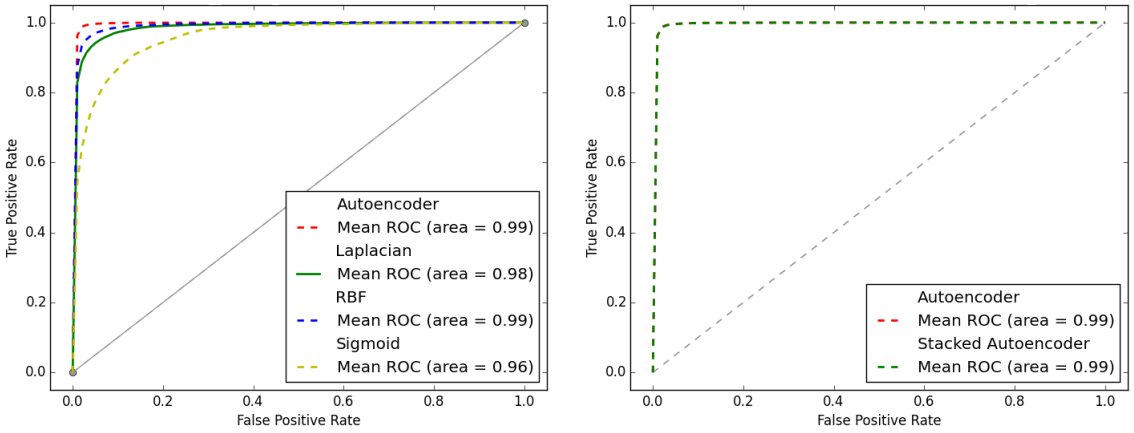
In Figure 4.24 DM + DBOW is introduced for feature representation and we observe a significantly better performance of classifier when Trec07 data is utilized for performance evaluation in Figure 4.24b. Furthermore, in Figure 4.24a and Figure 4.24b the performance of classifiers with different datasets is not significantly different. The introduction of DM + DBOW overcame the issue we had in Figure 4.20 - Figure 4.23 were



(A) C4.5 performance performance based on ROC curve.

(B) RF performance performance based on ROC curve.

FIGURE 4.26: RF and C4.5 performances based on ROC curve with various feature representations using Trec07 dataset for performance evaluation.



(A) Comparison of various feature transformations based on ROC curve.

(B) Comparison of Autoencoder and Stacked Autoencoder for feature reduction based on ROC curve.

FIGURE 4.27: SVM performance based on ROC curve with various feature representations using Trec07 dataset for performance evaluation.

the classifiers performance was significantly different with different datasets.

In Figure 4.25 for feature representation DM + DBOW and CS + Autoencoder approach is employed. There was a further improvement on all classifiers compared to in Figure 4.24. Regarding the performance of the C4.5 in Figure 4.25b we observe a significant improvement as compared to in Figure 4.24b.

Figure 4.26 presents C4.5 and RF performance based on variation of threshold function. Figure 4.26a presents the performance of C4.5. In Figure 4.26a C4.5 had a similar performance when CS measure, Laplacian kernel and RBF kernel are used for feature

transformation. This means when DM + DBOW and CS + Autoencoder, DM + DBOW and Laplacian + Autoencoder, DM + DBOW and RBF + Autoencoder approaches are used for feature representation makes C4.5 have a similar performance with varying threshold function. The worst performance was experienced when Sigmoid is employed for feature transformation.

Again in Figure 4.26b the worst performance was experienced when Sigmoid kernel is employed for feature transformation. Note that in this case the learning classifier is RF in contrast to Figure 4.26a where C4.5 is employed. The best performance was experienced when employing CS measure for feature transformation. The order of performance from best to worst is as follow: CS measure, RBF kernel, Laplacian kernel and Sigmoid kernel.

SVM in Figure 4.27b has a similar performance when Autoencoder and Stacked Autoencoder approaches are used for feature reduction. In Figure 4.27a CS measure ensures the best performance compared to when Laplacian kernel, Sigmoid kernel and RBF Kernel are utilized for feature transformation.

4.11 Summary

In this chapter we observe that as much as word count based feature construction approaches have been used mostly according to the literature as prepossessing phase to automatically identify spam emails, most of the time they fail to ensure good performance with varying datasets. As a result, classifiers tend to have inconsistent performance. For example, the results in Table 3.3 and Table 4.3 are based on the use of Enron data for performance evaluation. The difference in these two tables is that in Table 3.3 a term count based approach for feature construction is employed. Whereas in Table 4.3 the proposed DM + DBOW and CS + Autoencoder approach is employed for feature construction. The word count based approach lead to a more similar performance as the proposed approach. However, when Trec07 dataset is being introduced in Table 4.4a where the same approach as in Table 3.3 is employed, the performance significantly drops. But in Table 4.5 we still have comparable performance results as in Table 4.3. This shows reliability in the proposed approach.

4.12 Conclusions

The results in this chapter clearly shows that word count based feature construction approaches may lead classifiers to an inconsistent performance. For instance, when Enron data is employed for performance evaluation most classifiers shows a better performance.

However, when Trec07 data is being utilized the classifiers performance significantly degrades. Whereas, when using dense feature representation which take word order into consideration classifiers still maintain a good performance with variation of data.

The proposed feature construction approach (*i.e.*, DM + DBOW and CS + Autoencoder) has enabled the classifiers to identify spam emails with high value of Fm in a lower dimensional feature space. Furthermore, the proposed approach has significantly outperformed the traditional feature construction techniques such as BoW, TF-IDF, IG, χ^2, PCA and LSA most of the time. In addition, the proposed approach was more reliable with the variation of the data. The classifiers still managed to reach an excellent performance even when the data was changed from Enron to Trec07. However, the traditional feature construction techniques in most cases led classifiers to a poor performance with Trec07 dataset. From the experiment which was conducted, we found out that the traditional feature construction techniques may not always be reliable for identifying spam emails. The DM+DBOW approach for feature representation was also considered for a further comparison. However, the performance of the classifiers on features generated by DM+DBOW was bellow in most cases compared to when the DM + DBOW and CS + Autoencoder approach is employed for feature construction. Furthermore, DM + DBOW approach tends to lead classifiers to a poor performance in most cases when the feature size increases. The DM + DBOW and CS + Autoencoder approach ensures consistency with the variation of the feature size most of the time. The DM + DBOW and CS + Stacked Autoencoder approach most of the time led to a slightly lower performance compared to DM + DBOW and CS + Autoencoder approach. In addition, this study has shown that stopwords removal and stemming processes when word order is taken into consideration may negatively affect classifiers performance for identifying spam emails. A further investigation on the capability of the Autoencoders to generate meaningful representation from a noisy data will be conducted. This will ensure that classifiers do not suffer from spontaneously changing noisy data in the future.

