Effective Features and Machine Learning Methods for Document Classification

Maysa I Abdulhussain Almulla Khalaf

Thesis submitted for the degree of Doctor of Philosophy

School of Computer Science and Electronic Engineering

University of Essex

July 2019

I would like to dedicate this thesis to my loving husband Tuhsin Al-Jebur
for his endless love, help and support

Abstract

Document classification has been involved in a variety of applications, such as phishing and fraud detection, news categorisation, and information retrieval. This thesis aims to provide novel solutions to several important problems presented by document classification. First, an improved Principal Components Analysis (PCA), based on similarity and correlation criteria instead of covariance, is proposed, which aims to capture low-dimensional feature subset that facilitates improved performance in text classification. The experimental results have demonstrated the advantages and usefulness of the proposed method for text classification in high-dimensional feature space in terms of the number of features required to achieve the best classification accuracy. Second, two hybrid feature-subset selection methods are proposed based on the combination (via either union or intersection) of the results of both supervised (in one method) and unsupervised (in the other method) filter approaches prior to the use of a wrapper, leading to low-dimensional feature subset that can achieve both high classification accuracy and good interpretability, and spend less processing time than most current methods. The experimental results have demonstrated the effectiveness of the proposed methods for feature subset selection in high-dimensional feature space in terms of the number of selected features and the processing time spent to achieve the best classification accuracy. Third, a class-specific (supervised) pre-trained approach based on a sparse autoencoder is proposed for acquiring low-dimensional interesting structure of relevant features, which can be used for high-performance document classification. The experimental results have demonstrated the merit of this proposed method for document classification in high-dimensional feature

space, in terms of the limited number of features required to achieve good classification accuracy. Finally, deep classifier structures associated with a stacked autoencoder (SAE) for higher-level feature extraction are investigated, aiming to overcome the difficulties experienced in training deep neural networks with limited training data in high-dimensional feature space, such as overfitting and vanishing/exploding gradients. This investigation has resulted in a three-stage learning algorithm for training deep neural networks. In comparison with support vector machines (SVMs) combined with SAE and Deep Multilayer Perceptron (DMLP) with random weight initialisation, the experimental results have shown the advantages and effectiveness of the proposed three-stage learning algorithm.

Acknowledgement

First of all, I would like to thank ALLAH Almighty for giving me the strength and determination to complete this work. The last four years have been miraculous for me due to the mercy of ALLAH Almighty; I don't have the words to express my gratitude.

Without any doubt, this thesis will be incomplete without the great supervision and support of Prof. John Q. Gan. John is hugely responsible for transforming my outlook as a researcher and always motivated me to achieve high research standards. I wish to sincerely thank him for his patience, understanding and support.

I would like to sincerely thank my supervisory panel members: Dr. Martin J. Reed and Dr. Luca Citi for their helpful suggestions.

I really appreciate the never-ending support of my parents. Whatever I am today is due to their determined efforts. I cannot pay them back for all their love, care and prayers.

My deepest gratitude goes to my family, my husband Tuhsin and my children Radhwan, Rawan, and little boy Ali the shine and joy of my life, without their love and support this work would have never happened.

I would also like to acknowledge the support of my brother Basil, my sister Shaima, and their families.

I am also grateful to the Ministry of Higher Education and Scientific Research (MOHESR) in Iraq for offering me a scholarship, which helps me undertake my Ph.D. study, and the Computer Science Department at Baghdad University for supporting me through the course of my Ph.D. research.

I would like to thank all of my uncles, aunts, cousins, friends and colleagues in Iraq and in the UK.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Data science has attracted widespread attention in academic and business circles in recent years, and many real-world problems associated with text data have been studied in the context of natural language processing (NLP). Examples of such problems include learning representation and text categorization for structured information extraction (Arras et al., 2017). Data mining or knowledge discovery has played a big role in data science, which can be described as the management, analysis, and extraction of useful information from and the use of the resources inherent in data (Roiger, 2017; Zins, 2007). Document representation can be defined as a process that extracts the terms used (words and phrases) from the text of documents and weights each term in order to indicate its importance within the document (Cobern & Loving, 2001). The representation of a set of documents as a set of vectors in a common vector space is known as the vector space model (VSM) and is fundamental to scoring documents for document classification (Abdulhussain & Gan, 2016). Learning feature representation is a very important pre-processing step in pattern recognition in high-dimensional feature space (Chandrasekaran et al., 2006; Lewis et al., 2004; Ng, 2011; Roiger, 2017). Text or document classification has been involved in a variety of applications such as phishing detection, news categorisation, and information retrieval. The main dilemma in text categorisation (TC) is how to reduce the potentially high feature dimensionality by looking for the best combination of features (feature

selection or dimensionality reduction) that can achieve high classification accuracy. The interpretability of the results of data mining is also an important issue and is relevant in many fields, such as social sciences and medicine (García et al., 2009; Schielzeth, 2010). The dimensionality reduction achieved by feature selection allows for significant improvements in the accuracy and interpretability of classifiers which are built for applications where the number of features is overwhelming compared to the number of obtainable training patterns (Martin-Smith et al., 2016). Many conventional methods aimed at dimensionality reduction can lead to highly accurate data classification results which, however, are often difficult to interpret. Recently, most state-of-the-art research on text classification has focused on introducing various machine learning methods, including deep learning, rather than on discussing specific features of text documents relevant to particular classification tasks (Abdulhussain & Gan, 2016; Sebastiani, 2002). The overfitting problem is one of the leading issues in machine learning, deep learning in particular (Goodfellow et al., 2016; Hawkins, 2004). However, the solutions to the issues mentioned above have not fully answered the main challenges related to the various aspects of big data analysis and classification.

1.2 Research Objectives

This Ph.D. thesis focuses on data/text analysis, via data mining, machine learning and deep learning using supervised and unsupervised methods, in order to achieve dimensionality reduction and subsequently high-performance in terms of document classification accuracy, processing time consumed, and interpretability. There are four main objectives: first, to investigate new methods for feature extraction and

dimensionality reduction for text classification; second, to propose new methods for feature subset selection and interpretable representation learning; third, to explore a class-specific, supervised, pre-trained approach based on deep learning, which is intended to gain a low-dimensional but powerful structure of features with high performance for document classification; and fourth, to investigate deep classifier structures for higher-level feature extraction in order to overcome difficulties that have become apparent when training deep neural networks with limited training data in high-dimensional feature space, such as overfitting and vanishing/exploding gradients.

1.3 Contributions

Four major contributions have been made by this Ph.D. work and can be summarised as follows:

- This research proposes an improved PCA (Principle Component Analysis) approach to feature extraction and dimensionality reduction. The experimental results show that the proposed methods improve the performance of document classification in terms of classification accuracy and the number of extracted features, as compared with the original PCA method.
- Two new hybrid approaches for feature subset selection are proposed, which employ both supervised and unsupervised filters plus a wrapper method. The experimental results demonstrate the advantages of the proposed methods over individual filter and wrapper methods with the complete feature set (full-features without any selection or dimensionality reduction) in terms of classification

accuracy, the number of features required, processing time consumed, and interpretability.

- A new way of pre-training a sparse autoencoder is introduced. This relates to a class specific sparse autoencoder which is pre-trained and learns effective features for document classification. The experimental results demonstrate that the proposed methods have advantages over the standard sparse autoencoder and full-feature approaches in terms of performance as related to the number of required features.
- Deep classifier structures used with a stacked autoencoder (SAE) for higher-level feature extraction are investigated; this approach aims to overcome the difficulties of training deep neural networks with limited training data in high-dimensional feature space, e.g., overfitting and vanishing/exploding gradients. A three-stage learning algorithm is proposed for training a deep multilayer perceptron (DMLP) as the classifier. The first stage consists of unsupervised learning using an SAE to obtain the initial weights for the feature extraction layers of the DMLP. In the second stage, error back-propagation is used to train the DMLP by fixing the weights obtained in the first stage into its feature extraction layers. In the third stage, all the weights of the DMLP obtained at the second stage are refined by error back-propagation. The experimental results demonstrate the advantages and effectiveness of the proposed methods.

Some of the above contributions have been reported in peer-reviewed papers, which are listed in Appendix A.

1.4 Thesis Organization

The remainder of this thesis is structured as follows.

Chapter 2: The literature review presents the approaches that have been used for document representation, feature selection and dimensionality reduction, and classification, including machine learning and deep learning methods for document representation learning and classification. It also introduces evaluation methods and provides an example of applications and assessment of document classification.

Chapter 3: This chapter explains the experimental design used in this thesis. It begins by illustrating the general steps of research methodology and the main tasks that need to be considered to answer the research question. It also describes the work packages, datasets, and experimental procedure.

Chapter 4: It begins with a brief review of the methods used for feature extraction and dimensionality reduction and identifies the limitations of each of these methods. There follows an **"**experimental investigation," leading to an improved PCA method based on calculating cosine similarities and correlations between pairs of features for text feature dimensionality reduction.

Chapter 5: It begins with a brief review of the methods used for feature selection and identifies the limitations of each method. This chapter proposes two hybrid methods for feature subset selection, each consisting of two stages. The first stage selects feature subsets based on the union or intersection of features selected according to distance or similarity measures (unsupervised) and mutual information measures (supervised). The second stage employs a wrapper approach on the selected features to

further reduce the feature dimensionality. Also, this second stage addresses the accuracy-interpretability dilemma, which is an important issue in many fields of research where data mining is used and the methods for dimensionality reduction are unsatisfactory in terms of the interpretability of the features.

Chapter 6: This chapter focuses on deep learning methods for learning feature representation and sheds some light on the limitations of each method. It then proposes a class-specific (supervised) pre-trained approach, which employs a sparse autoencoder algorithm to encode the features of entire training samples (unsupervised). After these processes, all the features yielded are merged into one vector. Classifier fusion is also investigated in this chapter.

Chapter 7: This chapter proposes a three-stage learning algorithm for training DMLP with effective weight initialisations based on SAE, aiming to overcome the difficulties involved in training deep neural networks with limited training data in high-dimensional feature space. Experiments were conducted to evaluate the performance of the proposed method on various datasets, as compared to the performance of existing methods.

Finally, the overall contributions, findings, and limitations of this study plus suggested future work are detailed in **Chapter 8**.

Chapter 2
Literature Review

2.1 Introduction

There are many real-world systems that can produce and/or process massive amounts of data, such as social networks (Cambria, 2016; Scott, 2017), network intrusion detection systems (Bergholz et al., 2008; Almusallam, 2017; Alheeti et al., 2018), information retrieval software (Plansangket, 2017), speech recognition systems, and computer vision systems (Chandrasekaran et al., 2006; Raghupathi, 2014). This type of massive data is often called big data. One of the most popular definitions of big data was proposed by Gartner (2012) "Big data is high-volume, high-velocity, and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision-making." (Walker, 2014; Gandomi & Haider, 2015). Research in knowledge discovery and data mining has seen rapid advances in recent years because of the vast amounts of data which are collected daily. Driving this research is the fact that the ability to analyse such data is a significant requirement. Data mining is the computational process of discovering patterns in large datasets; it involves approaches at the intersection of artificial intelligence (AI), machine learning (ML), statistics, and database systems (Liu, 2009; Russom, 2011; Han et al., 2011; Chen et al., 2014; Buczak & Guven, 2016; Roiger, 2017). There are many other terms which have a similar meaning to data mining, for instance knowledge mining from data, knowledge extraction, and data analysis (Han et al., 2011).

There is a need to use big data across various research areas. We should take advantage of the massive datasets which are available and extract much informative data and valuable information from them. However, the data analysis and knowledge extraction processes involved with big data are very challenging in relation to most of the standard and even advanced data mining and machine learning tools which have been developed (Russom, 2011; Triguero et al., 2015). Also, the use of big data brings specific problems to the fore, such as low efficiency and overfitting in machine learning algorithms. It is clear that many of the features present in massive datasets can be uninformative, irrelevant or redundant (LaValle et al., 2011; Panday et al., 2018). Therefore, it is essential to tackle this over-load problem by employing feature extraction methods, feature selection methods, machine learning, and deep neural networks in order to obtain informative and effective features for tasks like document classification.

This Ph.D. thesis aims to develop new methods for extracting effective features and relevant learning representations by employing machine learning methods including deep learning for document classification. Particularly, this study proposes a modified method for feature extraction and dimensionality reduction, two hybrid methods for the best combination of subset features (feature selection), and a supervised pre-training approach employing a deep learning algorithm. In relation to the latter, it then uses an unsupervised deep learning method to encode the features from training samples. Finally, deep classifier structures based on stacked autoencoder (SAE) are proposed for higher-level feature extraction, aiming at overcoming the difficulties involved in training deep neural networks with limited training data in high-

dimensional feature space. This chapter reviews related concepts and existing methods.

Figure 2.1 illustrates a summary of the topics covered in the literature review chapter.

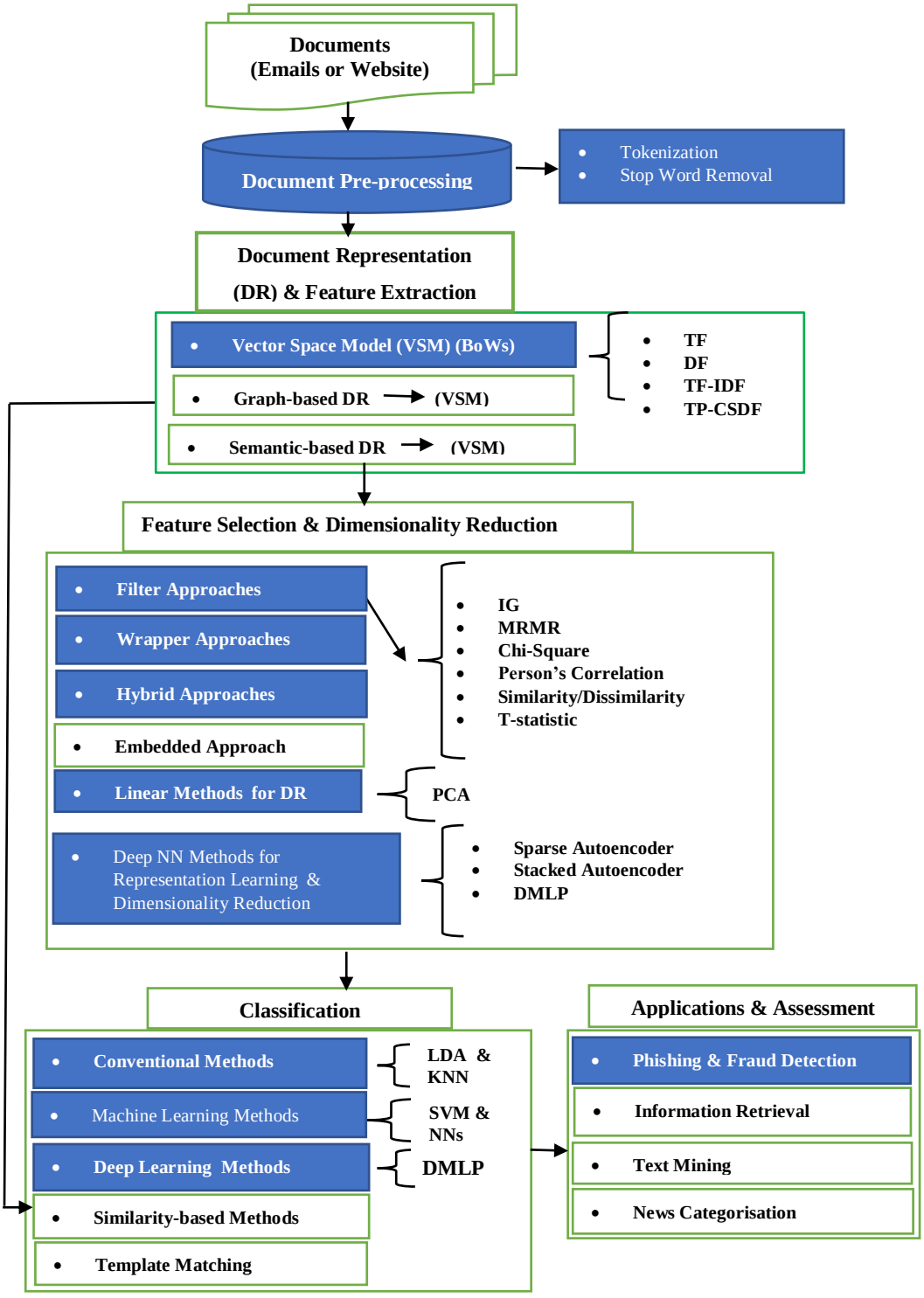

Figure 2.1: Summary of the literature review.

2.2 Data Pre-processing (Document /Text Pre-processing)

Data pre-processing is one of the crucial and time-consuming steps in natural language processing (NLP), knowledge discovery and data mining. This term refers to any process implemented on raw data as an initial data mining practice to transform the data into a numerical format that will be more easily and effectively processed by the next procedure (Jain & Yu, 1998; Todorovski & Džeroski, 2006; Arguello et al., 2008; LaValle et al., 2011; Babaie et al., 2017). For example, the pre-processing steps involved with the natural language processing of texts or documents include tokenisation, stop word removal, and stemming. Tokenization is an essential technique for most NLP tasks. This process splits a sentence or document into tokens; each token will represent a word or phrase (possibly related to the research area in question); some additional knowledge should be taken into consideration when tokenizing, for instance, the names of relevant entities. Also, some stop words such as “the” and “a” will be removed as these words do not provide any worthwhile information (Sun et al., 2017). The general steps for text or document pre-processing are shown in Figure 2.2. In our work we applied tokenization and stop word removal.

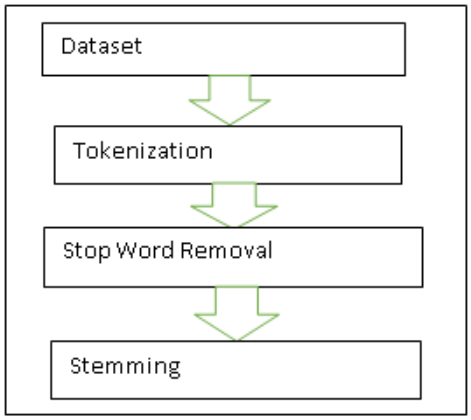

Figure 2.2: Document pre-processing steps.

2.3 Document Representation and Feature Extraction

Document representation is a very important issue in document classification. Each document should be transformed into a machine-readable format to be classified. In general, there are two ways to extract features: manual and automatic. The former is very slow, particularly when the data size is massive. The latter is used for feature extraction through the application of various unsupervised and supervised approaches: such as the methods used in information retrieval (IR), text or data mining, image processing, and natural language processing (Solomatine et al., 2009).

Single word representation is a very common grammatical unit in document representation techniques. There are two major types of document representation using single word: vector space model and graph-based model (Allahyari et al., 2017).

A domain model is a conceptual model involving the concepts of significance to a certain domain as well as the relationships between these concepts (Peng and Choi, 2005). Domain models have been employed and developed in many applications and transformed into many forms such as graph, semantic, and ontology.

This subsection reviews some models and feature extraction methods for document representation and their applications in detection of phishing e-mails and phishing websites (Jiang et al., 2018; Clark et al., 2012).

2.3.1 Vector Space Model (VSM) for Document Representation

The vector space or bag-of-words model is one of the quickest and simplest document representation methods, in which the content of a document is organised as a vector in term space $d(w_1,\ldots,w_k)$, where k is the total number of terms (words or features) (George & Joseph, 2014). This method considers the input data to be an unstructured set of words (the ordering of the terms is ignored) and the frequency of each word as a critical datum (which can easily be used via machine learning for classification purposes). VSMs mostly work with supervised learning algorithms (Liu et al., 2016). There is much research focused on VSMs for pattern recognition, such as detecting phishing e-mails and websites (Ramanathan & Wechsler, 2012; Dobša, 2014; Harish et al., 2014; Mohammad et al., 2014; Plansangket & Gan, 2015b). However, if there are similar meanings within two different documents represented by different sets of words in each case, this method cannot take into account these similarities (Stephens et al., 2004). Also, spelling errors cause an incorrect frequency to be assigned to a word. Thus, it is helpful to perform pre-processing and error detection. Although new VSM methods as extensions to the original VSM (Thanh & Yamada, 2011), such as the Tolerant Rough Set Model (TRSM) and the Similarity Rough Set Model (SRSM), have been developed, this thesis employs bag-of-words for document representation

for the sake of easy comparison between the newly proposed methods and existing methods.

2.3.1.1 Document Frequency (DF)

Document frequency is a simple feature used in document representation and vocabulary reduction. It is based on counting the number of documents, in the training data, in which a term appears (Rogati & Yang, 2002). A predetermined low frequency threshold is applied to decide which terms are removed. It seems that this approach reduces the dimensionality of a feature set representing a document, and is effective if terms with lower frequencies turn out to be those which are irrelevant. However, it can be presumed to be an ineffective method if the remaining terms are not informative, particularly in information retrieval (IR).

2.3.1.2 Term Frequency or Term Strength (TF or TS)

Term frequency or term strength estimates the significance of a term based on how prevalent it is in a document. Wilbur & Sirotkin (1992) proposed and evaluated this measure for use in document representation and vocabulary reduction. Yang & Wilbur (1996) employed it for text categorisation. Guzella & Camin (2009) proposed to use this method for spam filtering, but their method was insufficient in terms of document representation because it depended on this single aspect of the features (term frequency) for spam filtering (Almomani et al., 2013).

2.3.1.3 Term Frequency–Inverse Document Frequency (TF–IDF)

This is a numerical statistical feature used in IR and text mining (Trstenjak et al., 2014). Many studies have applied TF–IDF to extract features from websites and e-mails for detecting phishing e-mails and websites (Stephens et al., 2004; Ramanathan

& Wechsler, 2012; Plansangket & Gan, 2015a; Cong et al. 2017). Equation 2.1 clarifies how TF–IDF works.

$$tfidf_{ji} = \begin{cases} \left(1+\log_2 TF_{ji}\right)\times \log_2 \frac{N}{DF_i}, if \quad TF_{ji} > 0 \\ 0, otherwise \end{cases} \tag{2.1}$$

where $tfidf_{ji}$ is the TF–IDF score of term *i* in document *j*, TF_{ji} is the term frequency of term *i* in document *j*, *N* is the number of documents in the training document set, and DF_i is the number of documents in which term *i* appears in the training document set. Our use of this feature extraction method will be demonstrated in chapters 3, 4, and 5.

2.3.1.4 Term Presence – Class-Specific Document Frequency (TP–CSDF)

Term presence and class-specific document frequency is a new term weighting scheme proposed by Plansangket & Gan (2015a). The score $TP-CSDF_{ji}$ of term *i* in document *j* is calculated as follows:

$$TP-CSDF_{ji} = \frac{TP_{ji}\cdot DF_{ic_j} / N_{c_j}}{(DF_i - DF_{ic_j})/(N - N_{c_j}) + 1} \tag{2.2}$$

where TP_{ji} is term presence whose value is 1 if term *i* is in document *j* and 0 otherwise, DF_{ic_j} is the document frequency of the term *i* based on the documents in the training document set and in the class that document *j* belongs to, DF_i is the document frequency of term *i* based on all the documents in the training document set, N_{c_j} is the number of documents in the training document set in the class that document *j* belongs to, and N is the number of documents in the training document set. This feature

extraction method will be used in chapters 3 and 4 for developing feature selection and dimensionality reduction methods.

2.3.1.5 The Dynamic Markov Chain (DMC) Approach

The Dynamic Markov Chain approach was first proposed by Andrey Markov (Markov, 2008). It is used for arithmetic compression and feature extraction, and it depends on information theory and the probability of the message (i.e., the receipt of the message) in relation to the class to which it belongs. (Andrew, 2000; Levin & Peres, 2017). Frank et al. (2000) proposed to use DMC for text categorisation across different domains. Also, it was used for improving phishing detection in relation to e-mails and websites (Andrew, 2000). This is a text-based method. It is based on the bag-of-words representation which determines the language-use of each class of messages. DMC has the advantage of reducing the amount of memory consumed for text categorisation because the next point depends only on the current point. The mathematical formulation of DMC is as follows:

$$H(x, M) = -\frac{1}{n} \log \prod_{i=1}^{n} P(b_i \mid b_1^{i-1}, M) \tag{2.3}$$

where $H(x, M)$ is the Cross-Entropy (CE) between the message *x* and the model M and $P(b_i \mid b_1^{i-1}, M)$ is the probability of encountering feature b_i, depending on the previous features $b_1 ... b_{i-1}$.

2.3.1.6 Supervised Term Weighting Methods

The notion of term weights is borrowed from the IR field (Debole & Sebastiani, 2003; Lan et al., 2009; Deng et al., 2014). Text categorisation (TC) based on term weighting is usually implemented via supervised learning, which can use information included with the training documents with predefined categories (class labels) (Sergienko et al.,

2014). Term weighting methods assign an appropriate weight to each term, which provide a natural way for feature selection as it is possible to ignore the terms with the lowest weights (Sergienko et al., 2017). The following are various supervised term weighting approaches.

1) Term weighting using information theory functions or statistical metrics: This approach weighs terms by applying feature selection criteria, such as the Chi-statistic, information gain (IG), or gain ratio (GR). The above feature selection criteria are usually combined with term frequency (e.g., $tf.x^2$ or $tf.ig$) to obtain term scores. The terms with higher scores are deemed to be able to contribute more to TC than the terms with lower scores (Debole & Sebastiani, 2003).

2) Term weighting via interaction with a text classifier: The idea of this approach is similar to that of the above. The text classifier discriminates positive test documents from negative ones by assigning different scores to the documents in the test data. Bergholz et al. (2010) applied an iterative approach which included a KNN text classifier at each step. The weights are slightly modified by the results of the text classifier, and then the accuracy is measured by employing another evaluation criterion. The limitation of this method is that it is too slow to be used, especially for large problems (Basnet et al., 2012).

2.3.2 Semantic-based Document Representation

The main issue of the single word representation is the loss of semantic meaning of words. Therefore, a lot of recent research aims to solve this problem by exploiting

semantic features using knowledge-based approaches. For example, WordNet based similarity rough set model (WSSM) (Thanh et al., 2011).

In recent years, acquisition of semantic information is necessary for proper understanding of natural language text. Semantic-based document representation has been investigated in information retrieval (Elhadad et al., 2017). If senses are used to represent a document, the relations between senses play a key role in capturing the ideas in the document. Peng & Choi (2005) proposed to automatically classify documents based on the meaning of words and the relationships between groups of concepts. Their classification algorithm builds on the word structures provided by WordNet that exploits the semantic hierarchy and they developed a corresponding semantic hierarchy classification system.

Probabilistic Latent Semantic Analysis (PLSA) is a simple example of numerical means of semantic-based document representation. It relies on the discovery of topics from a collection of text documents. This method was proposed by Hofmann (1999) for document indexing based on the statistical latent class model. It has been applied in IR, natural language processing, and machine learning by text filtering (Aggarwal et al., 2013; Zhai, 2017). It contributes to many information retrieval applications by improving the representation of texts by representing each document via low-dimensional latent topic vectors and summarisations of search results (Zhai, 2017). This method addresses the problem of single word representations. It also addresses the issues of synonymy and polysemy in phishing emails and websites detection. A great deal of research has aimed at overcoming these problems by exploiting semantic

features using knowledge-based approaches. PLSA is, however, vulnerable to overfitting. Equation 2.4 explains how it works.

$$P(w,d) = P(d)\sum_c P(c \mid d)P(w \mid c) \tag{2.4}$$

where *w* represents words, *d* refers to documents, *c* is the topic, $P(c \mid d)$ and $P(c)$ are initialised to $(1/d)$ and $(1/d)$ respectively and $P(w \mid c)$ is initialised randomly.

2.3.3 Graph-based Document Representation

Graphs are one of the visualisation methods in text mining to measure the influence of terms in documents or texts, using the notion of centrality and other graph theoretical features. A graph used for this purpose can be described as a pair $G=(V,E)$. The elements of V are the nodes of the graph and the elements of E represent its edges (Mihalcea, 2004). A graph is represented visually by drawing a number of dots equivalent to the number of nodes and connecting each pair of dots by a line if they are connected by an edge (Valle & Ozturk, 2011). Such a graph can be used directly for document classification based on graph distance measures. Figure 2.3 shows an example of a typical graph document representation (Phukon, 2012). For machine learning based classifiers, it is easy, in principle, to convert graphs into vectors representing various graph measures. However, the computational complexity implied by this method is very high (Marko et al., 2008; Phukon, 2012). Nevertheless, a large number of published studies have applied this method (Mihalcea, 2004). Mihalcea & Tarau (2004) studied graph-based representations in their Text-Rank system, which is mainly used for text summarisation. There are various criteria which can be used to measure the centrality of terms in each sentence. These are based on degree, closeness,

betweenness, and eigenvector construction (Faust & Wasserman, 1992). After centrality values are computed, documents are represented as vectors (Freeman, 1979). Eberle and Holder (2007) proposed graph-based approaches for discovering abnormalities in domains consisting of unexpected relationship variations that resemble normal behaviour. They applied graph-based algorithms to intrusion detection datasets. They concluded that their method could discover the small differences between normal and abnormal graphs. However, their method could not recognise abnormalities in complex graphs. Rosiello et al. (2007) studied graph-based representation in order to implement their "DOMAntiPhish" system which was aimed at discriminating between malicious and benign web pages. The limitation of their work was that it is possible for attackers to use similar text and images to those presented by legitimate websites in order to create mimic websites; this makes it difficult to distinguish between malicious and benign websites.

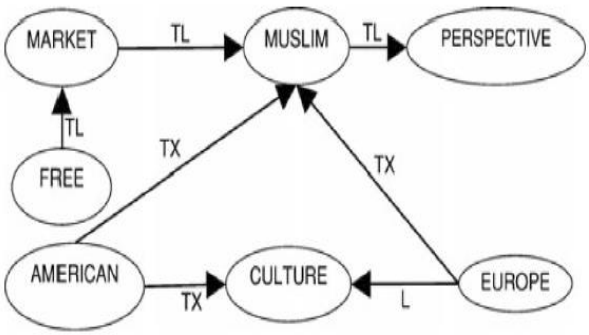

Figure 2.3: An example of standard graph document representation (Phukon, 2012).

2.4 Feature Selection and Dimensionality Reduction

In this 21st century, the number of samples or observations and the number of features available for data mining have increased significantly across many different

applications, such as text categorisation (Yang & Pedersen, 1997; Xing & Karp, 2001; Aldehim & Wang, 2017; Zheng et al., 2018), image retrieval (Rui et al., 1999; Xing & Karp, 2001; Liu & Motoda, 2007; Hira et al., 2015), speech recognition, computer vision, and intrusion detection (Almusallam, 2017). Using all the possible features which can be extracted from all the samples or observations is impractical and might not even be useful for enhancing the accuracy of classification. Therefore, by reducing the dimensionality of the input patterns, it is possible to limit computational complexity and remove irrelevant/redundant features that would make it more difficult to train the classifier; dimensionality reduction addresses the issues which are associated with high dimensionality (Gan et al., 2016). Meng et al. (2011a) proposed a two-stage feature selection algorithm for text categorisation. First, features are selected by using a novel feature selection method, named the feature contribution degree (FCD), which tends to construct a reductive feature vector space. Second, LSI (Latent semantic indexing) is applied to construct a new conceptual vector space on the basis of the reductive feature vector space, where LSI can greatly reduce the dimension and can discover the important associative relationship between terms. Then, the feature-based method and the semantic-based method were combined to reduce the vector space. Feature selection is applied to reduce the number of featuress by removing irrelevant, noise-dominated or redundant features or by choosing a relevant subset of features (Kwak & Choi, 2002; Liu & Motoda, 2007; Nemati et al., 2009; Ortega et al., 2016). An abstraction of the feature selection process as a four-step procedure is shown in Figure 2.4. There are three types of methods which are used for feature selection (Liu et al., 2002): filter methods, wrapper methods, and embedded methods. Filter methods are applied to select features independently of the classification

approach (Guyon & Elisseeff, 2003). Wrapper methods are based on performing classification with a specific classifier to rank possible feature combinations (Nemati et al., 2009; Gan et al., 2014). Embedded methods can be included within the classification procedure. Liu & Motoda (2007) applied feature selection methods in order to rank the features relevant to phishing detection. There are considerable studies that address various issues in feature selection, which are reviewed in the following subsections respectively.

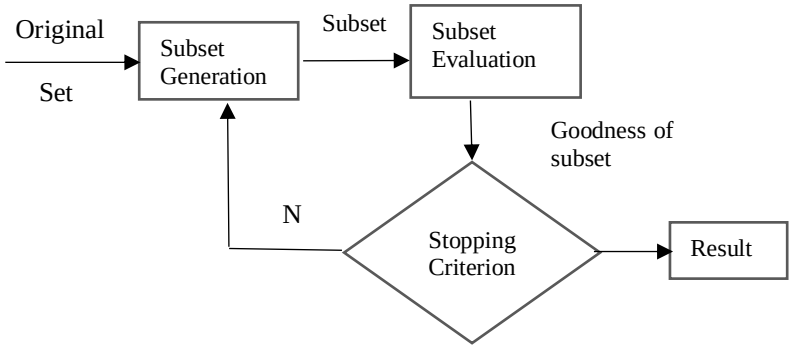

Figure 2.4: Feature selection steps with validation (Dash & Liu, 1997)

2.4.1 Filter Approach

Filter methods have been widely used for selecting a subset of features or for ranking the features of training data based on general characteristics, independently (without including any classification methods). The features with the lowest rankings are generally eliminated, and the filter method need be performed only once for a given task. Thus, filter methods can be easily scaled to very high-dimensional datasets, are computationally simple and fast, and are independent of the classification algorithm (Martin-Smith et al., 2016).

A common drawback of filter approaches is that they ignore the interaction with the classifier. This means that each feature is considered separately, thereby ignoring feature dependencies; this may lead to worse classification performance than other types of feature selection techniques (Saeys et al., 2007). One possible solution for overcoming this issue, of ignoring feature dependencies, is to use multivariate filter techniques, aiming at incorporating feature dependencies to some degree.

Search based methods (Aggarwal & Zhai, 2007) can be utilised to find the best subset of features for classification, which select features using evaluation criteria to measure the accuracy of the subset of features. Martin-Smith et al. (2016) proposed and evaluated a filter approach for evolutionary multi-objective feature selection for classification problems with a large number of features. This thesis applied various evaluation criteria in its filter approach, which are described further in chapter 5. The following is a summary of various evaluation criteria applied in supervised or unsupervised filter approaches.

2.4.1.1 Information Gain (IG) and Mutual Information (MI)

Information Gain is a measure based on the concept of entropy. It can be used to determine the similarities between subsets of any dataset and then assign the highest weight or ranking to the most significant features (Yang & Pedersen, 1997). For example, in e-mail and website phishing detection applications, the IG method has been applied for feature selection to find out how well a subset of features differentiates classes such as legitimate as opposed to phishing e-mails and websites (Chandrasekaran et al., 2006). However, this measure depends only on the probability distributions of random variables rather than on their values.

The information gain of a given feature *X* with respect to a class feature *Y* is the reduction in uncertainty about the value of *Y* when the value of *X* is known. The uncertainty about the value of *Y* is measured by entropy *H*(*Y*). The uncertainty about *Y*, when the value of *X* is known, is yielded by the conditional probability of *Y*, given *X*, *H(Y|X)*. The information gain is defined as

$$I(Y,X) = H(Y) - H(Y \mid X) \tag{2.5}$$

where *Y* and *X* are discrete variables that take values in { $y_{1,\ldots,}y_k$ } and { $x_{1,\ldots,}x_k$ }, the entropy of *Y* is yielded by:

$$H(Y) = -\sum_{i=1}^{k} P(Y_i)\log_2 P(Y_i) \tag{2.6}$$

and the conditional entropy of *Y*, given *X*, is:

$$H(Y \mid X) = -\sum_{j=1}^{l} P(X_j)H(Y \mid X_j) \tag{2.7}$$

Alternatively, the information gain is defined by:

$$I(Y,X) = H(X) + H(Y) - H(X,Y) \tag{2.8}$$

where $H(X,Y)$ is the joint entropy of X and Y:

$$H(X,Y) = -\sum_{i=1}^{k}\sum_{j=1}^{l} P(X_j,Y_i)\log_2 P(X_j,Y_i) \tag{2.9}$$

When the predictive variable, *X*, is not discrete but continuous, the information gain of *X* with respect to class feature *Y* is computed by considering all possible binary features, X_θ, that arise from *X* when we choose a threshold θ on *X* (Kalbhor et al., 2013).θ may take a value from any of the values of *X*. Then the information gain is simply:

$$I(Y,X) = \arg\max x_\theta \; I(Y, X_\theta) \tag{2.10}$$

The information gain measure tends to select features which have a large number of values, which is the major drawback of this measure. (Altidor et al., 2011; Karegowda et al., 2010).

Mutual information is also a measure based on probability theory and information theory. It determines the similarity between the pair of points *p(x,y)* (Tapia & Flores, 2013). This method has been widely used for feature selection in relation to various purposes to reduce the dimensionality of features. Koller et al. (2007) proposed applying information theory to feature selection. The following equation clarifies how it works.

$$MI(x,y) = \sum_{i,j} p(x_i, y_i) \log \frac{p(x_i, y_i)}{p(x_i)p(y_i)} \tag{2.11}$$

where MI refers to the similarity between two features, *x* and *y*, which is defined according to their joint probabilistic distribution *p(x,y)* and their marginal probabilities *p(x)* and *p(y)*. It can be seen that information gain is a special case of mutual information.