Chapter 5

Neural Network for Email Spam Detection

5.1 Introduction

This chapter aims to asses the capability of the multilayer perceptron (MLP) as an alternative technique on the feature representation that was proposed in Chapter 4 (*i.e.*, Distributed Memory + Distributed Bag of Words and Cosine Similarity + Autoencoder). Support Vector Machine (SVM) in most cases has shown a much better performance compared to the Random Forest (RF) and C4.5. One of the major challenges with the SVM for classification include the selection of an optimal kernel function and intensive hyperparmeter tuning. For the hyperparameter tuning process, the use of the k-fold cross validation (k-fold CV) technique is encouraged for a better generalization. This can be a challenging task, because it is time consuming. In this chapter we asses the performance of the MLP classifier when employed on the features generated using the DM + DBOW and CS + Autoencoder as it has been explained in Chapter 4. The CS and Kernel functions are employed on features generated using DM + DBOW for a further comparison to project the generated features into the higher dimensional feature space. The Autoencoder and Stacked Autoencoder are employed for feature reduction.

5.2 Experiment

This chapter conducts the experiment as outlined in Figure 5.1. The *document pre-processing* phase extracts meta-information (e.g. message) from the email document. This pre-processing phase involves removal of punctuations and null characters, conversion of all characters and words to their lower case.

DM + DBOW is utilized to generate and learn continuously distributed vectors (Le & Mikolov 2014). The Trec07 dataset is employed to learn DM + DBOW for feature construction. CS measure is then employed to generate features which ensure that similar emails in terms of content are closer to each other. Kernel functions are employed to make comparison with CS measure for feature transformation. The Autoencoder and Stacked Autoencoder are employed to make a comparison for feature reduction. However, Stacked Autoencoder is only employed on the feature representation generated using the DM + DBOW and CS measure approach. In this study MLP classifier was implemented in Keras library (Chollet 2015).

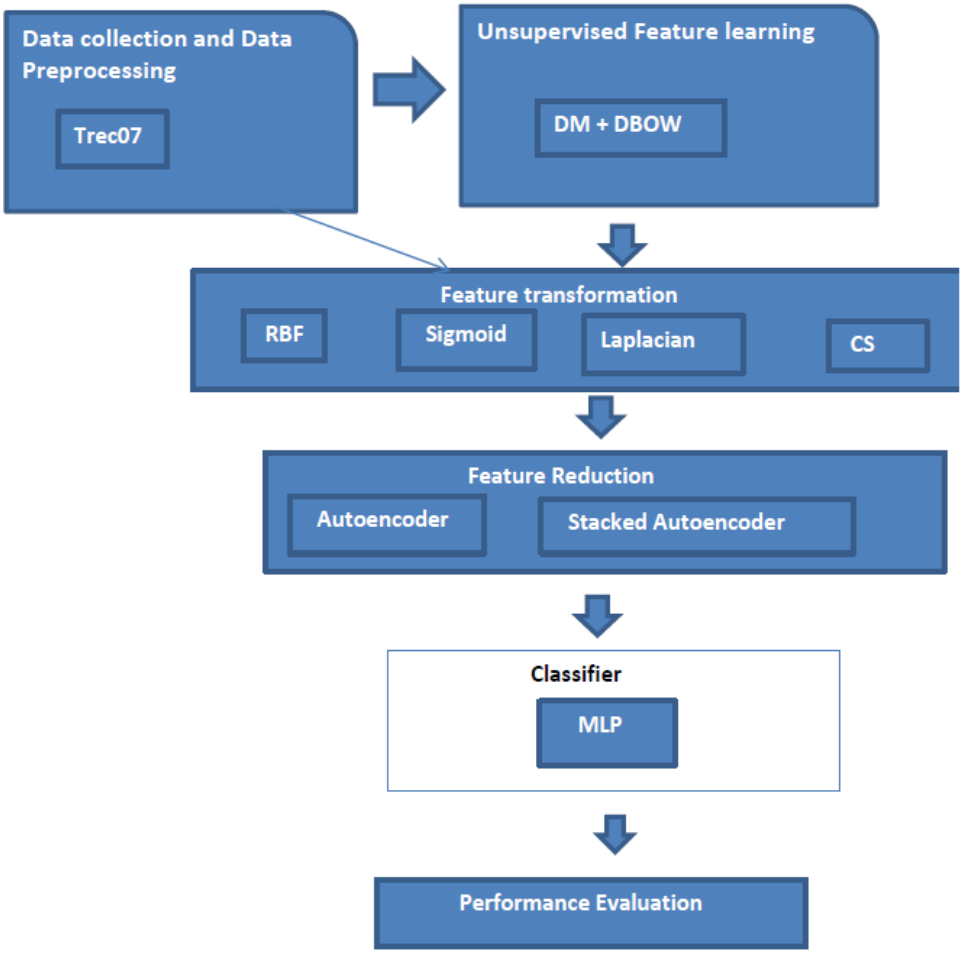


FIGURE 5.1: Autoencoder and Stacked Autoencoder for Feature Reduction

5.3 Performance Metrics

The performance of MLP is evaluated on the following performance measures:

- Recall (ReC)

- Precision (PrE)
- Accuracy (ACC)
- F-measure (Fm)

This performance measures have been discussed in chapter 3. We also employ ROC curve for performance evaluation which has been discussed in chapter 4.

5.4 Results Analysis

This section investigates performance of MLP on various feature representations which include.