2.4.1.2 Minimum Redundancy and Maximum Relevance (mRMR)

Minimum redundancy and maximum relevance is a maximal statistical dependency criterion based on MI. Peng et al. (2005) proposed this two-stage feature selection method for selecting a compact set of relevant features, and they applied it to a number of different datasets (handwritten digits, arrhythmia, NCI cancer cell lines and lymphoma).

Suppose *m*-1 features have already been selected from the available set of features *X*, forming a selected feature subset S_{m-1}. In order to select the next best feature, mRMR is calculated as follows (Peng et al., 2005):

$$\max_{x_j \in X - S_{m-1}} [I(x_j, c) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} I(x_j, x_i)] \tag{2.12}$$

where *c* represents class label, *I(x , y)* is the mutual information function defined in terms of the joint probability of *x* and *y* and their marginal probabilities, as follows:

$$I(x, y) = \iint p(x, y) \log \frac{p(x, y)}{p(x)p(y)} dxdy \tag{2.13}$$

By maximising the mRMR value, the method chooses the feature that has the maximum relevance to the class label and the minimum redundancy in relation to previously selected features. Gan et al. (2014) employed this method in their experiment testing their filter-dominating hybrid sequential forward floating search method for feature subset selection in high-dimensional space. This approach is adopted in chapter 5 for comparative studies.

2.4.1.3 Pearson's Correlation

Pearson's correlation is one of the more straightforward filtering methods. It exploits the inter-correlation of one feature with another in order to combine them into subsets of features. The goal is to improve classification performance and reduce the feature dimension. It evaluates the effectiveness of subsets by measuring the dependents of each feature according to the degree of redundancy between them (Basnet et al., 2012). Correlation is a linear criterion used to compute interconnection between nominal features. However, it does not depend on any particular data transformation. A

correlation coefficient is a number between -1 and 1 that refers to the strength of correlation (connection) between features (Guyon & Elisseeff, 2003). Xu et al. (2017) investigated mRMR based on Pearson's correlation for semi-supervised feature selection. Chapter 4 also uses this criterion for developing an improved PCA method.

The Pearson's correlation, for measuring the strength of a linear association between two variables *x* and *y*, is defined as follows:

$$R(\mathbf{x},\mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.14}$$

where $\bar{x}$ is the mean of x, and $\bar{y}$ is the mean of *y*.

2.4.1.4 Chi-Square (X^2- Statistic)

Chi-square is a statistical test criterion used for comparing the distribution of a categorical variable in a sample with the distribution of a categorical variable in another sample (Yu & Liu, 2003). It is not just a metric but a statistical test, which, in this case, can be used to evaluate a value for the chi-squared statistic with respect to the class, using "feature x is independent of the class" as the null hypothesis. Thaseen & Kumar (2017) employed the chi-square as a feature selection criterion for intrusion detection. The following equation explains how it works:

$$X^2 = \frac{(O - E)^2}{E} \tag{2.15}$$

where X^2 is the chi-square, *O* is the observed frequency of a feature in a category, and E is the expected frequency of the feature in the corresponding category (which is the "degree of freedom").

2.4.1.5 T-statistic

The T-statistic is an effective feature selection method which is based on estimating whether the means of two groups are different from each other (Loo et al., 2005). This statistical measure has been used to calculate the strength of association between variables, which is often used to perform feature selection across many applications, such as gene selection (Jain et al., 2000; Bergholz et al., 2010). For each feature f_j occurring in two different classes (Andrew, 2000), the mean μ_j+, μ_j- and standard the deviation δ_j+, δ_j- are calculated. The following equation illustrates how we can obtain the score of this feature $T(f_j)$ (Loo et al., 2005).

$$T(f_j) = \frac{\left|\mu_j^{+} - \mu_j^{-}\right|}{\sqrt{\frac{(\delta_j+)^2}{n+} + \frac{(\delta_j-)^2}{n-}}} \tag{2.16}$$

where $n+$ and $n-$ are the number of samples labelled as + and - respectively. Features with the highest scores are selected as the most discriminatory features.

2.4.1.6 Distance or Similarity/Dissimilarity Criteria for Feature Selection

Distance or similarity/dissimilarity criteria are based on calculating the similarity/dissimilarity between every two pairs of features using measures such as the Euclidean (for dissimilarity measure) and the cosine (for similarity measure) distances. These measures are encountered in many different fields such as biology, chemistry, and computer science (Ester et al., 1996; Liu & Motoda, 2007; Hasan et al., 2010). However, these measures are easily affected by noise or outlier data. The following are four criteria based on the idea of similarity and dissimilarity.

- Davies-Bouldin Index (DBI): The Davies-Bouldin index measures the similarity between two classes/clusters by calculating the within-class scatters and the inter-class distances (Gan et al., 2014).
- Fisher's linear discriminant: Fisher's criterion is applied for feature selection that projects high-dimensional data onto a line and then performs discrimination along this one-dimensional area. This criterion maximises the distance between the mean of the two different classes and minimises the variance within each class (Gu et al., 2012).
- Dunn's Index: Dunn's index is a measure introduced by Dunn (1974) for evaluating clustering algorithms (Bolshakova & Azuaje, 2003). The method aims to find a well-discriminated, small variance between members belonging to the same cluster.
- Cosine Similarity: Cosine similarity is a very common measure used for high-dimensional feature spaces. For instance, in information retrieval and text mining each term is hypothetically assigned a different dimension and a document is characterised by a vector (Bag-of-words) where the value of each dimension corresponds to the number of times that a term appears in the document (Cha, 2007; Hasan et al., 2010). Cosine similarity then gives a useful measure of how similar two documents are in terms of their subject matter (Singhal, 2001). Cosine similarity is adopted in chapter 4 for developing an improved PCA method for dimensionality reduction.

2.4.2 Wrapper Approach

The wrapper approach is a technique in widespread use for feature selection and evaluation, in which a classifier is defined beforehand and the classification accuracy is used as the evaluation criteria for the subset (of features) which is created. This approach can guarantee higher accuracy than the filter approach as it selects features by looking for features suitable to the classifier (Kohavi & Sommerfield, 1995; Kohavi & John, 1997; Das, 2001; Saeys et al., 2007; Ma et al., 2017). However, it needs significant processing time to choose such features, and the features selected for the subset may not be generalised for other classification methods.

The wrapper approach starts from a given subset G_0 which can be an empty set, a full subset, or any randomly selected subset. It then searches through the feature space using one of the search strategies suitable to this purpose. Subsequently, it evaluates each generated subset G_i by applying a learning model to the data labelled with G_i . If the performance of the learning model using G_i becomes better, G_i is considered to be the most recent best subset. For that reason, the wrapper approach then modifies G_i by adding or removing features to or from G_i (as dictated by the learning model) and the search iteration continues until a predefined stopping criterion is achieved (Kabir et al., 2010).

Basnet et al. (2012) proposed wrapper feature selection techniques to be used with various classification methods, such as naïve Bayes, logistic regression, and random forest. For testing, they used datasets of real phishing e-mails with 177 initial features. Khammassi & Krichen (2017) applied a wrapper approach based on a genetic

algorithm as a search strategy and logistic regression as a learning algorithm for network intrusion detection systems to select the best subset of features.

2.4.3 Hybrid Approach

A hybrid approach is a combination of both wrapper and filter methods. The filter approach tends to have a bias towards features with low dimensions. On the other hand, in very high-dimensional applications, the wrapper approach is impractical because of the high computational expense (Sebban & Nock, 2002; Fazil & Abulaish, 2018). Hybrid approach is promising to take the advantages of both filter and wrapper methods to reduce the redundant features efficiently without degrading the accuracy (Hu et al., 2015). However, the hybrid approach with single filter and single wrapper still has drawbacks, such as the selected features depend on the choice of a specific filter or wrapper. Thus, combining multiple filters and/or multiple wrappers is another way to identify potential and reliable features and also to improve the accuracy and robustness of the classification. Chapter 5 presents two hybrid approaches that combine wrapper and filter approaches (supervised and unsupervised) to gain high classification accuracy with low computational cost.

2.4.4 Embedded Approach

The embedded approach is one of the feature selection approaches that consider relations between one input feature and the output and searches locally for features that can achieve better local discrimination (Li et al., 2017). It employs independent measures to choose the best subsets then applies a learning algorithm to select the final

best subset among the candidate best subsets. It interacts with machine learning at a lower computational cost than the wrapper approach (Kumar & Minz, 2014).

2.4.5 Search Strategies

In feature selection a search algorithm is responsible for deriving potential feature subsets (Molina et al., 2002). Commonly used search strategies are summarised as follows.

1. **Exponential Search:** Exponential search is an optimised search process that guarantees the best solution (Larsson et al., 2017). However, optimal search doesn't need to visit all possible locations in the whole search space. Different heuristic functions can be applied to reduce the search space without tampering with the viability of the optimal search, such as branch and bound, and beam search (Doak, 1992; Molina et al., 2002).
2. **Sequential Search:** Sequential search is based on choosing only one candidate among all successors. This can be achieved in an iterative manner and in a finite number of steps. The results tend to be satisfactory (Aldehim, 2017) but do not always represent an optimal feature subset.
3. **Genetic Algorithms:** a genetic algorithm (GA) is a global heuristic search technique used to find exact or approximate solutions to optimisation problems. GA, as a search method for feature selection has been widely studied (Andrew, 2000; Hao et al., 2003; Oskoei & Hu, 2006; Nemati et al., 2009).
4. **Random Search:** Random search starts with a randomly selected subset. There are various ways to proceed from this to achieve an optimal subset, such as the Las Vegas algorithm (Kumar & Minz, 2014).

2.4.6 Other Methods for Dimensionality Reduction

High-dimensional features can be mapped into a lower-dimensional feature space. There are two kinds of dimensionality reduction methods: linear and non-linear.

2.4.6.1 Linear Methods

Principal component analysis (PCA), or Karhunen-Loeve expansion, is a very commonly used technique which transfers data linearly in such a way so as to highlight similarities and differences. It is used for feature extraction and dimensionality reduction (Cordero & Blain, 2007; Guyon et al., 2008; Liu et al., 2010; Aït-Sahalia & Xiu, 2017). The technique identifies the eigenvectors which match the largest eigenvalues of a covariance matrix. It does not depend on pre-labelled data for training purposes; thus, it is an unsupervised method. Gomez et al. (2012) employed PCA for e-mail classification based on the features of the text content. They proved by their experiments that PCA was able to produce good results and the performance of PCA was better than that of the popular VSM based techniques (Yu et al., 2009; Gbashi et al., 2014). However, PCA often lacks interpretability, which means that we cannot specify or understand in terms of features dealing with any particular subject because it combines such features together. In chapter 4, an improved PCA method is proposed and evaluated in comparison with the standard PCA.

2.4.6.2 Nonlinear Methods

Most of the non-linear methods used for dimensionality reduction are based on manifold learning theory. The main idea behind these methods is to build a weighted graph of the data points, which depends on neighbourhood relations. The data is

projected onto a lower dimension while keeping the relative relations among the graph nodes (Garrett et al., 2003).

There are three main types of non-linear methods for dimensionality reduction (Schölkopf et al., 1998; Saxena et al., 2004; Lee & Verleysen, 2007; Van et al., 2009).

1. Methods which attempt to preserve the local properties of the original data in the low-dimensional representation, for example, local linear embedding (LLE) and Laplacian Eigen-maps.
2. Methods which attempt to preserve global properties, for example, Isomap, Kernel PCA and MDS (Multidimensional Scaling).
3. Methods which perform a global alignment of a mixture of linear models, such as Manifold charting and LLC (Locally Linear Coordination).

2.5 Conventional Methods for Pattern Classification

Pattern classification is concerned with the method of making deductions from perceptual data, using a number of different tools from statistics, probability theory, computational geometry, signal processing and algorithm design (Raudys et al., 1991). The techniques which have emerged have been successfully applied to text categorisation (Lakshmi & Vijaya 2012), computer vision, and speech recognition (Bishop, 2006; Duda et al., 2012). For these purposes, a pattern is defined as a combination of features that is characteristic of an individual. In classification, a pattern is a pair of variables (x, y) (Juang & Katagiri, 1992) where x is a collection of samples or observations or features (feature vector) and y is the concept behind the samples (label). The quality of a feature vector is related to its ability to discriminate

between examples from different classes. Figure 2.5 a) demonstrates the distinction between good and poor features, and 2.5 b) illustrates feature properties (Duda et al., 2012). Examples from the same class should have similar feature values, while examples from different classes should have different feature values.

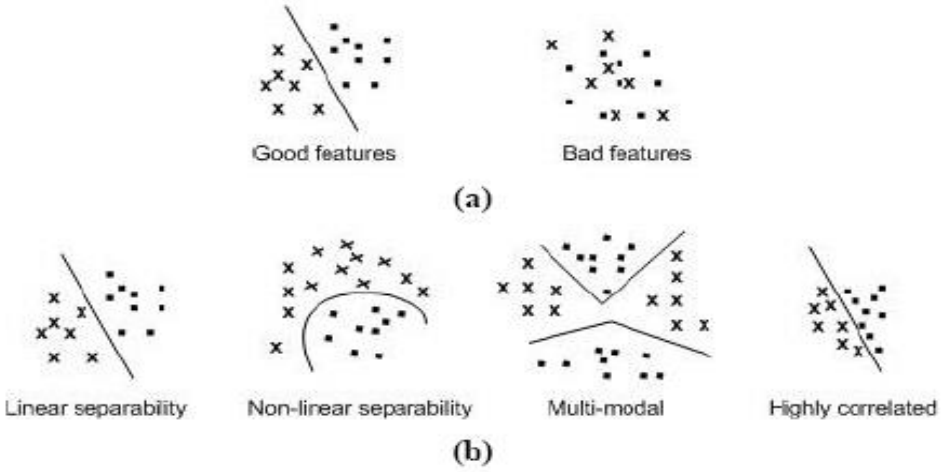

Figure 2.5: a) the distinction between good and poor features, and b) feature properties (Bishop, 2006).

2.5.1 Linear Discriminant Analysis (LDA)

Linear discriminative analysis (LDA) is a classification approach that projects high-dimensional data onto a one-dimensional line and performs discrimination across this one-dimensional space (Belhumeur et al., 1997). LDA is very commonly used for face detection, document classification and phishing detection (Altman, 1992; Peng et al. 2018).

Consider a two-class problem in which there are N_1 points from class C_1 and N_2 points from class C_2, the mean vectors of the two classes are given by:

$$m_1 = \frac{1}{N_1}\sum_{n\in C_1} X_n \quad , \qquad m_2 = \frac{1}{N_2}\sum_{n\in C_2} X_n \tag{2.17}$$

A method first put forward by Fisher was to maximise a function that provides a significant separation between the projected class means $(m_k = w^T m_k : k = 1,2)$ while also giving a small variance within each class. The within-class variance of the transformed data $(y = w^T x)$ from class C_k is therefore given by:

$$S_k^2 = \sum_{n\in C_K} \left(y_n - m_k\right)^2 \tag{2.18}$$

The Fisher criterion can be represented as the ratio of the between-class distance to the within-class variance:

$$J(w) = \frac{\left(\mu_2 - \mu_1\right)^2}{S_1^2 + S_2^2} \tag{2.19}$$

Equation (2.19) can be re-written as:

$$J(w) = \frac{w^T S_B w}{w^T S_w w} \tag{2.20}$$

where S_B is the between-class matrix and is given by:

$$S_B = (m_2 - m_1)(m_2 - m_1)^T \tag{2.21}$$

and S_w is the total within-class matrix

$$S_w = \sum_{n\in C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n\in C_2} (x_n - m_2)(x_n - m_2)^T \tag{2.22}$$

$J(w)$ is maximised when

$$\left(w^T S_B w\right) S_w w = \left(w^T S_w w\right) S_B w \tag{2.23}$$

If we multiply both sides of equation (2.24) by S_W^{-1} we obtain:

$$w = S_W^{-1}(m_2 - m_1) \tag{2.24}$$

The data points to be categorised are projected by *w*, and then a threshold that best separates the data is chosen from the analysis of the one-dimensional distribution.

2.5.2 The K-Nearest Neighbour (KNN) Method

K-Nearest Neighbour method is a non-parametric method used for classification which is based on computing a distance (for example, the Euclidean distance) between the new data item and every example in the sample dataset (Huh & Kim, 2012; Trstenjak et al., 2014; Lin et al., 2014). The class label of the new item will depend on the label shared by the majority of the K nearest samples found. If K=1, the decision concerning class will be made based on a single (nearest) neighbour. K is usually set to an odd number so that the result can never be a tie.

Huh and Kim (2012) used the KNN method for detecting domain name system (DNS) based phishing attacks and compared the performance of this method with the performance of the three other classifiers (Garera et al., 2007; Huh & Kim, 2012; Al-Janabi et al., 2017). KNN has been employed for image classification and it provides an impartial and consistent measurement (Awty-Carroll, 2018). However, when large sets of observations must be processed, this can affect its performance. Also, it can be

sensitive to irrelevant features. To solve this problem, feature selection should be applied in order to reduce the number of features.

2.6 Machine Learning Methods for Classification (Supervised Learning Methods)

Machine learning from data is a very challenging task. To adequately solve the various problems involved, we need to combine many different types of algorithms. This section will review a number of different approaches to learning from data, with a particular emphasis on addressing classification problems. Supervised learning is one of the most influential methods used in artificial intelligence across a number of disciplines: such as email classification, text categorisation, and document retrieval (Lewis et al., 2004; Rodrigues et al., 2017). Supervised learning can perform very well if a good feature representation is provided. The methodology is based on training using a set of labelled data representing a specific problem, for example, a set of emails labelled according to whether they are considered phishing attempts or not then evaluating the effectiveness of this training by providing unlabelled (as far as the software is concerned) data to the trained software (Peng et al., 2018). Supervised learning methods have been applied to natural language processing for topic classification and also for word embedding (Conneau et al., 2017).

The general task of a classification method is to assign a class label to an object/data point based on previously known observations. Consider *K* training data points generated according to an unknown probability distribution *P(x,y)*: $(x_1, y_1),.....,(x_K, y_K) \in R^N \times \{-1,+1\}$, where *N* is the dimension of the input data and

$\{-1,+1\}$ are the class labels (a two-class problem). The classification method can be described by the following function in general:

$$f : R^N \rightarrow \{-1,+1\} \tag{2.25}$$

A data point x will be assigned to class +1 when $f(x) > 0$ and to class -1 otherwise. The aim of the classifier is to partition feature space into class labelled decision regions by finding the optimal function $f(x)$, which can be learnt from training data. Borders between decision regions are called decision boundaries (Andrew, 2000), and the following expected error between the predicted and actual class labels is minimized:

$$R[f] = \int l(f(x), y) dP(x, y) \tag{2.26}$$

where l denotes an appropriately selected loss function. The most common loss function is the square error (Müller et al., 2008; Müller, 2015):

$$l(f(x), y) = ((sign(f(x)) - y)^2 \tag{2.27}$$

This lost function cannot be minimised directly as the underling distribution is unknown. Consequently, it is necessary to find an estimation of the optimal function, based on learning from training data. Let θ be the set of parameters for f. The goal then is to find the optimal θ that minimises the following loss function:

$$R_{emp}[f, \theta] = \frac{1}{K} \sum_{k=1}^{K} l(f(x_k, \theta), y_k) \tag{2.28}$$

However, the problem with this approach is possible overfitting (Bishop, 2006). Figure 2.6 provides example plots of training data sets of 10 points (blue circles); each plot

includes a sample of the input variable x along with the corresponding target variable t.

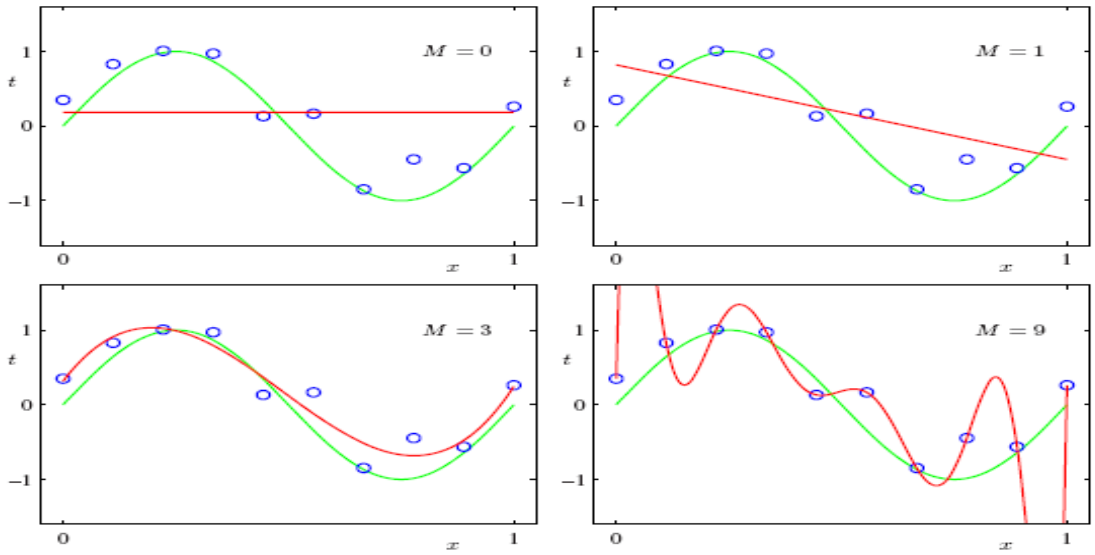

Figure 2.6: Overfitting problem associated with model complexity (Bishop, 2006).

The green curve describes the function $\sin(2\pi x)$ which is used to generate the training data and the red curves show the results of fitting polynomials of various orders to it. Four examples are shown of the results of fitting polynomials with different values of $M = (0, 1, 2, \text{and } 9)$ to the training dataset. The aim is to predict the value of t from the value of x, without prior knowledge of the green curve. The machine learning goal is to generalise the results it has achieved for new data points. Figure 2.6 shows that the higher-order model ($M = 9$) can fit exactly the training dataset, but it may not generalise well to new data points. The overfitting problem can be mitigated by restricting the complexity of the model learned by adding a regularising term to penalise higher model complexity. However, this can raise the dilemma of model selection (Bishop, 2006).

This study employs supervised learning for document classification. There are four classifiers which are used in this research: support vector machine (SVM) (Smola & Schölkopf, 2004; Bishop, 2006; Vapnik, 2013), multilayer perceptron (MLP) (Bishop, 2006), linear discriminant analysis (LDA) (Belhumeur et al.,1997; Altman, 1992), and K-Nearest Neighbour (KNN) method (Vapnik, 2013).

2.6.1 Support Vector Machines (SVMs)

Support Vector Machines represent a prevalent method used for classification, and SVMs can process a considerable number of features (Smola & Schölkopf, 2004; Vapnik, 2013; Abu-Nimeh et al., 2007; Almomani et al., 2012; Khonji et al., 2013). A SVM attempts to find the optimal hyper-plane that has maximum separation margins. A margin is defined to be the smallest distance between the decision boundary and any of the sample points (Smola & Schölkopf, 2004; Vapnik, 2013). Figure 2.7 illustrates the concept of SVM. The margin is defined as the orthogonal distance between the decision boundary and the closest of the data points. Optimising the margin leads to a specific choice of decision boundary. The points lying on the boundaries are called support vectors. The two-class classification problem in relation to the SVM method can be represented as follows (Bishop, 2006):

$$y(x) = w^T \phi(x) + b \tag{2.29}$$

where $\phi(x)$ is a fixed feature-space nonlinear transformation and b is a bias. This transformation allows a nonlinear classification problem to be solved in a higher dimensional feature space using simple linear methods (Almomani et al., 2012; Smola & Schölkopf, 2004).

SVM can be very effective for solving problems across a number of different disciplines. A linear SVM classifier was used for detecting and filtering phishing e-mails (Basnet et al., 2008; Basnet et al., 2012; Peng et al., 2018). However, in cases where the data are not linearly separable in the feature space, SVM will not be able to generalise very well (Bishop, 2008). A new, extended version of SVM has been introduced to overcome this problem incorporating what is known as the soft margin extension. This extension of SVM introduces a slack variable ξ_i that measures the degree of misclassification of data point x_i. A non-zero value for ξ_i allows x_i to not meet the margin requirement at a cost proportional to the value of ξ_i. The SVM method can be very time-consuming in terms of training the system using training data, and it is vulnerable to overfitting.

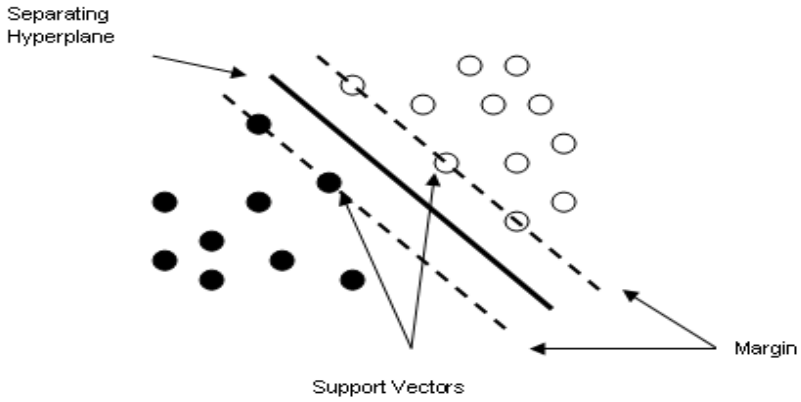

Figure 2.7: Support vector machines (Almomani et al., 2012)

2.6.2 Multilayer Perceptron (Feedforward Neural Networks)

A multilayer perceptron (a feed-forward neural network) is a supervised learning model using multiple hidden layers. It consists of a set of nodes or neurons that are organised into layers, usually the input layer, the output layer, and hidden layers in

between as shown in Figure 2.8, (Haykin, 2001; Gurney, 2014; DeSousa, 2016). Feed-forward networks are often trained using a back propagation learning scheme. Back propagation learning works by making modifications in weight values starting at the output layer and then moving backward through the hidden layers of the network (Gurney, 2014). The results from neural networks often lack interpretability since it is difficult for humans to interpret any symbolic meaning behind the learned weights (Abraham et al., 2018). The advantages of neural networks, however, include their high acceptance of noisy data, which often does not affect their ability to classify patterns on which they have not been trained. Mathematically, a node or neuron in the multilayer perceptron can be described as

$$O_j = f\left(net_j\right) \tag{2.30}$$

$$net_j = \sum_i W_{ji} O_i + \theta_j \tag{2.31}$$

where W_{ji} is the weight of the connection from the i^{th} node to the j^{th} node, 0_i is the output of the i^{th} node, θ_j is a bias with a similar function to a threshold, and the summation is over all the units feeding into node j. The activation function used is given by:

$$O_j = \frac{1}{1+e^{-net_j}} \tag{2.32}$$

The term back propagation refers to an iterative training process in which an output error E is defined by:

$$E = \sum_p E_p = \frac{1}{2}\sum_p \sum_j (t_{pj} - O_{pj})^2 \tag{2.33}$$

where the summation is performed over all the output nodes, and t_{pj} is the desired or target value of output O_j for a given input vector. The direction of steepest descent of error gradient in the parameter space is determined by the partial derivatives of E with respect to the weights and the bias in the network, i.e.,

$$\frac{\partial E}{\partial W_{ji}} = -\sum_p \delta_{pj} O_{pj} \tag{2.34}$$

$$\frac{\partial E}{\partial \theta_j} = -\frac{\partial E_p}{\partial net_{pj}} \tag{2.35}$$

It can be proved that,

$$\delta_{pj} = (t_{pj} - O_{pj})(1 - O_{pj})O_{pj} \tag{2.36}$$

The learning rule is given by

$$\Delta W_{ij}(t) = \eta \sum_p \delta_{pj} O_{pj} \tag{2.37}$$

$$\Delta \theta_j(t) = \eta \sum_p \delta_{pj} \tag{2.38}$$

where t denotes a given instant in time, δ_{pj} is the error, and η is the learning parameter.

Abu-Nimeh et al. (2007) compared six classifier techniques for phishing detection, including multilayer perceptron. Jameel & George (2013) applied a feed-forward neural network to some features which appear in the headers and HTML bodies of e-mails in order to classify these e-mails into phish and ham e-mails (Fette et al., 2007; Zhang & Yuan, 2012). Sachan et al. (2018) applied netural networks to text classification.

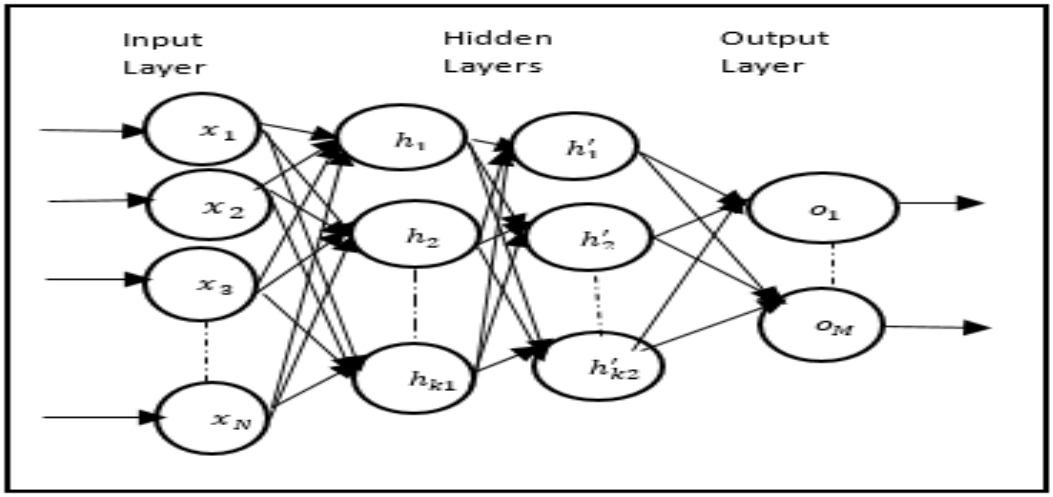

Figure 2.8: Multilayer perceptron.

2.7 Deep Neural Network Methods for Representation Learning

In recent years, big data analysis via deep learning has attracted much attention across a number of different areas of research such as computer vision (Bradski & Kaehler, 2008; Goodfellow et al., 2016; Girshick, 2017), speech recognition (Hinton et al., 2012; Zhang et al., 2017a), social media analysis (Siemens et al., 2011; Zhang et al., 2017b), fraud detection (Saxe et al., 2018), and medical informatics (Längkvist et al., 2014). One of the main advantages of deep learning, which is due to the use of deep neural network structures, is that it can learn feature representations without using a separate feature extraction process (Jiang et al., 2017), although feature extraction is a very significant activity within the processing required for pattern recognition (Touretzky et al., 1996). Using deep NNs, this is achieved by feature detector units arranged in layers. Lower layers detect simple features and feed them into higher layers, which in turn detect more sophisticated features (Bengio et al., 2013a).

LeCun et al. (1998) proposed a gradient-based learning method which could be applied to the detection and classification of handwritten digits in the MNIST (Modified National Institute of Standards and Technology) digit image database, a large database of handwritten digits that is commonly used for training image processing systems. Deep learning was first investigated by Hinton (2006) and then his methods were improved by Ranzato et al. (2007) and Lee et al. (2009). Much research has focused on the MNIST database in relation to implementing a new representation learning algorithm, which would challenge the supremacy of SVMs that achieved an error rate of only 1.4% (Lang et al., 1990; Touretzky et al., 1996; Bengio et al., 2007). Speech recognition and image processing were early applications of deep neural networks, especially with respect to using them for sound recognition by the application of shared weights in a temporal dimension (Bengio, 2009; Ngiam et al., 2010; Bengio et al., 2011; Dahl et al., 2012; Deng et al., 2013; Zhang et al., 2017b). In 2011, two transfer learning challenges were mooted, which were intended to encourage the exploitation of common features of different learning tasks in order to help distribute statistical and other knowledge more widely. The first challenge was presented at an ICML conference (Bengio, 2012; Mesnil et al., 2012) and required the use of unsupervised layers. The second challenge was proposed by Goodfellow et al. (2012) and involved the learning of hierarchical models. In 2012, a new version of the Microsoft audio video indexing service system was released, which was based on deep learning (Lee et al., 2009; Seide et al., 2011). Glorot et al. (2011) proposed a deep learning system for natural language processing, which learns to extract a meaningful representation of each customer review in an unsupervised fashion. This method

successfully performed domain adaptation on a large industrial-strength dataset of 22 domains.

Lai et al. (2015) applied recurrent convolutional neural networks to text classification and employed a max-pooling layer which automatically judged what words play key roles in the text. The results showed that their proposed model outperformed the state-of-the-art methods on four document-level datasets.

Natural language processing applications have been developed, which have employed representation learning and deep neural networks. Hinton (1986) proposed a distributed representation for symbolic data. Neural net language models were developed by Bengio et al. (2003), based on the context of statistical modelling. The SENNA (http://ml.nec-labs.com/senna) system was developed by Collobert et al. (2011), which implements representations across all the tasks involved with language modelling: parts-of-speech, recognition tags, chunking, embedded words, and semantic and syntactic labelling. Google's images search (Weston et al., 2010) learns word and image representations in combination. Google's system has been extended to multi-layer representations.

Many text classification systems have adopted deep neural networks (Young et al., 2017; Kowsari et al., 2017; Jiang et al., 2018). Deep learning has played a major role in phishing detection (Saxe et al., 2018; Nguyen et al., 2018; Le et al., 2018). Xu et al. (2018) employed deep neural networks for learning URL representations for malicious URL detection.

Feature learning usually requires unsupervised learning. Learning systems which can achieve this include the restricted Boltzmann machines (RBMs) (Salakhutdinov &

Hinton, 2009), sparse autoencoders (Lee, 2010, Abdulhussain & Gan, 2016), stacked autoencoders (SAE) (Zhou et al., 2015), de-noising autoencoders (Vincent et al., 2008; Vincent et al., 2010), and contractive autoencoders (Rifai et al., 2011).

Currently, the main focus of research on deep learning is that the compositions of nonlinearities, for example, those in deep feed-forward or recurrent networks, can be susceptible to initialisation problems. Other significant issues in deep learning include problems of overfitting and vanishing/exploding gradients emerging during error back-propagation due to the adoption of deep neural network structures such as deep multilayer perceptron (DMLP) (Geman et al., 1992; Glorot & Bengio, 2010).

Many techniques have been proposed to solve these problems which occur when training deep neural networks. A greedy layer-wise unsupervised pre-training approach was the first that mitigated the initialisation dilemma (Hinton et al., 2006). The central idea of this is to enable a neural network to learn a hierarchy of features one layer at a time by employing unsupervised feature learning for each new transformation at each layer so that all the transformations can then be collected together. The set of layers can be combined in order to initialise a supervised deep neural network (Hinton et al., 2006).

Bengio et al. (2007) proposed to train a deep neural network via a sequence targeted at an auxiliary objective and then the "fine-tuning" of the entire network using standard optimisation methods, such as stochastic gradient descent. Martens (2010) showed that the truncated Newton method has the ability to train deep neural networks from certain random initialisation states, without pre-training. However, these methods were found to be inadequate in relation to resolving the known training challenges. It is evident

that most deep learning models cannot be used with just random initialisation (Martens & Sutskever, 2012; Mohamed et al., 2012). Effective weight initialisation or pre-training has been widely explored for avoiding vanishing/exploding gradients (Aggarwal & Zhai, 2012; Basnet et al., 2012). Using a huge amount of training data can overcome overfitting, to some extent (Geman et al., 1992). However, in many applications, there is no large amount of training data available, or there is insufficient computer power available to handle huge amounts of training data. As a result, regularisation techniques, such as the sparse structure and dropout techniques, are extensively used for combatting overfitting (Aggarwal & Zhai, 2012; Basnet et al., 2012).

2.7.1 Sparse Autoencoders

An autoencoder is an unsupervised neural network trained by using back-propagation via a gradient descent algorithm, which learns a non-linear approximation of an identity function (Hinton & Salakhutdinov, 2006). The main idea of an autoencoder is to reconstruct its input through encoding and decoding so as to perform feature learning. Figure 2.9 illustrates a non-linear multilayer autoencoder network structure. There are many studies which have focused on learning feature representations that can reduce the dimensionality and improve the quality of text features (Saxe et al., 2018; Shen et al., 2017). By using sparsity penalty term in the learning objectives, sparse autoencoders can learn sparse feature representations.