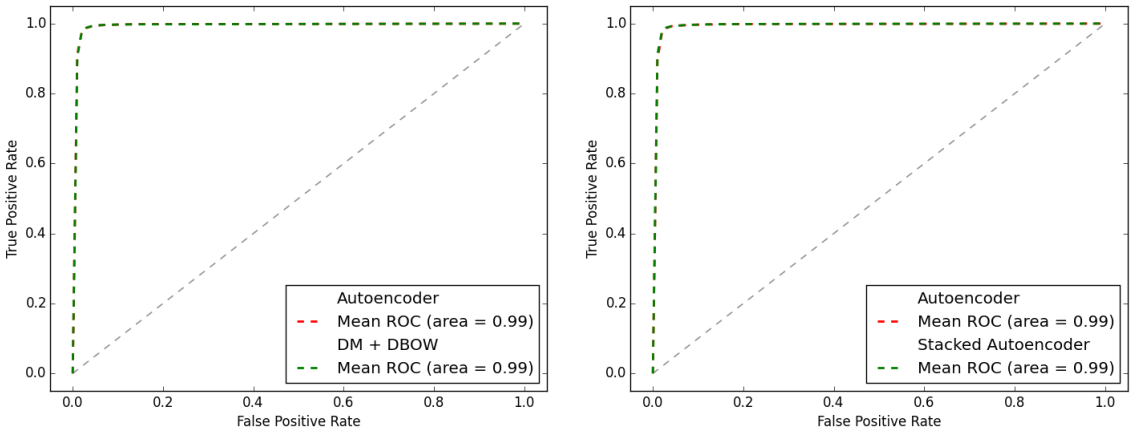
- DM + DBOW
- DM + DBOW and Sigmoid + Autoencoder
- DM + DBOW and Laplacian + Autoencoder
- DM + DBOW and RBF + Autoencoder
- DM + DBOW and RBF + Autoencoder
- DM + DBOW and CS + Stacked Autoencoder

With these feature extraction approaches 50 features were generated by each approach. The MLP algorithm introduced has 5 layers. Those layers are structured as follows: Input layer with 50 neurons, subsequently followed by 40 neurons layer, 30 neurons layer, 20 neurons layer, 10 neurons layer and lastly the output layer. Relu activation function was employed except for the last layer where sigmoid function is employed.

From Table 5.1 MLP shows to have similar performance when

- DM + DBOW
- DM + DBOW and CS + Autoencoder
- DM + DBOW and CS + Stacked Autoencoder

approaches are utilized for feature construction. We observe worst performance when the DM + DBOW with Sigmoid + Autoencoder approach is employed for feature representation. Compared to the other classifiers in Table 4.5 such as SVM, RF, and C4.5,



(A) MLP performance when DM + DBOW approach is used for feature representation.

(B) MLP performance when DM + DBOW and CS + Autoencoder approach is used for feature representation.

FIGURE 5.2: MLP performance based on ROC curve with varying feature representations.

the MLP classifier is outperformed by the only SVM with the difference value of 0.4% based on the Fm score. However, Table 4.8 as compared to Table 5.1 it is clear that MLP performs better than SVM when DM + DBOW and Sigmoid + Autoencoder approach is utilized for feature representation.

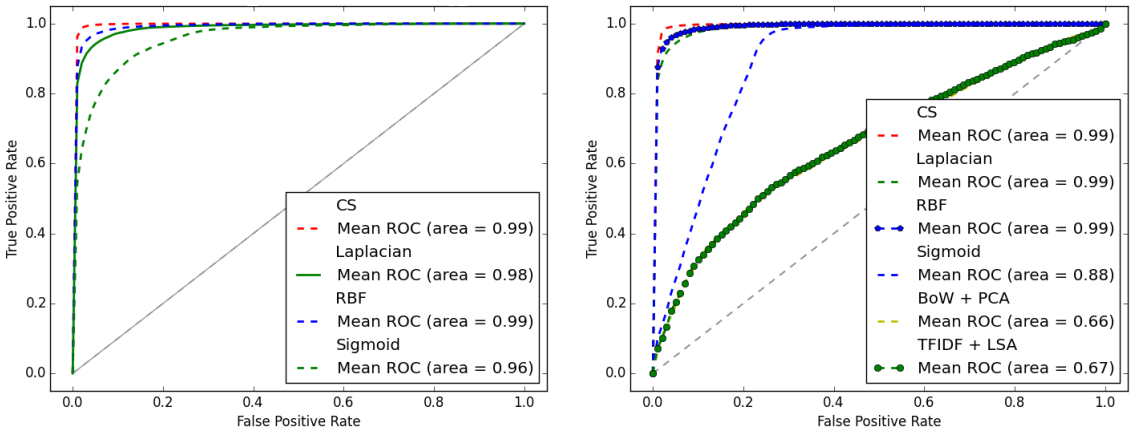
TABLE 5.1: MLP performance on varying feature representations

	PrE (%)	ReC (%)	Fm (%)
DM + DBOW	98.00	98.00	98.00
Sigmoid + Autoencoder	88.00	85.00	86.50
Laplacian + Autoencoder	95.00	95.00	95.00
RBF + Autoencoder	96.00	96.00	96.00
CS + Autoencoder	98.00	98.00	98.00
CS + Stacked Autoencoder	98.00	98.00	98.00

Figure 5.2 - Figure 5.4 presents performance results of the MLP classifier with various feature representations using the ROC curve as a performance measure. However, Figure 5.4 which makes a comparison of the MLP and SVM, has shown that SVM has a slightly better performance in most cases. The feature representations stated are then compared to other traditional unsupervised feature representation which are:

- BoW + PCA
- TF-IDF + LSA

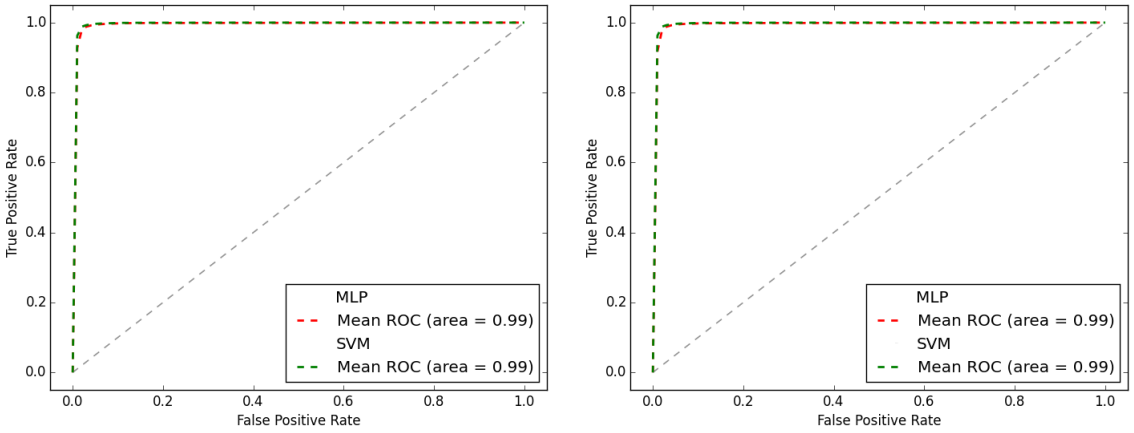
In Figure 5.2a DM + DBOW and CS + Autoencoder for feature representation is compared with the unsupervised feature learning method DM + DBOW. These feature



(A) MLP performance with varying approaches for feature transformation.

(B) MLP performance with traditional feature representation and proposed feature representation.

FIGURE 5.3: Comparison of the traditional feature representations and the proposed feature representations on MLP.



(A) Comparison of MLP and SVM performance when Autoencoder is employed for feature reduction.

(B) Comparison of MLP and SVM performance when Stacked Autoencoder is employed for feature reduction.

FIGURE 5.4: Comparison of the traditional feature representations and the proposed feature representations on MLP.

representation approaches have similar impact towards the performance of MLP with varying threshold function.

Figure 5.2b makes a comparison of Autoencoder and Stacked Autoencoder for feature reduction. Note that features on which these feature reduction techniques where employed, were generated using DM + DBOW and CS measure. Looking at the ROC curve we notice that both Autoencoder and Stacked Autoencoder leads to the same performance.

Figure 5.3a compares the CS measure with kernel functions such as Laplacian, Sigmoid and RBF for feature transformation. Same approach as in Figure 5.2b (*i.e.*, DM + DBOW and CS + Autoencoder) is used but CS measure being replaced by various kernel functions. However, it is clear that CS measure ensures that better feature representation is always generated. Figure 5.3b compares feature representations in Figure 5.3a with other traditional feature representations. TF-IDF + LSA and BoW + PCA fail to retain important features for identifying spam emails in this case.

Figure 5.4a make a comparison of SVM and MLP when Autoencoder is employed for feature reduction. Figure 5.4b make a comparison of SVM and MLP when Stacked Autoencoder is used for feature reduction. Both SVM and MLP according to ROC curve have a similar performance with variation of threshold function.

5.5 Conclusions

This chapter has examined the performance of the neural network approach for identifying spam emails. In this study, a DM + DBOW model was employed for generating a dense representation of email documents. The CS measure was also employed for generating feature representation by ensuring that similar documents are much closer in terms of representation. Autoencoder was then employed to reduce the feature space into a more robust feature representation that is more meaningful to classifiers. CS measure for feature transformation was compared to the kernel functions such as RBF, Sigmoid, and Laplacian. Furthermore, Stacked Autoencoder has been introduced for comparison with the basic Autoencoder for feature reduction. These proposed feature construction approaches have shown different impacts towards classifiers performance for identifying spam emails from non-spam emails. The following feature representations

- DM + DBOW and CS + Autoencoder
- DM + DBOW
- DM + DBOW and CS + Stacked Autoencoder

have led the MLP classifier to a better performance. However, the limitations we encountered in this study is that for DM + DBOW and CS + Stacked Autoencoder approach there are still many investigations which need to be conducted. This includes an optimal number of layers for training the Stacked Autoencoder, each layer requires an optimal number of neurons. These encountered limitations are some of the properties that could not be investigated due to memory issues we encountered. This proposed model needs

to be evaluated on other datasets as well. Due to time constraints, this study could not evaluate the model on the other benchmarking datasets for email spam detection problems. That will be important to show the reliability of this model.

Chapter 6

Conclusion

In this study, the main objective was to improve the performance of classifiers on spam email detection problem. This study is motivated by the literature which has shown that there are various things which may affect the performance of classifiers. This may include feature representation which is divided into feature extraction and feature reduction. Furthermore, for kernel-based classifiers, it is important to identify optimal hyperparameters for delineating negative and positive examples. The hyperparameter optimization in a spam email related problem has been overlooked in most cases which is one of the reason classifiers such as SVM has not performed well with some certain feature representations.

This study in contrast to previous studies in a spam email related problem area has introduced dense feature representation which captures semantic meaning within words and information regarding the closeness of documents. The study further introduced CS measure to transform the feature representation into a higher feature space which captures correlations of all email documents. The Autoencoder is introduced for feature reduction to capture more robust features for identifying spam emails from legitimate emails.

In addition, Stacked Autoencoder is also introduced for feature reduction for a further comparison with Autoencoder. CS measure is compared to distance based kernel functions such as Sigmoid, RBF, and Laplacian for feature transformation.

The uniqueness of this study is found in the way features are extracted for identifying spam emails. The proposed feature extraction approach extracts meaningful features to classifiers in a way that even in a lower feature space more robust features are captured for identifying spam emails. The proposed feature extraction approach is not limited to

spam email problem only. However, it may be used for other natural language processing problems.

In contrast to previous studies where the stopwords have been regarded as noisy words not significant for identifying spam emails, this study shows that the stopwords may be helpful for generating meaningful features when word sequence is taken into consideration. This has been clearly shown that when stemming and stopwords removal were employed before feature extraction phase, the performance of classifiers degraded as compared to when they are used to form part of the feature learning phase.

The proposed study has shown that word count based methods may make classifiers to be inconsistent with different datasets for feature construction. This has been shown with methods such as BoW + IG, BoW + χ^2, TF-IDF + IG, TF-IDF + χ^2, BoW + PCA, BoW + LSA and TF-IDF + PCA and TF-IDF + LSA. These mentioned approaches for feature representation were not reliable with a variation of datasets.