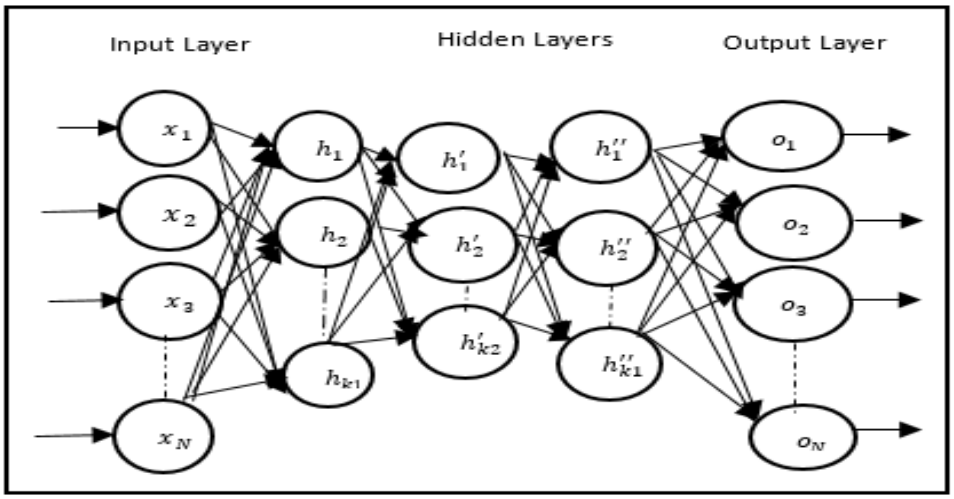

Figure 2.9: Typical autoencoder

2.7.2 The Restricted Boltzmann Machine (RBM)

A restricted Boltzmann machine has been applied to many machine learning domains, such as text, speech, motion data and images (Fatemi & Safayani, 2017). It was developed for the purpose of extracting highly discriminative low-dimensional features from high-dimensional raw data or complex data sets in an unsupervised manner by introducing hidden units (features) (Zhang et al., 2012; Cai et al., 2012). RBM consists of hidden and visible neurons (units), of which the visible units are used for input and the hidden units are used to model the data distribution (Bengio, 2009). It is restricted because the visible-visible and hidden-hidden connections are disallowed. Figure 2.10 illustrates a standard RBM with two layers: *v* denotes the visible units and *h* denotes the hidden units.

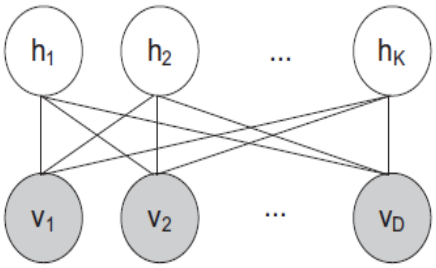

Figure 2.10: The restricted Boltzmann machine (Cai et al., 2012).

An RBM is able to model correlations of data by using fast learning algorithms such as Contrastive Divergence (CD) (Cai et al., 2012). The energy function $E(v,h|\Theta)$ of an RBM is defined as follows:

$$E(v,h|\Theta)=\frac{1}{2}(v^{T}W.h+b^{T}.v+c^{T}.h) \quad (2.39)$$

where v represents the visible units and h represents the hidden units, $\Theta\equiv(W,b,c)$, W represents the weights connecting the visible and the hidden units, and b,c are the biases of the visible and hidden layers respectively.

The partition function $Z(\Theta)$ is defined as follows:

$$Z(\Theta)=\sum_{x}\exp[-E(v,h|\Theta)] \quad (2.40)$$

The probability of the neurons' output is as follows:

$$P(v,h|\Theta)=\frac{1}{Z(\Theta)}\exp\{-E(v,h|\Theta)\} \quad (2.41)$$

The conditional probability of a visible unit is as follows:

$$p(v_i=1|h)=sigm(b_i+W_i.h) \quad (2.42)$$

The conditional probability of a hidden unit is as follows:

$$p(h_j = 1 \mid v) = sigm(c_j + W_j.v) \tag{2.43}$$

where *sigm(x)* is a sigmoid logistic function $(1 + exp(-x))^{-1}$. W_i is the i^{th} row vector of weight matrix *W*, and W_j is the j^{th} column vector of weight matrix W.

2.7.3 Deep Convolutional Neural Networks (DCNNs)

Deep convolutional neural networks have attracted much attention in many research areas such as computer vision, speech recognition, and natural language processing. DCNNs are designed to process data that come in the form of multiple layers (Zhang et al., 2017c). DCNNs are basically several layers of convolutions with non-linear activation functions (Sainath et al., 2013; Jalali et al., 2015; Zhou et al., 2015; Chen et al., 2018). There are three main types of layers as illustrated in Figure 2.11, which are used to build ConvNet architectures: the convolutional layer, the pooling layer, and the fully-connected layer.

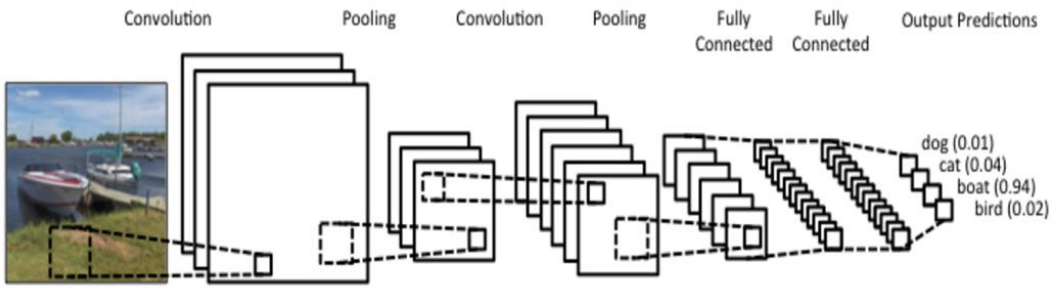

Figure 2.11: Typical structure of DCNNs for image classification (Chen et al., 2014b).

2.8 Performance Evaluation of Machine Learning Approaches

There are different types of criteria for performance evaluation, which are applied across different application areas such as information retrieval, document classification, and anti-phishing.

- Confusion Matrix

  A confusion matrix is often used to describe the performance of a classification model on a test dataset for which the true values are known. As an example, Table 2.1 shows the four classification possibilities of a phishing detection problem, which form a 2x2 confusion matrix (Khonji et al., 2013).

  Table 2.1: Confusion Matrix

		Predicted Class	
		Classified as C1 Phishing	Classified as C2 Non-phishing
Actual Class	C1 (yes)	TP	FP
	C2 (no)	FN	TN

- True Positive (TP) is the number of correctly predicted positive cases (e.g., phishing attacks), and true positive rate (TPR) is the ratio of TP to the number of real positive cases.
- False Positive (FP) is the number of incorrectly predicted positive cases, and false positive rate (FPR) is the ratio of FP to the number of real negative cases (e.g., benign web-sites/emails).
- True Negative (TN) is the number of correctly predicted negative cases, and true negative rate (TNR) is the ratio of TN to the number of real negative cases.

- False Negative (FN) is the number of incorrectly predicted negative cases, and false negative rate (FNR) is the ratio of FN to the number of real positive cases.
- Precision (P) is the ratio of the number of correctly predicted positive cases to the number of all the predicted positive cases, i.e.,

$$P = \frac{TP}{TP + FP} \tag{2.44}$$

- Recall (R) is defined as

$$R = \frac{TP}{TP + FN} \tag{2.45}$$

- F1 score is a measure based on both P and R, which is defined as

$$F1 = \frac{2PR}{P + R} \tag{2.46}$$

- Accuracy (ACC) is the ratio of the number of correctly predicted cases to the number of total cases, i.e.,

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.47}$$

2.9 Detection of Phishing Emails and Websites (Application of Document Classification)

2.9.1 Introduction

Phishing is a kind of attack on users carried out entirely via electronic communication channels, such as e-mails or fake websites, through which the attackers attempt to

defraud users into divulging sensitive information, such as usernames and passwords associated with their bank accounts (Jain & Gupta, 2018; Abu-Nimeh et al., 2007). The term "phishing" was coined in 1996 when scammers attacked the accounts of America Online (AOL). Phishing detection has become a crucial issue since then (Jakobsson & Myers, 2006; Ma et al., 2009).

The Anti-Phishing Working Group (APWG) (2017) tracks the number of unique phishing websites, the number of unique phishing e-mail reports, the number of brands targeted by phishing campaigns and the number of domain names used in attacks. Figure 2.12 shows these statistics for the first half of 2017.

	January	February	March	April	May	June
Number of unique phishing websites detected	42,889	50,567	51,265	50,328	45,327	50,720
Number of unique phishing e-mail reports (campaigns)	96,148	100,932	121,860	87,453	93,285	92,657
Number of brands targeted by phishing campaigns	424	423	444	460	457	452
Number of domain names used in attacks	13,977	15,877	17,397	21,652	21,373	18,404

Figure 2.12 : Statistical highlights for the first half of 2017 (APWG, 2017).

2.9.2 Types of Phishing Attacks

Several different types of phishing attacks have been identified. However, all of them aim at defrauding social media users in order to steal their personal information (such as usernames and passwords of bank accounts) and, of course, their money (Peng et al., 2018). Almomani et al. (2013) categorised phishing attacks into two categories: deceptive phishing and malware-based phishing. In this section, we categorise

phishing attacks into three types: phishing e-mails, fake websites, and malicious websites.

Phishing e-mails generally contain warning messages with links to spoofed websites (Conti et al., 2018), which have been set up for the purpose of procuring the personal information of the user. Based on the receivers, phishing e-mails are divided into two kinds: clone phishing and spear phishing. In the first type, attackers procure information about their victims from legitimate e-mails, such as content and recipient addresses, and then they resend a deceptive e-mail with a spoof link that appears original (Gupta et al., 2017; McCrohan & Harvey, 2011). In the second type, the attackers select a specific group of users, for example, people from the same organisation or university, instead of randomly disseminating thousands of e-mails. The procedure for sending phishing e-mails has three components: Message Transfer Agents (MTA), Message Delivery Agents (MDA), and Mail User Agents (MUA). Figure 2.13 shows the procedure for sending phishing e-mails and how such emails are transferred across a computer network.

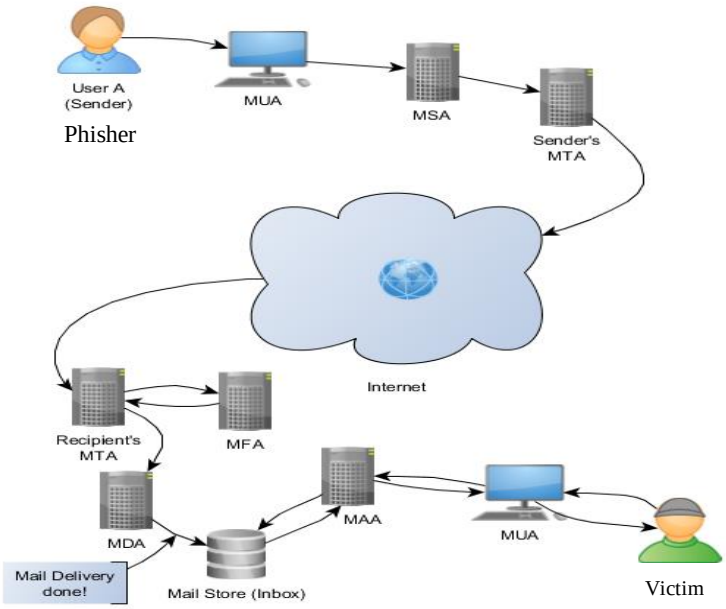

Figure 2.13: Procedure for sending phishing via e-mails (Almomani et al., 2013).

In 2011, the clients of two well-known security companies, RSA and HB Gary, received spear phishing e-mails (Basnet et al. 2008; Zhang et al., 2007). This shows that phishing attacks threaten not just unsuspecting (naïve) users, but that even companies with technical expertise can fall prey to the danger of phishing.

Fake websites are imitative e-commerce websites that resemble the websites of legitimate businesses, such as banks or other businesses. Such fake websites are designed by cybercriminals to misdirect the user to illegitimate websites or to a legitimate one, such as eBay, PayPal, or various banking services, which they can monitor by proxies in order to steal the customer's money (Cordero & Blain, 2007). Figure 2.14 shows a sample of a fake PayPal website.

Figure 2.14 : Fake PayPal website (Aggarwal et al., 2013).

A malicious or malware website is a website which offers a downloadable file, intended to be attractive to the user, which contains malicious software (e.g., spyware or Trojans) (Antonakakiset al, 2018). Upon installation of the downloaded file, it can gather information about the person or organisation without their knowledge (Aggarwal et al., 2013).

It is plausible to argue that there is an overlap between the above three phishing types. The phishing e-mails may contain a spoof link to fake or malicious websites.

2.9.3 Types of Features for Phishing Attacks Detection

This thesis focuses on categorising the features of e-mails and websites, which are used to manipulate the victims, into two kinds: URL-based and content-based features. The former can be further divided into two sub-types: technical features related to the operation of search engines and URL structures which identify the URL, such as the domain name. Content-based features can also be divided into two sub-types: user-

visible features (external), such as text features and the frequency of words, and user-invisible features (internal), such as HTML and JavaScript code (Badadhe et al., 2014). Figure 2.15 shows the categorisation of the features found in phishing e-mails and websites. This research has looked in particular at the external features: both content-based features and URL-based features.

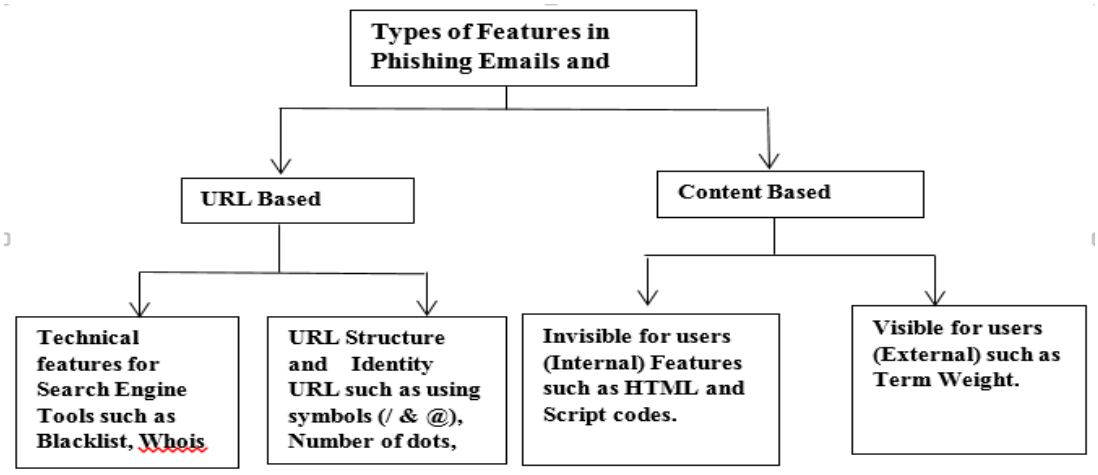

Figure 2.15: Types of features in phishing e-mails and websites.

Machine learning methods have played a significant role in phishing detection and classification (Islam & Abawajy, 2013). Detecting phishing e-mails and websites has been formulated as a statistical pattern recognition task and machine learning is at the core of building phishing detection systems due to its efficiency and effectiveness for relatively simple systems.

2.10 Summary

This chapter has reviewed the state-of-the-art document representation and feature extraction methods, feature selection and dimensionality reduction methods, machine learning and deep learning approaches for document classification, and example

applications and assessment of document classification. Although the different techniques investigated do achieve high classification accuracy, they still suffer from some problems: 1) Lack the ability to generate effective features and in particular lack efficient feature selection and dimensionality reduction methods; 2) Lack interpretability. The weakness of current learning algorithms is that they are not good at extracting and organising discriminative information from data in an automatic manner. There are different types of metrics for document classification methods. The features used by most existing document classification methods are insufficient. We need to focus, in particular, on extracting informative features for document classification. It is very difficult to interpret the effectiveness of feature selection methods because in many applications datasets often have a large number of irrelevant and redundant features in high-dimensional space and not much knowledge about the large amount of features. In addition, the overfitting problem is one of the leading issues in machine learning and deep learning in particular. However, the available solutions to the above problems do not fully answer the main challenges related to the important aspects of big data analysis and classification. This thesis focuses on proposing and investigating effective methods for feature selection and document classification. The remaining chapters will attempt to provide answers to some of the main issues relating to feature extract and feature selection for document classification.

In the next chapter, we will present a general explanation of the proposed methods for feature extraction and feature selection for document classification and will describe in detail the evaluation methods we used in this thesis to assess the performance of the proposed methods by measuring the classification performance.

Chapter 3

Methodology

3.1 Introduction

A review of the main topics was presented in the previous chapter, including the methods for document representation, feature extraction, and feature selection and dimensionality reduction, conventional and machine learning methods for classification, and deep neural networks for representation learning.

The research methodology adopted in this thesis consists of three steps:

- Problem analysis and investigation of potential methods: Based on the outcomes of problem analysis, theoretical approaches are investigated first, and then applicable algorithms and methods are developed and subsequently tested on multiple datasets.
- Experimental testing: Improved or newly proposed approaches are tested on various datasets collected from various sources: such as emails datasets, technical websites features for fraud detection, and others. On each dataset experiments are repeated with different data splits for training, validation and testing respectively.
- Results analysis: The experimental results are analysed using various statistical tests to assess whether the performance differences among the methods used are statistically significant. It is also exploited whether the results obtained are interpretable.

This chapter presents general steps of the research methodology for developing new methods for feature extraction, feature selection and dimensionality reduction. Work packages are presented in section 3.2 and the general experiment procedure adopted is described in section 3.3.

3.2 Work Packages

This Ph.D. thesis aims to develop new methods for extracting effective features and relevant learning representations by employing machine learning methods including deep learning for document classification, with a focus on text and document analytics.

The first work package is to propose an improved PCA method based on the singular value decomposition of a matrix of cosine similarity or correlation between pairs of feature vectors. For initial feature extraction, four term weighting schemes are applied for document representation: term frequency (TF), term presence (TP), term frequency and inverse document frequency (TF-IDF), and term presence and class-specific document frequency (TP-CSDF). The proposed approach is evaluated by employing three well-known classifiers SVM, KNN and LDA to classify emails/documents/news-items using the best number of principal components or features to achieve the highest performance in terms of classification accuracy and the minimum number of selected features. Experiments are conducted with performance evaluated by comparison of the proposed method with the standard PCA based on the singular value decomposition of covariance matrix.

The second work package is to propose hybrid methods for feature subset selection, aiming to further reduce feature dimensionality and thus improve classification

accuracy and interpretability as well. The proposed methods use a two-stage process for selecting a subset of relevant features. The first stage selects feature subsets based on either the union or intersection of features selected according to distance or similarity measures (unsupervised approach) and mutual information measures (supervised approach). The second stage employs a wrapper approach on the features selected at the first stage. The proposed methods are evaluated by employing three well-known classifiers (SVM, KNN, and LDA) to classify emails/documents/news-items. Performance comparison with individual filter approaches and full wrapper approach is carried out in terms of classification accuracy, the processing time consumed, and the number of selected features.

The third work package is to develop a new scheme for employing sparse autoencoder to extract features, which uses a class-specific (supervised) pre-trained approach to learn extracting features for each class separately, which is evaluated by employing two well-known classifiers (SVM and LDA) in comparison with feature extraction based on unsupervised trained sparse autoencoder.