Supervised feature reduction techniques such as χ^2 and IG have shown to lose more robust features in a lower dimensional feature representation. Unsupervised models such as PCA and LSA in some cases fail to capture more robust features in a lower dimensional feature space. However, they only ensured a better performance with the Enron dataset, but a significantly poor performance with the Trec07 dataset.

The DM + DBOW and CS + Autoencoder approach has shown to be more effective for feature extraction. Classifiers in most cases have shown a significant improvement on classifiers. DM + DBOW and CS + Stacked Autoencoder approach also has shown to have promising performance on MLP. Although further investigation on its reliability is required. However, in overall DM + DBOW and CS + Autoencoder has shown to be reliable based on a dataset which was taken into consideration for performance evaluation.

Future Work

For future work we need to consider deep learning techniques which will be able to handle noisy dataset. This is because handling more noisy dataset is still a question to the researchers. We want to learn a model which will generate feature representation which learns from the character level to word level and to document as a whole.

The model that needs to be invented will be able to identify noisy words. By being able to understand the character combination of words and the probable noise which might occur we enable the learning models to map noisy words into their formal representation.

We will not only map those words but also consider word sequence to find meaning out of the whole document.

For hyperparameter optimization we need to consider other optimization methods which will overcome the issue of time complexity. Because grid search algorithm is computationally expensive.

Other feature reduction techniques such as PCA and LSA needs to be investigated how they impact classifiers if they are utilized for feature reduction on features extracted using DM + DBOW and CS.

Since the problem of spam is not only limited to content, for future work we also need to generate a model which will also handle email attachments. We need to invent models which can inspect the attachments of emails. Sometimes spam emails may contain images which are for spam purposes. We need to invent models which will determine if the attachments of an email are for spam purposes or not. Also we need to learn models to differentiate between malicious codes and non-malicious codes in the email attachments.

Bibliography

Abdi, H. & Williams, L. J. (2010), 'Principal component analysis', *Wiley Interdisciplinary Reviews: Computational Statistics* **2**(4), 433–459.

Agarwal, S., Godbole, S., Punjani, D. & Roy, S. (2007), How much noise is too much: A study in automatic text classification, *in* 'Seventh IEEE International Conference on Data Mining (ICDM 2007)', IEEE, pp. 3–12.

Agre, G. & Dzhondzhorov, A. (2016), 'A weighted feature selection method for instance-based classification', pp. 14–25.

Alpaydin, E. (2014), *Introduction to machine learning*, MIT press.

Amayri, O. & Bouguila, N. (2010), 'A study of spam filtering using support vector machines', *Artificial Intelligence Review* **34**(1), 73–108.

Arauzo-Azofra, A., Benitez, J. M. & Castro, J. L. (2004), 'A feature set measure based on relief', pp. 104–109.

Baldi, P. (2012), 'Autoencoders, unsupervised learning, and deep architectures', *Unsupervised and Transfer Learning Challenges in Machine Learning, Volume 7* p. 43.

Barford, P. & Yegneswaran, V. (2007), An inside look at botnets, *in* 'Malware detection', Springer, pp. 171–191.

Basavaraju, M. & Prabhakar, D. R. (2010), 'A novel method of spam mail detection using text based clustering approach', *International Journal of Computer Applications* **5**(4), 15–25.

Bebis, G. & Georgiopoulos, M. (1994), 'Feed-forward neural networks', *IEEE Potentials* **13**(4), 27–31.

Beleites, C., Neugebauer, U., Bocklitz, T., Krafft, C. & Popp, J. (2013), 'Sample size planning for classification models', *Analytica chimica acta* **760**, 25–33.

Belhumeur, P. N., Hespanha, J. P. & Kriegman, D. J. (1997), 'Eigenfaces vs. fisherfaces: Recognition using class specific linear projection', *Pattern Analysis and Machine Intelligence, IEEE Transactions on* **19**(7), 711–720.

Bo Yu, Z.-b. X. (2008), 'A comparative study for content-based dynamic spam classification using four machine learning algorithms', *Knowledge-Based Systems* pp. 355–362.

Breiman, L., Friedman, J., Stone, C. J. & Olshen, R. A. (1984), *Classification and regression trees*, CRC press.

Brereton, R. G. & Lloyd, G. R. (2014), 'Partial least squares discriminant analysis: taking the magic away', *Journal of Chemometrics* **28**(4), 213–225.

Cai, D., Chang, L. & Ji, D. (2012), 'Latent semantic analysis based on space integration', **3**, 1430–1434.

Calderbank, R., Jafarpour, S. & Schapire, R. (2009), 'Compressed learning: Universal sparse dimensionality reduction and learning in the measurement domain', *Manuscript* .

Chen, Y.-H. & Lin, T.-C. (2014), 'Dimension reduction techniques for accessing chinese readability', **1**, 434–438.

Cheng, Y., Zhang, R., Wang, X. & Chen, Q. (2008), Text feature extraction based on rough set, *in* 'Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on', Vol. 2, IEEE, pp. 310–314.

Chollet, F. (2015), 'keras', https://github.com/fchollet/keras.

Ciresan, D. C., Meier, U., Gambardella, L. M. & Schmidhuber, J. (2011), 'Convolutional neural network committees for handwritten character classification', pp. 1135–1139.

Clark, J., Koprinska, I. & Poon, J. (2003), A neural network based approach to automated e-mail classification, *in* 'null', IEEE, p. 702.

Cormack, G. V. (2007), Trec 2007 spam track overview, *in* 'In Proc of TREC 2007: The 16th text retrieval conf.'.

Costanza, M. C. & Afifi, A. (1979), 'Comparison of stopping rules in forward stepwise discriminant analysis', *Journal of the American Statistical Association* **74**(368), 777–785.

Dash, M. & Liu, H. (1997), 'Feature selection for classification', *Intelligent data analysis* **1**(1-4), 131–156.

Desai, M. B., Patel, S. & Prajapati, B. (2016), 'Anova and fisher criterion based feature selection for lower dimensional universal image steganalysis', *International Journal of Image Processing (IJIP)* **10**(3), 145.

Dhinakaran, C., Lee, J. K. & Lee, J. K. (2007), An empirical study of spam and spam vulnerable email accounts, *in* 'Future generation communication and networking (fgcn 2007)', Vol. 1, IEEE, pp. 408–413.

Diale, M., Van Der Walt, C., Celik, T. & Modupe, A. (2016), 'Feature selection and support vector machine hyper-parameter optimisation for spam detection', pp. 1–7.

Ergin, S. & Isik, S. (2014), The assessment of feature selection methods on agglutinative language for spam email detection: A special case for turkish, *in* 'Innovations in Intelligent Systems and Applications (INISTA) Proceedings, 2014 IEEE International Symposium on', IEEE, pp. 122–125.

Ferreira, A. & Figueiredo, M. (2011), Efficient unsupervised feature selection for sparse data, *in* 'EUROCON-International Conference on Computer as a Tool (EUROCON), 2011 IEEE', IEEE, pp. 1–4.

Fusilier, D. H., Montes-y Gómez, M., Rosso, P. & Cabrera, R. G. (2015), Detection of opinion spam with character n-grams, *in* 'International Conference on Intelligent Text Processing and Computational Linguistics', Springer, pp. 285–294.

Gansterer, W. N. & Pölz, D. (2009), 'E-mail classification for phishing defense', pp. 449–460.

Garner, S. R. et al. (1995), Weka: The waikato environment for knowledge analysis, *in* 'Proceedings of the New Zealand computer science research students conference', Citeseer, pp. 57–64.

Goodman, E., Ingram, J., Martin, S. & Grunwald, D. (2015), Using bipartite anomaly features for cyber security applications, *in* '2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)', IEEE, pp. 301–306.

Grimes, G. A., Hough, M. G. & Signorella, M. L. (2007), 'Email end users and spam: relations of gender and age group to attitudes and actions', *Computers in Human Behavior* **23**(1), 318–332.

Gromski, P. S., Muhamadali, H., Ellis, D. I., Xu, Y., Correa, E., Turner, M. L. & Goodacre, R. (2015), 'A tutorial review: Metabolomics and partial least squares-discriminant analysis–a marriage of convenience or a shotgun wedding', *Analytica chimica acta* **879**, 10–23.

Günal, S. (2012), 'Hybrid feature selection for text classification', *Turkish Journal of Electrical Engineering & Computer Sciences* **20**(Sup. 2), 1296–1311.

Guzella, T. S. & Caminhas, W. M. (2009), 'A review of machine learning approaches to spam filtering', *Expert Systems with Applications* **36**(7), 10206–10222.

Halder, S., Tiwari, R. & Sprague, A. (2011), Information extraction from spam emails using stylistic and semantic features to identify spammers, *in* 'Information Reuse and Integration (IRI), 2011 IEEE International Conference on', IEEE, pp. 104–107.

Hayati, P. & Potdar, V. (2008), Evaluation of spam detection and prevention frameworks for email and image spam: a state of art, *in* 'Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services', ACM, pp. 520–527.

Heron, S. (2009), 'Technologies for spam detection', *Network Security* **2009**(1), 11–15.

Hsu, C.-W., Chang, C.-C., Lin, C.-J. et al. (2003), 'A practical guide to support vector classification'.

Jacob, S. G. & Ramani, R. G. (2011), 'Discovery of knowledge patterns in clinical data through data mining algorithms: Multi-class categorization of breast tissue data', *International Journal of Computer Applications (IJCA)* **32**(7), 46–53.

Jin, C., De-Lin, L. & Fen-Xiang, M. (2009), 'An improved id3 decision tree algorithm', pp. 127–130.

Jinzhu, H., Jiangbo, S. & Yuying, H. (2008), Text feature extraction based on the extension of topic words and fuzzy set, *in* 'Computer Science and Software Engineering, 2008 International Conference on', Vol. 1, IEEE, pp. 219–222.

Jivani, A. G. et al. (2011), 'A comparative study of stemming algorithms', *Int. J. Comp. Tech. Appl* **2**(6), 1930–1938.

Joachims, T. (1996), A probabilistic analysis of the rocchio algorithm with tfidf for text categorization., Technical report, DTIC Document.

Karthikeyani, V., Begum, I. P., Tajudin, K. & Begam, I. S. (2012), 'Comparative of data mining classification algorithm (cdmca) in diabetes disease prediction', *International Journal of Computer Applications* **60**(12).

Kira, K. & Rendell, L. A. (1992), 'The feature selection problem: Traditional methods and a new algorithm', **2**, 129–134.

Klimt, B. & Yang, Y. (2004), The enron corpus: A new dataset for email classification research, *in* 'European Conference on Machine Learning', Springer, pp. 217–226.

Kohavi, R. et al. (1995), 'A study of cross-validation and bootstrap for accuracy estimation and model selection', **14**(2), 1137–1145.

Kononenko, I. (1994), 'Estimating attributes: analysis and extensions of relief', pp. 171–182.

Kononenko, I., Robnik-Sikonja, M. & Pompe, U. (1996), 'Relieff for estimation and discretization of attributes in classification, regression, and ilp problems', *Artificial intelligence: methodology, systems, applications* pp. 31–40.

Kononenko, I., Šimec, E. & Robnik-Šikonja, M. (1997), 'Overcoming the myopia of inductive learning algorithms with relieff', *Applied Intelligence* **7**(1), 39–55.

Krawetz, N. (2004), 'Anti-honeypot technology', *IEEE Security & Privacy* **2**(1), 76–79.

Kullback, S. (1997), *Information theory and statistics*, Courier Corporation.

Kumar, R. K., Poonkuzhali, G. & Sudhakar, P. (2012), 'Comparative study on email spam classifier using data mining techniques', **1**, 14–16.

Kumaran, G. & Allan, J. (2004), Text classification and named entities for new event detection, *in* 'Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 297–304.

Le, Q. V. & Mikolov, T. (2014), Distributed representations of sentences and documents., *in* 'ICML', Vol. 14, pp. 1188–1196.