The final work package is to explore deep classifier structures using SAE and MLP for higher-level feature extraction, leading to a three-stage learning algorithm that can overcome the difficulties encountered when training deep neural networks with limited training data in high-dimensional feature space. The performance of the proposed three-stage learning algorithm for DMLP is evaluated via comparisons with the performance of support vector machines combined with SAE and DMLP trained with random weight initialisation.

3.3 Experiment Procedure

3.3.1 Datasets

Eight datasets are used in the experiments in this thesis. First, an emails_v1 dataset (http://snap.stanford.edu/data/) has 6,000 samples from two classes (3,000 ham/non-phishing and 3,000 phishing) from different resources such as Cornel University and Enron Company. Second, an email_v2 dataset (https://www.kaggle.com/wcukierski/enron-email-datase) has 1,000 samples from two classes (500 ham/non-phishing and 500 phishing) from Cornel University and Kaggle competition website. Third, a document dataset consisting of 10 categories of the Reuters-21578 dataset (http://www.daviddlewis.com/resources/testcollection/reuters21578) has 1885 samples from 10 classes. Fourth, the 20 Newsgroup corpus dataset is a collection of approximately 20,000 newsgroup documents divided into 20 discussion groups (https://archive.ics.uci.edu/ ml/datasets/Twenty+Newsgroups). Because some news groups are very closely related to each other, seven relatively distinguishable categories were used in the experiments, producing a dataset named 20news_v1. Fifth, a document dataset consists of four relatively distinguishable categories from the 20 Newsgroup corpus dataset, which is named 20news_v2. Sixth, a technical website features dataset (http://khonji.org/phishing_studies) has 4,230 samples from two classes (2,115 phishing and 2,115 non-phishing). Seventh, the Musk dataset (http://archive.ics.uci.edu/ml/datasets.html) has 6,598 samples from two classes (musk and non-musk). Eighth, a self-drive intrusion detection dataset has 10,000 samples

from two classes (5,000 malicious and 5,000 non-malicious). Table 3.1 summaries the general information about these datasets.

Five of these datasets are text, which were pre-processed by tokenization, removing stop words, such as ‘the’, ‘for’, numbers and symbols, and stemming words such as ‘attachment’ and ‘attached’, which help to produce a bag of words (BOW) as original features. After that, four term weighting schemes were applied to weight the words in the BOWs of the document datasets: term frequency (TF), term presence (TP), term frequency and inverse document frequency (TF-IDF), and term presence and class-specific document frequency (TP-CSDF) (Plansangket & Gan, 2015a). The other three datasets (technical website features, Musk, and self-drive intrusion detection) are numerical, with feature values normalised in the experiments.

Table 3.1: Description of the Datasets.

No.	Dataset	No. of Samples	No. of Classes	No. of Features
1	Emails_v1	6,000	2	1,014
2	Emails_v2	1,000	2	465
3	Reuters-21578	1,885	10	421
4	20news_v1	20,000	7	100
5	20news_v2	20,000	4	2,591
6	Technical Website Features	4,230	2	47
7	Musk	6,598	2	166
8	Self-drive Intrusion Detection (malicious dataset)	10,000	2	80

3.3.2 Splitting Data for Reliable Validation and Testing

For each dataset, the experiment was repeated five times with different data partition obtained by shuffling with different random seeds for each run. The mean and standard deviation (Std.) of the experimental results were calculated to assess the consistency of the results. In each run, a dataset was partitioned into a training set and a testing set. Part of the training set was used as validation data for choosing the best parameter values for the various methods for comparison. The proposed approaches were applied to select the best number of discriminate features in the sense that the highest cross-validation (5-folds) performance was achieved.

3.3.3 Algorithms for Classification

There are many classifiers of different natures which can be used for supervised classification. Generally, the classification performance may be dependent on the types of classifiers used, under exactly the same conditions, subset of features, number of samples, and training procedure. To verify the consistency of the proposed feature selection methods, we have used four types of classifiers in our experiments: linear discriminant analysis (LDA) (Belhumeur et al.,1997; Altman, 1992), K-Nearest Neighbour (KNN) method (Vapnik, 2013), support vector machine (SVM) (Smola & Schölkopf, 2004; Bishop, 2006; Vapnik, 2013), and multilayer perceptron (MLP) (Bishop, 2006), as described in section 2.5.1, 2.5.2, 2.6.1, and 2.6.2. These four classifiers have been chosen because they represent three quite different approaches in machine learning and are commonly used in data mining practice.

3.3.4 Criteria for Performance Evaluation

There are different types of criteria for performance evaluation, as described in section 2.8, which are applied across different application areas such as information retrieval, document classification, and anti-phishing. In this work, we used the classification accuracy, which is defined as the ratio of the number of correctly predicted cases to the number of total cases, i.e.,

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{3.1}$$

3.3.5 Statistical Significance Test

Many studies adopt various statistical methods to assess whether the performance differences among the methods are statistically significant. The selection of the test should be based on statistical suitability and what we intend to assess. The statistical tests adopted in this research are as follows:

T-test, which is parametric method to determine whether two sets of performance data are significantly different from each other. We used the T-test in chapters 4, 5, 6, and 7 to compare the performances in terms of classification accuracy and number of selected features.

Wilcoxon's rank sum test, which is a non-parametric method to determine if two sets of data are significantly different from each other. We used the Wilcoxon's rank sum

test in chapters 5, 6, and 7 to compare the classification performances in terms of classification accuracy and number of selected features.

Chapter 4

An Improved PCA Approach Based on Cosine Similarity and Correlation for Text Feature Dimensionality Reduction

4.1 Introduction

Text analysis refers to techniques that extract information from textual data such as that provided by social networks, emails, documents, and news media. These techniques include statistical analysis, computational linguistics, and machine learning. Social media has been a very recent target of text analysis of both structured and unstructured data such as those yielded by Facebook, Blogger, and Twitter (Gandomi & Haider, 2015). The number of samples or observations and the number of features available for data mining have increased significantly in different applications such as fraud detection and document retrieval (Almusallam, 2017). This could be problematic and may cause the computational complexity of machine learning approaches employed for text and document classification. Due to the possible existence of a large number of irrelevant and redundant terms in high-dimensional feature space, feature dimensionality reduction is a very important pre-processing step in text categorization and pattern recognition in high-dimensional feature space (Yu & Liu, 2003). In general, selecting an appropriate smaller subset of features can make the training of a classifier more robust (Gan et al., 2014).

There are many different criteria applied to filter-based feature selection and dimensionality reduction, such as distance or similarity/dissimilarity criteria (Cha, 2007; Saeys et al., 2007). Principal component analysis (PCA) is a commonly used method for feature extraction and dimensionality reduction, which combines features by using orthogonal transformation to convert a set of samples of possibly correlated (similar) variables into a set of data of linearly uncorrelated variables (Gomez et al., 2012; Tian et al., 2010; Kim et al., 2002). It has been used in almost all scientific disciplines, for instance, Liu et al. (2010) employed PCA for content-based email classification. Kumar and Ravi (2017) adopted PCA for text classification.

This thesis focuses on text and document analytics. This chapter proposes a PCA method based on the singular value decomposition of a matrix of cosine similarity or correlation between pairs of feature vectors. Experiments were conducted with performance evaluated by comparison of the proposed method with the standard PCA based on the singular value decomposition of covariance matrix. The remainder of the chapter is structured as follows: sections 4.2-4.5 describes the basic principles of PCA, correlation, cosine similarity, and the proposed approach. Section 4.6 presents experimental results and discussion. A summary of the work described in this chapter is drawn in Section 4.7.

4.2 Principal Component Analysis

The steps of the standard PCA algorithm based on the singular value decomposition of the covariance matrix are as follows (Yu et al., 2009; Gbashi et al., 2014).

• Organise training data in an $m \times n$ matrix X, where m is the number of samples and n is the number of variables or features in each sample.

• Set the mean of each variable to zero, transferring matrix X to a mean removed matrix D

• Compute an $n \times n$ matrix C, the covariance of matrix D

• Compute eigenvalues and eigenvectors of C

• Sort eigenvectors (i.e., principal components) in the descending order of eigenvalues.

• Decide the number of top principal components, k, to keep using some criteria, forming an $n \times k$ projection matrix P, with each column vector being a kept principal component.

• Compute the projected data by D×P.

4.3 Cosine Similarity

As discussed in section 2.4.1.6, cosine similarity is a measure of similarity between two non-zero vectors of an inner product space that measures the cosine of the angle between them (Singhal, 2001). The cosine of 0 is 1, and it is less than 1 for any other angle in the interval (0, 2π). Given two vectors of features, $x=\{x_1, x_2, \ldots, x_n\}$ and $y=\{y_1, y_2, \ldots, y_n\}$, the cosine similarity is defined as follows (Cha, 2007; Hasan et al., 2010):

$$Sim(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^{n} x_i \cdot y_i}{\sqrt{\sum_{i=1}^{n} x_i^2 \sum_{i=1}^{n} y_i^2}} \tag{4.1}$$

4.4 Pearson's Correlation

A correlation coefficient is a number between -1 and 1 that refers to the strength of correlation between features (Guyon & Elisseeff, 2003). The Pearson's correlation for measuring the strength of a linear association between two variables is defined as follows:

$$R(\mathbf{x},\mathbf{y}) = \frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}} \tag{4.2}$$

where $\bar{x}$ is the mean of x and $\bar{y}$ is the mean of y.

4.5 Proposed Approach

The proposed approach in this chapter employs the cosine similarity or correlation defined in (4.1) and (4.2) to compute the similarity or correlation between each pair of feature vectors in the training dataset, generating a correlation matrix or similarity matrix of the training data, which are used to replace the covariance matrix in the standard PCA algorithm. In the standard PCA, the *k* principal components corresponding to the *k* largest eigenvalues are selected to form the projection matrix. In our approach, the selected *k* principal components correspond to the *k* smallest eigenvalues. This can be explained by the fact that the smallest similarities or correlations between features mean the largest divergence between them.

The number of principal components to keep or the number of features to select, i.e., the value of *k*, can be determined by the classification accuracy on a training dataset with *k* selected features and then validated with a test dataset. We obtain performance

curves with accuracy against the number of kept principal components, i.e., the number of selected features. The value of *k* is chosen to correspond to the highest accuracy. Three well-known classifiers were used in this study. They are support vector machine (SVM) (Smola & Schölkopf, 2004; Vapnik, 2013), K-nearest neighbour (KNN) (the best value of *K* is chosen by cross-validation) (Altman, 1992), and linear discriminant analysis (LDA) (Belhumeur et al., 1997).

4.6 Experimental Results

4.6.1 Datasets

Eight datasets as described in Chapter 3 were used in the experiments (email_v1, email_v2, Reuters-21578, 20news_v1, 20news_v2, Musk, technical website features, and self-drive intrusion detection).

4.6.2 Experiment Procedure

For each dataset, the experiment was repeated five times with different data partition obtained by shuffling with different random seeds for each run to assess the consistency of the results. The full description of the procedure is in section 3.3.

4.6.3 Results

Classification accuracy, standard deviation and the corresponding number of required features: Tables 4.1-4.24 show the classification accuracy, standard deviation (Std.) and the corresponding number of required features by three PCA methods (PCA-Cov: covariance matrix based, PCA-Corr: correlation matrix based, and PCA-

cos: cosine similarity matrix based). It is clear that the proposed PCA using the cosine similarity criterion achieved competitive accuracy with a smaller number of required features compared to the PCA based on covariance and correlation, across different datasets and different classifiers.

Table 4.1: Performance of SVM on email-v1 dataset.

Methods	**TF**			**TP**			**TF-IDF**			**TP-CSDF**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	88.6%	3.8	1014	90.6%	3.01	1014	90.6%	3.2	1014	83.9%	3.1	1014
PCA-Cov	87.5%	2.3	30	90.7%	0.7	10	88.4%%	1.1	29	86.3%	2.3	36
PCA-Corr	88.7%	1.1	32	91.4%	0.2	9	91.6%	0.6	18	84.4%	0.6	20
PCA-Cos	90.5%	0.9	19	92.5%	0.1	7	91.4%	0.5	24	85.9%	0.7	18

Table 4.2: Performance of SVM on reuters-21578 dataset.

Methods	**TF**			**TP**			**TF-IDF**			**TP-CSDF**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	81.5%	2.2	421	80.5%	0.013	421	80.5%	0.121	421	81.7%	3.47	421
PCA-Cov	80.3%	1.03	47	79.6%	0.009	46	80.2%%	0.021	39	80.7%	2.25	38
PCA-Corr	80.2%	1.01	37	79.7%	0.012	43	80.7%	0.009	35	80.8%	1.07	20
PCA-Cos	81.5%	0.93	18	81.5%	0.007	35	81.8%	0.008	30	81.6%	0.24	19

Table 4.3: Performance of SVM on 20news_v1 dataset.

Methods	**TF**			**TP**			**TF-IDF**			**TP-CSDF**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	60.6%	3.05	100	72.4%	2.57	100	72.3%	0.145	100	73.5%	4.46	100
PCA-Cov	61.8%	1.17	50	73.3%	2.19	29	73.3%	0.035	44	73.7%	2.48	32
PCA-Corr	65.3%	0.87	26	73.8%	1.64	22	73.5%	0.024	40	74.1%	1.28	28
PCA-Cos	67.9%	0.69	17	74.9%	1.35	21	73.9%	0.013	35	74.8%	1.17	24

Table 4.4: Performance of SVM on email_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	92.7%	2.06	465	83.4%	2.06	465	90.3%	2.59	465	90.5%	3.46	465
PCA-Cov	95.8%	1.37	57	88.3%	1.95	59	91.3%	2.13	42	92.7%	1.18	40
PCA-Corr	96.3%	0.92	53	89.5%	1.36	32	92.5%	1.46	23	92.1%	0.86	33
PCA-Cos	97.8%	0.73	31	90.6%	1.04	30	93.9%	1.29	17	94.6%	0.79	28

Table 4.5: Performance of SVM on 20news_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full	74.7%	2.52	2592	71.9%	3.13	2592	74.7%	0.422	2592	72.9%	2.97	2591
PCA-Cov	79.3%	1.08	78	75.6%	0.94	65	82.6%	0.020	83	76.7%	1.48	54
PCA-Corr	80.4%	1.83	67	77.3%	1.06	58	84.7%	0.013	32	78.6%	1.16	28
PCA-Cos	82.4%	1.06	52	79.4%	0.35	45	86.3%	0.012	21	79.4%	0.92	17

Table 4.6: Performance of SVM on technical website features dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	82.1%	0.331	47	84.9%	0.230	47
PCA-Cov	84.1%	0.009	35	83.6%	0.007	29
PCA-Corr	86.4%	0.007	23	86.8%	0.006	26
PCA-Cos	88.1%	0.006	21	89.7%	0.005	19

Table 4.7: Performance of SVM on musk dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	84.4%	0.132	166	85.7%	0.232	166
PCA-Cov	88.2%	0.021	56	87.6%	0.009	48
PCA-Corr	89.4%	0.012	35	88.6%	0.008	37
PCA-Cos	93.7%	0.021	29	93.9%	0.001	24

Table 4.8: Performance of SVM on the malicious dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	91.3%	2.54	80	92.8%	2.16	80
PCA-Cov	92.8%	1.76	42	93.7%	1.19	43
PCA-Corr	94.3%	1.18	35	94.2%	0.39	38
PCA-Cos	97.6%	0.76	27	97.8%	0.2	25

Table 4.9: Performance of LDA on email_v1 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	82.4%	0.138	1014	85.7%	0.236	1014	85.6%	0.137	1014	85.6%	0.235	1014
PCA-Cov	87.1%	0.023	12	90.7%	0.005	10	88.3%%	0.013	29	86.3%	0.037	42
PCA-Corr	90.4%	0.009	10	91.3%	0.0035	9	90.3%	0.004	12	87.5%	0.005	27
PCA-Cos	91.5%	0.007	9	92.5%	0.0028	7	90.4%	0.003	10	88.1%	0.001	20

Table 4.10: Performance of LDA on reuters-21578 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	71.4%	0.126	421	74.7%	0.211	421	78.5%	0.113	421	80.7%	0.234	421
PCA-Cov	70.3%	0.011	42	71.6%	0.010	48	72.2%%	0.025	42	74.7%	0.026	40
PCA-Corr	71.2%	0.015	34	75.7%	0.026	43	73.7%	0.010	38	75.5%	0.021	24
PCA-Cos	72.5%	0.012	29	76.5%	0.009	35	75.4%	0.006	36	77.1%	0.003	22

Table 4.11: Performance of LDA on 20news_v1 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	61.6%	3.54	100	71.4%	3.53	100	72.3%	4.50	100	73.5%	4.48	100
PCA-Cov	62.8%	1.23	53	72.3%	2.12	39	73.3%	3.58	44	73.7%	2.43	38
PCA-Corr	66.3%	1.07	23	73.5%	1.65	46	73.5%	2.45	40	76.1%	1.23	18
PCA-Cos	66.9%	0.92	21	73.6%	1.32	29	73.9%	1.30	35	76.8%	1.19	16

Table 4.12: Performance of LDA on email_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	85.6%	2.01	465	83.4%	2.51	465	79.3%	2.41	465	80.5%	3.40	465
PCA-Cov	92.8%	1.32	57	90.3%	2.04	59	83.3%	2.26	49	81.7%	1.32	68
PCA-Corr	93.3%	0.93	53	91.5%	1.34	42	84.5%	1.63	43	82.1%	0.91	38
PCA-Cos	95.8%	0.72	31	92.6%	1.19	36	86.9%	1.47	32	84.8%	0.73	31

Table 4.13: Performance of LDA on 20news_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	64.7%	3.08	2591	68.9%	3.80	2591	64.7%	3.26	2591	70.9%	2.91	2591
PCA-Cov	80.3%	2.21	58	87.6%	1.16	45	82.7%	2.16	83	83.7%	1.72	25
PCA-Corr	81.6%	1.93	27	87.3%	1.63	38	84.6%	1.62	32	84.6%	1.23	12
PCA-Cos	84.5%	1.19	22	89.4%	0.71	35	87.3%	1.51	21	86.4%	1.08	11

Table 4.14: Performance of LDA on technical website features dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	82.1%	3.81	47	87.7%	3.42	47
PCA-Cov	91.1%	0.23	20	92.6%	0.61	16
PCA-Corr	94.4%	0.65	11	95.6%	0.45	10
PCA-Cos	95.1%	2.01	9	96.4%	0.18	8

Table 4.15: Performance of LDA on musk dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	82.4%	3.81	166	83.7%	3.41	166
PCA-Cov	89.2%	2.23	41	90.6%	0.61	46
PCA-Corr	93.4%	1.67	32	92.6%	0.42	39
PCA-Cos	94.7%	2.60	19	93.9%	0.21	28

Table 4.16: Performance of LDA on the malicious dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	92.4%	3.91	80	93.6%	3.51	80
PCA-Cov	93.4%	2.12	43	95.2%	1.60	43
PCA-Corr	95.3%	1.62	34	96.1%	0.42	33
PCA-Cos	96.9%	2.61	20	96.8%	0.13	22

Table 4.17: Performance of KNN on email-v1 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	80.4%	3.61	1014	86.7%	2.31	1014	83.6%	3.71	1014	84.8%	3.51	1014
PCA-Cov	87.1%	1.21	58	90.3%	0.72	33	86.3%%	1.13	29	82.8%	1.34	44
PCA-Corr	90.4%	0.80	29	91.7%	0.32	26	88.3%	0.63	12	82.6%	0.41	27
PCA-Cos	91.5%	0.60	16	92.6%	0.24	6	89.9%	0.21	10	83.5%	0.23	24

Table 4.18: Performance of KNN on reuters-21578 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	70.6%	3.21	421	70.6%	3.12	421	78.5%	3.33	421	73.6%	3.71	421
PCA-Cov	71.6%	2.24	48	71.6%	2.04	48	76.2%%	2.54	42	74.8%	1.62	35
PCA-Corr	71.4%	1.32	38	72.7%	1.61	43	78.7%	1.09	38	75.5%	0.73	21
PCA-Cos	71.8%	1.25	34	74.5%	0.92	32	79.4%	0.61	36	79.1%	0.37	26

Table 4.19: Performance of KNN on 20news_v1 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	65.6%	2.81	100	70.4%	2.51	100	70.3%	3.51	100	72.5%	4.00	100
PCA-Cov	66.8%	1.14	38	70.3%	2.03	30	73.3%	2.23	34	72.7%	2.19	37
PCA-Corr	67.3%	0.92	24	71.5%	1.63	26	74.5%	2.03	30	75.1%	1.18	20
PCA-Cos	67.9%	0.73	20	71.6%	1.37	23	74.9%	1.25	19	75.8%	1.07	22

Table 4.20: Performance of KNN on email_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	88.4%	2.01	465	89.6%	2.04	465	83.8%	2.21	465	85.8%	3.19	465
PCA-Cov	95.6%	1.35	57	92.7%	1.81	52	93.5%	2.08	67	91.7%	1.06	63
PCA-Corr	96.1%	1.09	53	94.9%	1.26	42	94.4%	1.37	48	95.1%	0.23	46
PCA-Cos	97.2%	0.91	31	95.3%	1.07	33	96.1%	1.15	21	96.8%	0.48	32

Table 4.21: Performance of KNN on 20news_v2 dataset.