Lee, K., Caverlee, J. & Webb, S. (2010), Uncovering social spammers: social honeypots+ machine learning, *in* 'Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 435–442.

Lee, S. M., Kim, D. S., Kim, J. H. & Park, J. S. (2010), Spam detection using feature selection and parameters optimization, *in* 'Complex, Intelligent and Software Intensive Systems (CISIS), 2010 International Conference on', IEEE, pp. 883–888.

Leopold, E. & Kindermann, J. (2002), 'Text categorization with support vector machines. how to represent texts in input space?', *Machine Learning* **46**(1-3), 423–444.

Li, C.-H., Lin, C.-T., Kuo, B.-C. & Ho, H.-H. (2010), An automatic method for selecting the parameter of the normalized kernel function to support vector machines, *in* '2010 International Conference on Technologies and Applications of Artificial Intelligence', IEEE, pp. 226–232.

Li, Z. & Shen, H. (2011), Soap: A social network aided personalized and effective spam filter to clean your e-mail box, *in* 'INFOCOM, 2011 Proceedings IEEE', IEEE, pp. 1835–1843.

Liu, H., Sun, J., Liu, L. & Zhang, H. (2009), 'Feature selection with dynamic mutual information', *Pattern Recognition* **42**(7), 1330–1339.

Liu, N., Zhang, B., Yan, J., Chen, Z., Liu, W., Bai, F. & Chien, L. (2005), Text representation: From vector to tensor, *in* 'Fifth IEEE International Conference on Data Mining (ICDM'05)', IEEE, pp. 4–pp.

Maas, A. L., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y. & Potts, C. (2011), Learning word vectors for sentiment analysis, *in* 'Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies', Association for Computational Linguistics, Portland, Oregon, USA, pp. 142–150.
URL: *http://www.aclweb.org/anthology/P11-1015*

Maas, A. L., Hannun, A. Y. & Ng, A. Y. (2013), Rectifier nonlinearities improve neural network acoustic models, *in* 'Proc. ICML', Vol. 30.

Manevitz, L. M. & Yousef, M. (2002), 'One-class svms for document classification', *the Journal of machine Learning research* **2**, 139–154.

Masud, M. M., Khan, L. & Thuraisingham, B. (2007), Feature based techniques for auto-detection of novel email worms, *in* 'Pacific-Asia Conference on Knowledge Discovery and Data Mining', Springer, pp. 205–216.

Metsis, V., Androutsopoulos, I. & Paliouras, G. (2006), Spam filtering with naive bayes-which naive bayes?, *in* 'CEAS', pp. 27–28.

Mi, G., Gao, Y. & Tan, Y. (2015), 'Apply stacked auto-encoder to spam detection', pp. 3–15.

Mi, G., Gao, Y. & Tan, Y. (2016), Term space partition based ensemble feature construction for spam detection, *in* 'International Conference on Data Mining and Big Data', Springer, pp. 205–216.

Mikolov, T., Chen, K., Corrado, G. & Dean, J. (2013), 'Efficient estimation of word representations in vector space', *CoRR* **abs/1301.3781**.

Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J. & Khudanpur, S. (2010), 'Recurrent neural network based language model.', **2**, 3.

Mikolov, T., Kombrink, S., Burget, L., Černockỳ, J. & Khudanpur, S. (2011), Extensions of recurrent neural network language model, *in* '2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)', IEEE, pp. 5528–5531.

Mohamad, M. (2015), An evaluation on the efficiency of hybrid feature selection in spam email classification, *in* 'Computer, Communications, and Control Technology (I4CT), 2015 International Conference on', IEEE, pp. 227–231.

Moh'd A Mesleh, A. (2007), 'Chi square feature extraction based svms arabic language text categorization system', *Journal of Computer Science* **3**(6), 430–435.

Nancy, P. & Ramani, D. R. G. (2011), 'A comparison on performance of data mining algorithms in classification of social network data', *International Journal of Computer Applications* **32**(8), 47–54.

Obied, A. (2006), 'Honeypots and spam', *University of Calgary* .

Pandey, U. & Chakravarty, S. (2010), 'A survey on text classification techniques for e-mail filtering', pp. 32–36.

Patidar, R., Sharma, L. et al. (2011), 'Credit card fraud detection using neural network', *International Journal of Soft Computing and Engineering (IJSCE)* **1**(32-38).

Patil, T. R. & Sherekar, S. (2013), 'Performance analysis of naive bayes and j48 classification algorithm for data classification', *International Journal of Computer Science and Applications* **6**(2), 256–261.

Plagianakos, V. & Magoulas, G. (2013), 'Stochastic gradient descent', *Advances in Convex Analysis and Global Optimization: Honoring the Memory of C. Caratheodory (1873–1950)* **54**, 433.

Pradhan, B. (2013), 'A comparative study on the predictive ability of the decision tree, support vector machine and neuro-fuzzy models in landslide susceptibility mapping using gis', *Computers & Geosciences* **51**, 350–365.

Qian, F., Pathak, A., Hu, Y. C., Mao, Z. M. & Xie, Y. (2010), A case for unsupervised-learning-based spam filtering, *in* 'ACM SIGMETRICS Performance Evaluation Review', Vol. 38, ACM, pp. 367–368.

Quinlan, J. R. (2014), *C4. 5: programs for machine learning*, Elsevier.

Sahami, M., Dumais, S., Heckerman, D. & Horvitz, E. (1998), 'A bayesian approach to filtering junk e-mail', **62**, 98–105.

Sebastiani, F. (2002), 'Machine learning in automated text categorization', *ACM computing surveys (CSUR)* **34**(1), 1–47.

Shams, R. & Mercer, R. E. (2013), Classifying spam emails using text and readability features, *in* '2013 IEEE 13th international conference on data mining', IEEE, pp. 657–666.

Sharma, A. K. & Sahni, S. (2011), 'A comparative study of classification algorithms for spam email data analysis', *International Journal on Computer Science and Engineering* **3**(5), 1890–1895.

Silva, C. & Ribeiro, B. (2003), The importance of stop word removal on recall values in text categorization, *in* 'Neural Networks, 2003. Proceedings of the International Joint Conference on', Vol. 3, IEEE, pp. 1661–1666.