Methods	TF			TP			TF-IDF			TP-CSDF		
	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F	Av.	Std.	F
Full features	63.7%	3.04	2591	68.4%	3.41	2591	62.8%	3.23	2591	70.6%	2.91	2591
PCA-Cov	73.7%	2.19	67	72.6%	1.92	65	72.9%	2.18	93	73.4%	1.73	121
PCA-Corr	74.9%	1.81	43	73.3%	1.32	58	74.7%	1.63	38	74.6%	1.21	52
PCA-Cos	76.6%	1.28	33	73.6%	1.43	45	77.3%	1.54	86	76.4%	0.92	31

Table 4.22: Performance of KNN on technical website features dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	82.4%	3.62	47	82.9%	3.42	47
PCA-Cov	91.5%	1.33	51	92.6%	1.66	36
PCA-Corr	93.5%	0.52	41	93.6%	0.45	27
PCA-Cos	94.9%	0.31	29	96.4%	0.11	18

Table 4.23: Performance of KNN on musk dataset.

Methods	F			NF		
	Av.	Std.	F	Av.	Std.	F
Full features	83.3%	3.23	166	82.7%	3.14	166
PCA-Cov	91.2%	2.14	51	91.7%	1.92	46
PCA-Corr	94.4%	1.23	32	92.7%	0.63	36
PCA-Cos	95.8%	0.91	23	94.9%	0.37	31

Table 4.24: Performance of KNN on the malicious dataset.

Methods	**F**			**NF**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	91.3%	3.22	80	92.8%	3.14	80
PCA-Cov	92.8%	2.07	42	91.7%	1.45	40
PCA-Corr	94.3%	1.37	31	95.2%	0.65	31
PCA-Cos	95.6%	2.24	26	96.8%	0.16	23

Statistical Test and P-values: In order to assess whether the performance differences among the methods are statistically significant, we applied Wilcoxon's rank sum test, which is a non-parametric method to determine if two sets of data are significantly different from each other, to compare PCA-cov against PCA-corr, PCA-cov against PCA-cos, and PCA-cos against PCA-corr in terms of classification accuracy and the number of selected features. Table 4.25 shows the P-values for these pair comparisons, which demonstrate that, in terms of the number of required features to achieve the best classification performance, PCA-cos and PCA-corr significantly outperformed PCA-cov, and PCA-cos outperformed PCA-corr but not in a significant manner. In addition, PCA-cos achieved higher accuracy than PCA-cov, but not significantly.

Table 4.25: Statistical test results.

	Accuracy (p-value)	**Number of required features (p-value)**
PCA-Cov vs. PCA-Cos	P = 0.0549	P = 7.2471e-04
PCA-Cov vs. PCA-Corr	P = 0.05271	P = 1.6434e-06
PCA-Cosvs. PCA-Corr	P = 0.4715	P = 0.1432

4.7 Summary

Feature dimensionality reduction is a very crucial pre-processing step in document classification in high-dimensional feature space. Standard PCA based on the singular value decomposition of covariance matrix is a very common method for feature

extraction and dimensionality reduction. This chapter has presented an experimental investigation to improve the performance of standard PCA method by using cosine similarity matrix and correlation matrix to replace covariance matrix in PCA for feature dimensionality reduction. It was tested in a phishing email detection application and document classification using six other datasets. Preliminary experimental results have demonstrated the advantages of the proposed method over the standard PCA in terms of the number of required features. Further tests in other applications would be conducted in future investigations.

As mentioned above, PCA-Cov, PCA-Cos, and PCA-Corr methods combine original data into new features which are difficult to interpret. This can be regarded as a drawback with need for interpretability in different disciplines. Some different approaches would be trying to find the minimal number of features with capability to interpret. The next chapter proposes two hybrid approaches to feature subset selection for document classification in high-dimensional feature space.

Chapter 5

Hybrid Approaches to Feature Subset Selection for Document Classification in High-Dimensional Feature Space

5.1 Introduction

The previous chapter presented an improved PCA approach to feature extraction and dimensionality reduction, which lacks interpretability. Interpretability in data mining is an important issue in many fields such as social sciences and medicine (Katuwal & Chen, 2016; Doshi-Velez, & Kim, 2017). Many traditional methods for dimensionality reduction can achieve high accuracy in data classification, but their results are usually difficult to interpret (Zhou & Gan, 2008). As discussed in section 2.4, feature subset selection finds a subset of features with high predictive power. In general, there are three methods for performance evaluation of potential feature subsets: filter approach, wrapper approach, and embedded approach. Wrapper methods can be impractical when the number of features available for selection and the number of observation points are too large, whilst the computational cost of filter methods is much less than wrapper methods for large feature datasets. However, wrapper methods are usually more accurate than filter methods (Zhen et al., 2016; Dhote et al., 2015; Nam & Quoc, 2015; Tang et al., 2014; Gan et al., 2014; Guyon & Elisseeff, 2003; Dash & Liu, 1997). Embedded approach searches locally for features that allow better local discrimination. It uses independent criteria to decide on the best subsets for given cardinality, in which learning algorithms are usually used to select the final best

subset. Embedded approach interacts with learning algorithms at a lower computational cost than the wrapper approach (Dash & Liu, 1997; Guyon & Elisseeff, 2003; Cha, 2007).

As discussed in section 2.4.1, there are many different criteria applied to filter-based feature selection and dimensionality reduction, such as distance or similarity/dissimilarity criteria (Cha, 2007; Saeys et al., 2007), information theory measures, and statistics measures. Distance or similarity/dissimilarity criteria have been applied for feature selection in many application areas, such as pattern recognition, information retrieval and detection of phishing emails and websites (Saeys et al., 2007). However, these measures are easily affected by noise or outlier data. Information theory measures have been widely used for feature selection, such as information gain (IG) (Chandrasekaran et al., 2006; Gomez et al., 2012) and maximum relevance and minimum redundancy (mRMR) (Peng et al., 2005). Recently, ensemble feature selection approach (Wang et al., 2012; Fahad et al., 2014; Aldehim, 2015) has received much attention, which is commonly used for combining multiple models or methods to form a single effective method. Hybrid approach is employed to take the advantages of the strengths of both filter and wrapper methods to further reduce the irrelevant features without degrading the accuracy. Zheng et al. (2018) proposed a hybrid filter-wrapper feature subset selection algorithm called the Maximum Spearman Minimum Covariance Cuckoo Search (MSMCCS). This method combines the efficiency of filters with the greater accuracy of wrappers. Fazil & Abulaish (2018) proposed a hybrid approach for automated detection of spammers in Twitter.

This chapter proposes two hybrid methods for feature subset selection, consisting of two stages to select a relevant subset of features. The first stage selects feature subsets based on the union or intersection of features selected according to distance or similarity measures (unsupervised) and mutual information measures (supervised). The second stage employs a wrapper approach on the selected features to further reduce the feature dimensionality and hopefully improve classification accuracy as well. Experiments were conducted with the performance of the proposed methods evaluated by comparison with the individual filter approaches and the full wrapper approach.

5.2 Feature Selection Approaches

5.2.1 Filter Approaches

A filter method selects a subset of features or ranks features based on general characteristics of the features, independently without including any classification methods (Zhen et al., 2016; Quoc, 2015; Dhote et al., 2015; Gan et al., 2014; Dash & Liu, 1997). There are two main types of filter-based feature selection: unsupervised and supervised. Unsupervised methods select features according to distance or similarity/dissimilarity between features whilst supervised methods select features according to their correlation or relevance with class labels.

1) Unsupervised Filter Approaches

Given two vectors of features, $\mathbf{x}=\{x_1, x_2, \ldots, x_n\}$ and $\mathbf{y}=\{y_1, y_2, \ldots, y_n\}$, where n is the number of observations, various evaluation criteria for feature selection can be defined without using the corresponding class information, such as those defined below.

- Euclidean Distance (Yang & Pedersen, 1997).

$$Euc\_dis(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{5.1}$$

- Hamming Distance (Kalbhor et al., 2013).

$$Ham\_dis(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n}(x_i \neq y_i) \tag{5.2}$$

- City Block Distance (Piramuthu, 2004).

$$Cityblock\_dis(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{n}| x_i - y_i | \tag{5.3}$$

- Hausdorff Distance (Aggarwal & Zhai, 2012).

$$H(\mathbf{x},\mathbf{y}) = \max(h(\mathbf{x},\mathbf{y}), h(\mathbf{y},\mathbf{x})) \tag{5.4}$$

where

$$h(\mathbf{x},\mathbf{y}) = \max_{x_i \in \mathbf{x}} \min_{y_i \in \mathbf{y}} | x_i - y_i | \tag{5.5}$$

2) Supervised Filter Approaches

When class information is available, various evaluation criteria for feature selection can be defined based on information theory. The two criteria adopted in this experiment are information gain and Minimum Redundancy and Maximum Relevance (mRMR), which has been described in detail in Section 2.4.

5.2.2 Wrapper Approaches

Wrapper approach is a very common technique for feature subset selection, in which classifiers usually built up via machine learning are used for the evaluation of potential feature subsets, aiming to improve the classification performance (Gan et al., 2014; Karegowda et al., 2010; Saeys et al., 2007). However, it can be unrealistic when the

number of features available for selection and the number of sample points are too large, especially when the classifier training is computationally expensive (Martin-Smith et al., 2016). As discussed in section 2.4.2, this approach suffers from overfitting as well.

5.3 Hybrid Approaches to Feature Subset Selection

Using a specific classifier, the wrapper approach compares cross-validation classification accuracies obtained with the potential feature subsets. Wrapper-based feature selection is vulnerable to overfitting due to its comprehensive search of the feature space and evaluation by a classifier constructed by machine learning. Hence, seeking reliable features using the wrapper method is sometimes impossible for datasets with a large feature space. In order to take the advantage of this highly accurate method and also to reduce its computational cost, hybrid approaches combining the advantages of the filter approach and the wrapper approach have been developed in recent years with different motivations and different search methods (Gan et al., 2014). The hybrid strategy is adopted in the two methods proposed in this chapter, which consists of two stages. In the first stage, union and intersection among features selected by six filter methods are constructed to reduce unrelated and redundant features before the application of the costly wrapper method. The second stage further reduces the feature space considerably by using the wrapper method.

As one of the baseline methods for comparison, the wrapper approach can be applied to the original full feature dataset. Figure 5.1 illustrates how the wrapper approach works. Three well-known classifiers with different structures and classification

mechanisms are used in this study: support vector machine (SVM) (Smola & Schölkopf, 2004; Vapnik, 2000), linear discriminant analysis (LDA) (Belhumeur et al., 1997) and K-nearest neighbour (KNN) (Altman, 1992).

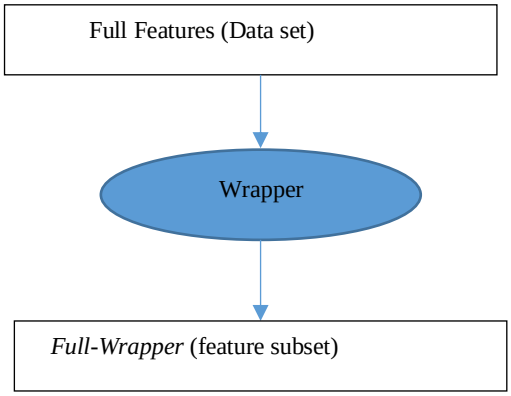

Figure 5.1: Wrapper approach

1) The Proposed Hybrid Approach 1

The first hybrid approach proposed in this chapter employs four (unsupervised) distance or similarity measures, i.e., Euclidean, Hamming, City Block, and Hausdorff, defined in (5.1), (5.2), (5.3) and (5.4) respectively, to compute the similarity between each pair of feature vectors in the training dataset, generating four different similarity matrixes of the training data, which are used to select *m* most useful features. For each criterion, the best value of *m* is chosen by cross-validation. The union of the four subsets of the selected features generates a combined feature subset called C1 at this first stage. In the second stage, the wrapper approach is applied to C1 to find the best feature subset H1. For comparison purposes, both C1 and H1 will be evaluated on test datasets. Figure 5.2 illustrates how this approach works.

2) The Proposed Hybrid Approach 2

The second hybrid approach proposed in this chapter employs six filter (unsupervised and supervised) criteria. Four (unsupervised) distance or similarity measures, i.e., Euclidean, Hamming, City Block, and Hausdorff, defined in (5.1), (5.2), (5.3), and (5.4) respectively, compute the similarity between each pair of feature vectors in the training dataset, generating four different similarity matrixes of the training data, which are used to select *m* most useful features. Two (supervised) mutual information based criteria IG and mRMR, defined in section 2.4.1.1 and 2.4.1.2 respectively, select a compact set of relevant features from the training dataset, generating two different matrixes of the training dataset, which are used to select *m* most useful features. For each criterion, the best value of *m* is chosen by cross-validation. The union of the two subset results, which are generated from the intersection of the four sets of features selected by unsupervised filters and the intersection of the two sets of features selected by supervised filters, forms a combined feature subset called C2 at this first stage. In the second stage, the wrapper approach is applied to C2 to find an optimal feature subset H2. For comparison purposes, both C2 and H2 will be evaluated on test datasets. Figure 5.3 illustrates how this approach works.

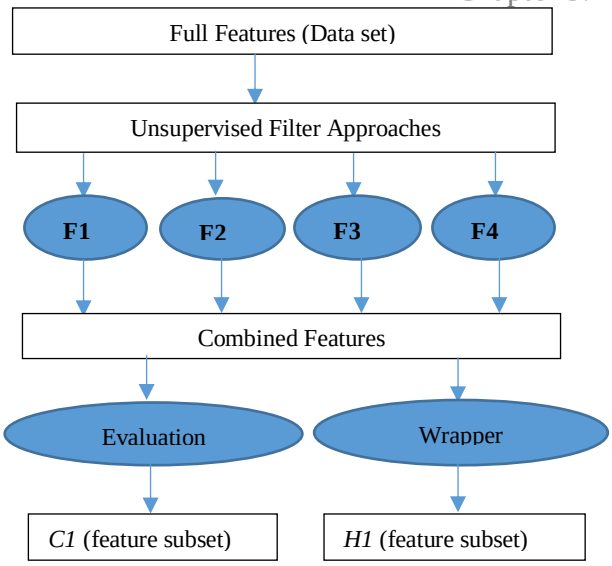

Figure 5.2: The proposed hybrid approach 1

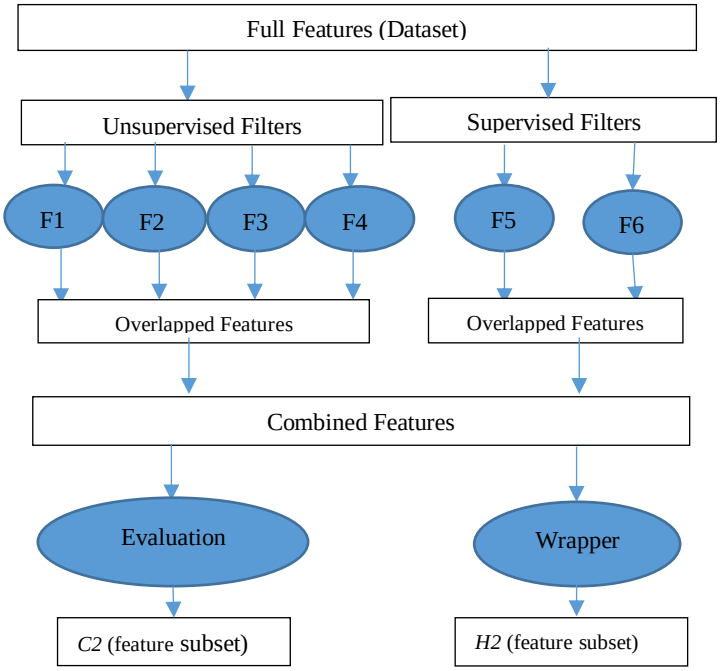

Figure 5.3: The proposed hybrid approach 2

5.4 Experimental Results

5.4.1 Datasets

Seven datasets were used in the experiments (email_v1, email_v2, Reuters-21578, 20news_v2, Musk, technical website features, and self-drive intrusion detection). The full descriptions of these datasets are presented in chapter 3.

5.4.2 Experiment Procedure

The full descriptions of the experiment procedure are in section 3.3

The proposed approaches were applied to select the best number of discriminate features in the sense that the highest cross-validation (5-folds) performance was achieved, and they were evaluated by employing classifiers LDA, SVM, and KNN with the selected features on the test datasets.