Singh, S. & Gupta, P. (2014), 'Comparative study id3, cart and c4. 5 decision tree algorithm: a survey', *Int J Adv Inf Sci Technol [Internet]* **27**, 97–103.

Song, J., Lee, S. & Kim, J. (2011), Spam filtering in twitter using sender-receiver relationship, *in* 'International Workshop on Recent Advances in Intrusion Detection', Springer, pp. 301–317.

Song, Y., Kołcz, A. & Giles, C. L. (2009), 'Better naive bayes classification for high-precision spam detection', *Software: Practice and Experience* **39**(11), 1003–1024.

Sterne, J. & Priore, A. (2000), *Email marketing: using email to reach your target audience and build customer relationships*, John Wiley & Sons, Inc.

Tak, G. K. & Tapaswi, S. (2010), Knowledge base compound approach towards spam detection, *in* 'International Conference on Network Security and Applications', Springer, pp. 490–499.

Thompson, B. B., Marks, R. J., Choi, J. J., El-Sharkawi, M. A., Huang, M.-Y. & Bunje, C. (2002), Implicit learning in autoencoder novelty assessment, *in* 'Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on', Vol. 3, IEEE, pp. 2878–2883.

Tipping, M. (2003), 'Relevance vector machine'. US Patent 6,633,857.

Trivedi, S. K. & Dey, S. (2013), An enhanced genetic programming approach for detecting unsolicited emails, *in* 'Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on', IEEE, pp. 1153–1160.

Tseng, C.-Y. & Chen, M.-S. (2009), Incremental svm model for spam detection on dynamic email social networks, *in* 'Computational Science and Engineering, 2009. CSE'09. International Conference on', Vol. 4, IEEE, pp. 128–135.

Ture, M., Tokatli, F. & Kurt, I. (2009), 'Using kaplan–meier analysis together with decision tree methods (c&rt, chaid, quest, c4. 5 and id3) in determining recurrence-free survival of breast cancer patients', *Expert Systems with Applications* **36**(2), 2017–2026.

Turian, J., Ratinov, L. & Bengio, Y. (2010), Word representations: a simple and general method for semi-supervised learning, *in* 'Proceedings of the 48th annual meeting of the association for computational linguistics', Association for Computational Linguistics, pp. 384–394.

Turney, P. D., Pantel, P. et al. (2010), 'From frequency to meaning: Vector space models of semantics', *Journal of artificial intelligence research* **37**(1), 141–188.

Tzikas, D. G., Wei, L., Likas, A., Yang, Y. & Galatsanos, N. P. (2006), 'A tutorial on relevance vector machines for regression and classification with applications', *EURASIP News Letter* **17**(2), 4.

Uğuz, H. (2011), 'A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm', *Knowledge-Based Systems* **24**(7), 1024–1032.

Uysal, A. K. & Gunal, S. (2012), 'A novel probabilistic feature selection method for text classification', *Knowledge-Based Systems* **36**, 226–235.

Von Ahn, L., Blum, M., Hopper, N. J. & Langford, J. (2003), Captcha: Using hard ai problems for security, *in* 'International Conference on the Theory and Applications of Cryptographic Techniques', Springer, pp. 294–311.

Wall, M. E., Rechtsteiner, A. & Rocha, L. M. (2003), Singular value decomposition and principal component analysis, *in* 'A practical approach to microarray data analysis', Springer, pp. 91–109.

Wan, J., Liu, M., Yi, J. & Zhang, X. (2015), Detecting spam webpages through topic and semantics analysis, *in* 'Computer & Information Technology (GSCIT), 2015 Global Summit on', IEEE, pp. 1–7.

Webb, S., Chitti, S. & Pu, C. (2005), An experimental evaluation of spam filter performance and robustness against attack, *in* '2005 International Conference on Collaborative Computing: Networking, Applications and Worksharing', IEEE, pp. 8–pp.

Wright, D. B. (2005), 'Receiver operating characteristics curves', *Encyclopedia of statistics in behavioral science* .

Wu, C.-T., Cheng, K.-T., Zhu, Q. & Wu, Y.-L. (2005), Using visual features for anti-spam filtering, *in* 'Image Processing, 2005. ICIP 2005. IEEE International Conference on', Vol. 3, IEEE, pp. III–509.

Xie, J., Chen, W., Zhang, D., Zu, S. & Chen, Y. (2017), 'Application of principal component analysis in weighted stacking of seismic data', *IEEE Geoscience and Remote Sensing Letters* .

Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G. & Osipkov, I. (2008), 'Spamming botnets: signatures and characteristics', *ACM SIGCOMM Computer Communication Review* **38**(4), 171–182.

Xu, H. & Yu, B. (2010), 'Automatic thesaurus construction for spam filtering using revised back propagation neural network', *Expert Systems with Applications* **37**(1), 18–23.

Xu, S., Li, X. & Lu, W. F. (2016), Randomized kd tree relieff algorithm for feature selection in handling high dimensional process parameter data, *in* 'Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on', IEEE, pp. 1–8.

Yang, Y. (1995), Noise reduction in a statistical approach to text categorization, *in* 'Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval', ACM, pp. 256–263.

Yang, Y. & Pedersen, J. O. (1997), A comparative study on feature selection in text categorization, *in* 'ICML', Vol. 97, pp. 412–420.

Yang, Y. & Wilbur, J. (1996), 'Using corpus statistics to remove redundant words in text categorization', *JASIS* **47**(5), 357–369.

Youn, S. & McLeod, D. (2007), 'A comparative study for email classification', pp. 387–391.

Zeng, X.-Q., Li, G.-Z., Wu, G.-F. & Zou, H.-X. (2007), 'On the number of partial least squares components in dimension reduction for tumor classification', pp. 206–217.

Zhang, H. & Li, D. (2007), Naive bayes text classifier, *in* 'Granular Computing, 2007. GRC 2007. IEEE International Conference on', IEEE, pp. 708–708.

Zhang, W., Yoshida, T. & Tang, X. (2011), 'A comparative study of tf* idf, lsi and multi-words for text classification', *Expert Systems with Applications* **38**(3), 2758–2765.