5.4.3 Results

Classification accuracy and the corresponding number of required features: Figures 5.4–5.9 show the mean (over TF, TP, TF-IDF and TP-CSDF) of the classification accuracy, standard deviation on the testing datasets and the mean of the corresponding number of required features using LDA, SVM, and KNN (the best value of K is chosen by cross-validation) respectively, by comparing among six filter approaches, the combined filter approaches, the proposed hybrid approaches, and the full wrapper approach. It can be seen that the two proposed hybrid approaches achieved competitive accuracy with a significantly smaller number of required features compared to the full wrapper approach.

Computational time: Figures 5.10–5.12 demonstrate the mean of the time spent (in seconds) by various feature selection methods. As expected, the two proposed hybrid methods spent much less time than the full wrapper approach.

Interpretability of the selected features: Table 5.1 shows the top five candidate features of the emails dataset, from each of the selected feature subsets H1, H2, and Full-Wrapper. As a matter of common sense, it seems that the top ten terms selected by the two proposed hybrid approaches are more relevant to phishing, such as 'attached' and 'click'. Similar interpretability of the selected features was also observed on the news topics datasets. It seems that the terms (features) selected by using the hybrid approaches are more interpretable than those by the full wrapper approach.

***Statistical significance test*:** In order to assess whether the performance differences among the methods are statistically significant, we applied T-test, a parametric method, and Wilcoxon's rank-sum test, a non-parametric method, to determine whether two sets of performance data are significantly different from each other. The statistical tests were conducted to compare H1 against C1, H1 against C2, H2 against C1, H2 against C2, H1 against Full-Wrapper, H2 against Full-Wrapper, and H1 against H2, in terms of classification accuracy, time consumed, and the number of selected features, respectively. Tables 5.2 and 5.3 show the P-values for these pair comparisons, which demonstrate that H1 and H2 significantly outperformed C1 and C2 in terms of classification accuracy and number of selected features, and H1 and H2 outperformed Full-Wrapper in terms of time consumed and number of selected features. In addition, H1 and H2 sometimes achieved higher accuracy than Full-Wrapper, but the difference is not significant. Finally, H1 consumed significantly less time than H2.

The stability of the selected features is desirable in practical feature selection applications, which has not been investigated yet in this chapter. It is also desirable to compare with more other hybrid feature selection methods, such as those proposed in Gan et al. (2014) and Aldehim (2015), to further evaluate the proposed methods. These would be considered in our future work in this line of research.

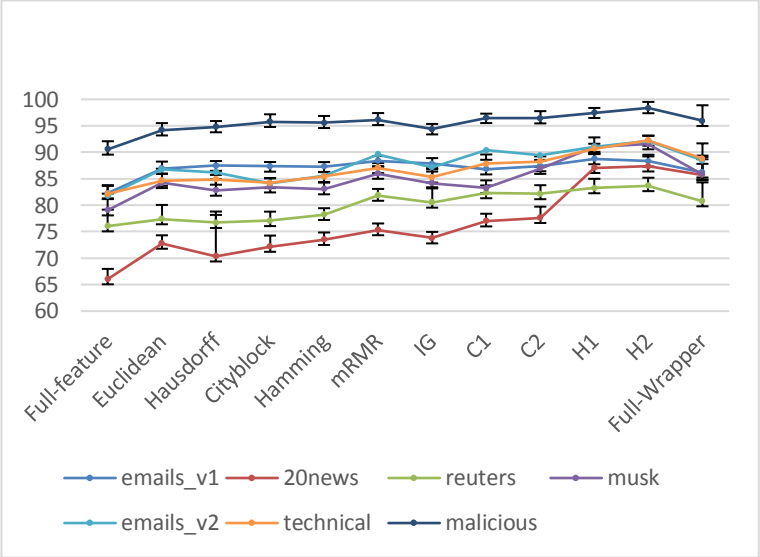

Figure 5.4: Accuracy and standard deviation of LDA with selected feature subsets on the seven datasets.

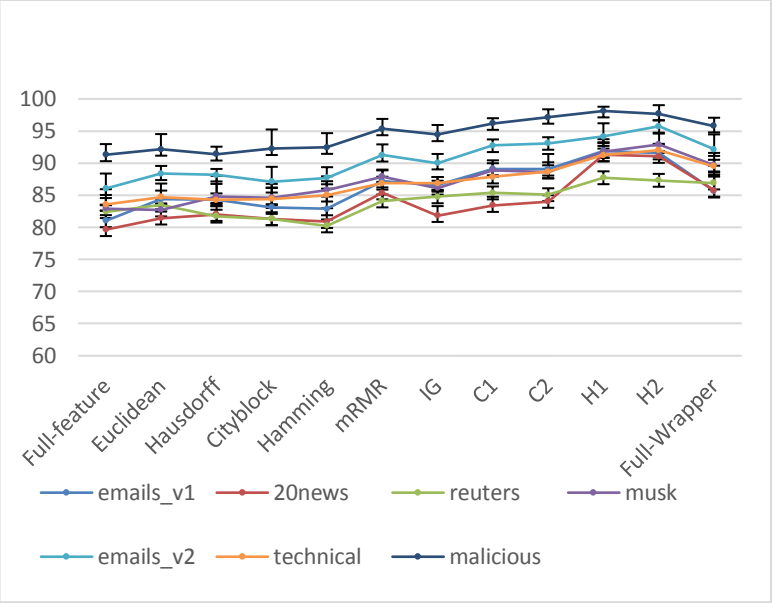

Figure 5.5: Accuracy and standard deviation of SVM with selected feature subsets on the seven datasets.

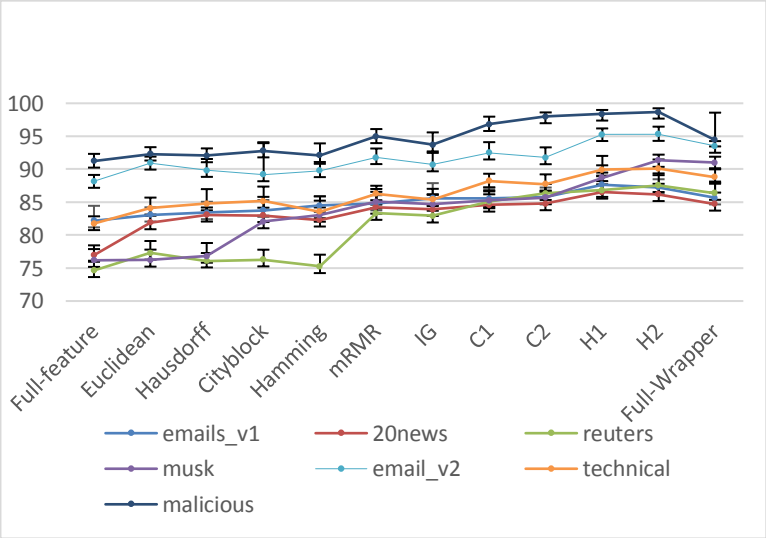

Figure 5.6: Accuracy and standard deviation of KNN with selected feature subsets on the seven datasets.

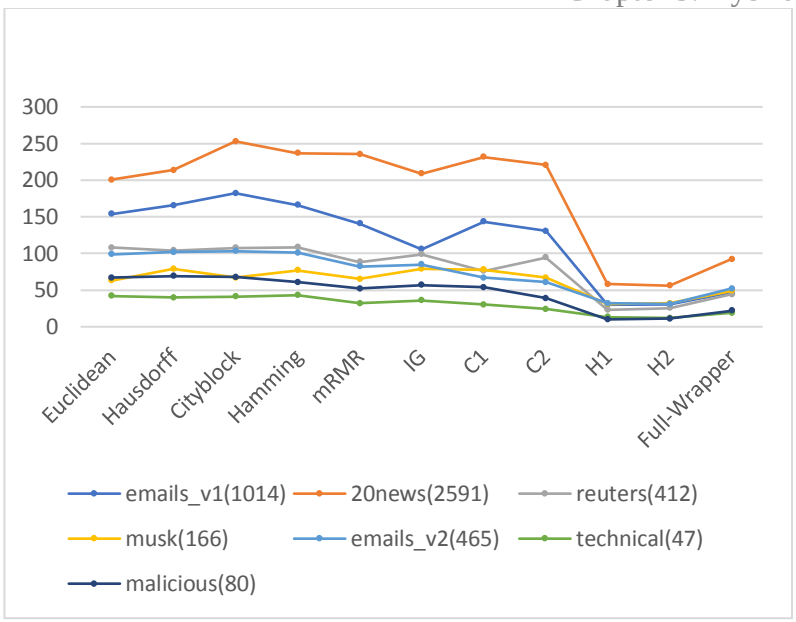

Figure 5.7: Number of selected features for LDA on the seven datasets.

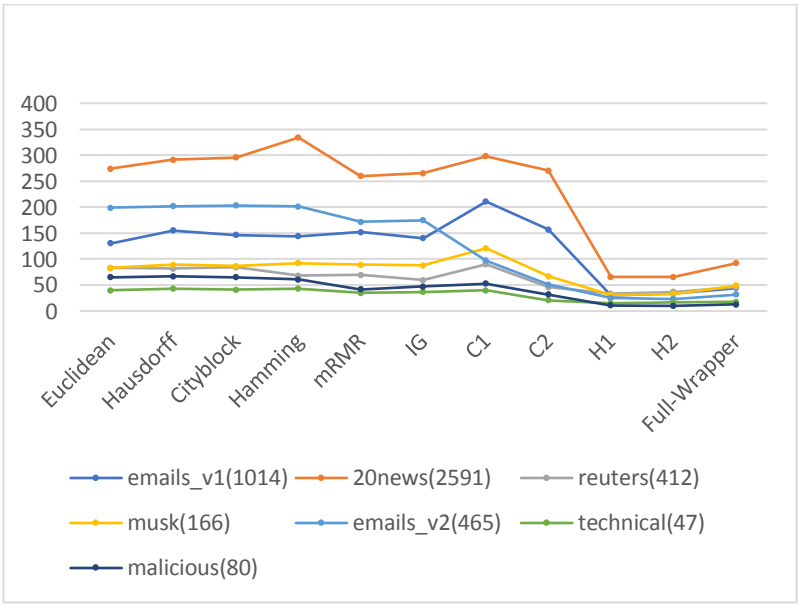

Figure 5.8: Number of selected features for SVM on the seven datasets.

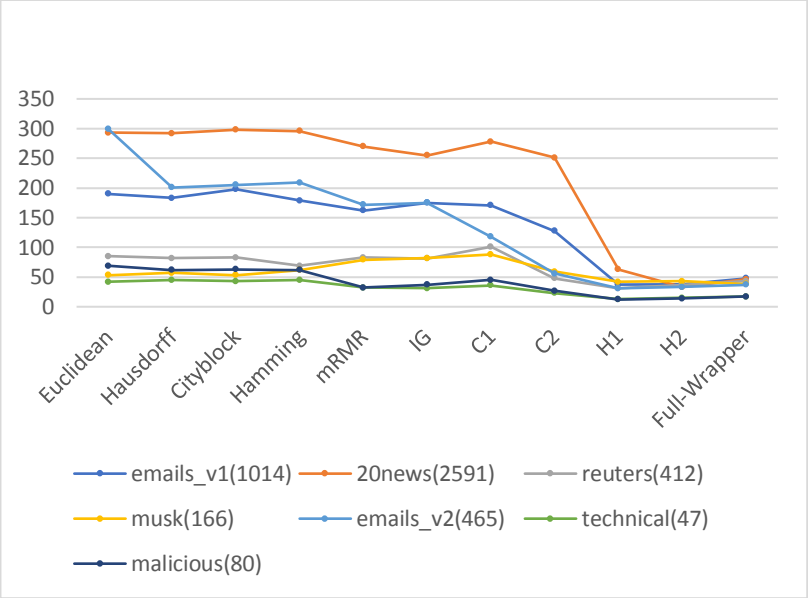

Figure 5.9: Number of selected features for KNN on the seven datasets

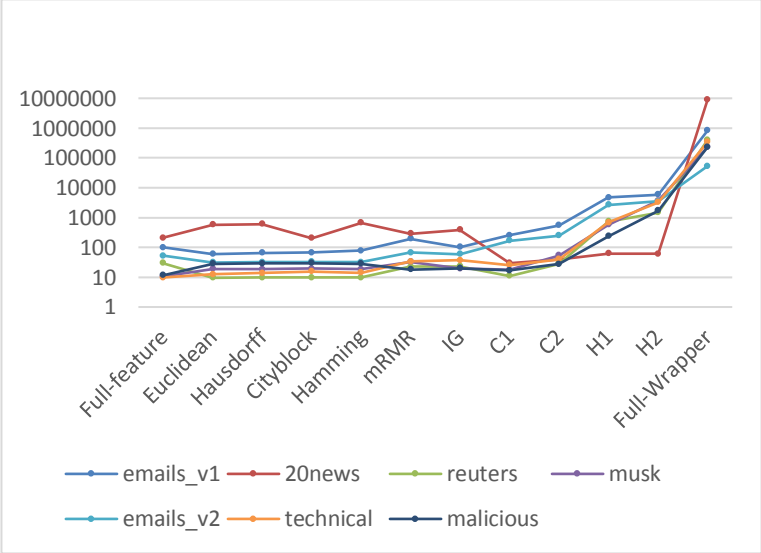

Figure 5.10: Time spent using LDA on the seven datasets.

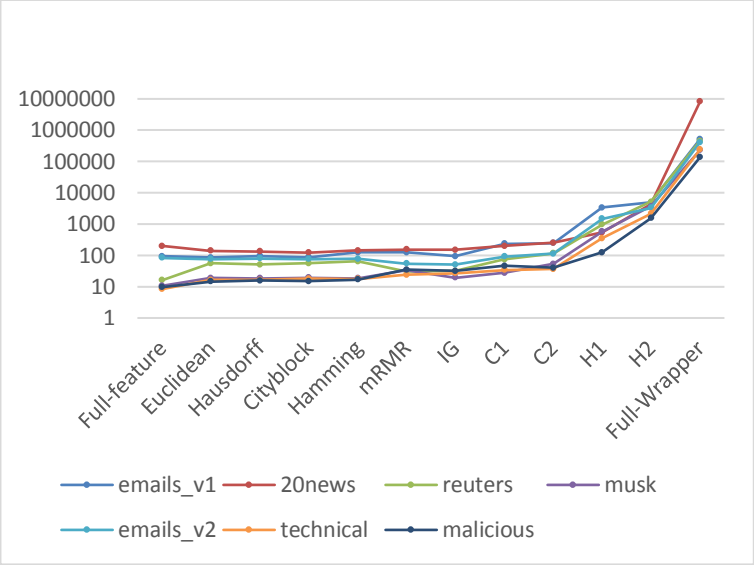

Figure 5.11: Time spent using SVM on the seven datasets.

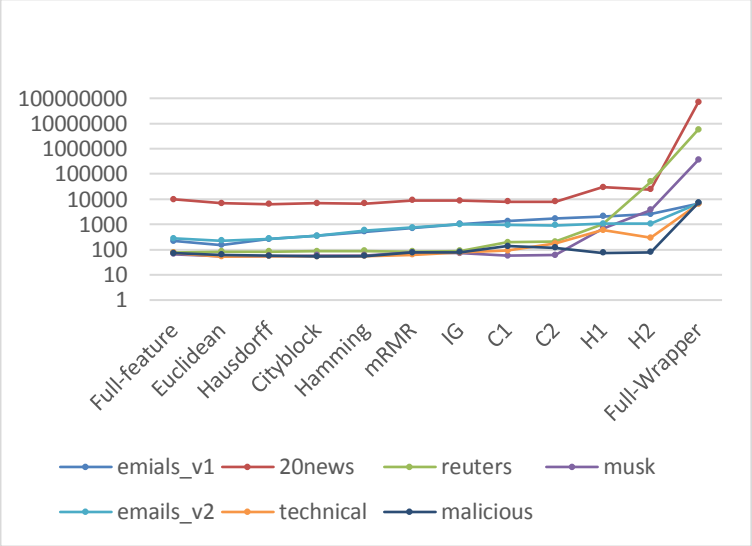

Figure 5.12: Time spent using KNN on the seven datasets.

Table 5.1: Top five terms selected from the emails dataset

H1	H2	Full-Wrapper
attached	Online	Shop
Bank	Update	Original
Online	Attached	Effective
Click	Deal	Resources
www	Link	Company

Table 5.2: Statistical test results (t-test)

Method pair	Accuracy (p-value)	No.of required features (p-value)	Time consumed (p-value)
H1 vs C1	P = 0.0783	P = 2.3298e-06	P = 0.5421
H1 vs C2	P = 0.0423	P = 7.162e-05	P = 0.164
H2 vs C1	P = 0.0258	P = 3.7452e-06	P = 0.2746
H2 vs C2	P = 0.0364	P = 7.8182e-05	P = 0.3567
H1 vs Full	P = 0.345	P = 4.3362e-07	P = 4.2516e-07
H2 vs Full	P = 0.4765	P = 6.4249e-05	P = 3.1142e-07
H1 vs H2	P = 0.6385	P = 0.2423	P =0.0050

Table 5.3: Statistical test results (rank-sum)

Method pair	Accuracy (p-value)	No. of required features (p-value)	Time Consumed (p-value)
H1 vs C1	P = 0.0034	P = 2.4321e-05	P = 0.6121
H1 vs C2	P = 0.0043	P = 2.2456e-05	P = 0.3151
H2 vs C1	P = 0.0307	P = 2.3287e-04	P = 0.5621
H2 vs C2	P = 0.0347	P = 3.2567e-04	P = 0.3421
H1 vs Full	P = 0.314	P = 3.3546e-06	P = 2.3467e-04
H2 vs Full	P = 0.4123	P = 4.3567e-04	P = 2.1361e-04
H1 vs H2	P = 0.3566	P = 0.2135	P = 0.0031

5.5 Summary and Discussion

Several hybrid feature subset selection methods have been developed in recent years, with different motivations and different search methods. This chapter proposes two hybrid approaches to feature subset selection based on the combination of unsupervised and supervised filter approaches and the wrapper approach for document classification in high-dimensional feature space. They were tested on seven datasets from different resources and with different properties. Preliminary experimental results have demonstrated the advantages of the proposed methods over individual filter approaches and the full wrapper approach in terms of classification accuracy, the number of required features, consumed time, and interpretability. Furthermore, the first hybrid approach H1 is better than the second approach H2 in terms of time consumed.

We observed that SVM is vulnerable to overfitting with the wrapper approach working on full features. This can be illustrated by the fact that the non-linear classification method with the wrapper method is not sufficient, particularly with complicated data space.

In the next two chapters we will focus on the role of deep learning in automatic feature learning in big data domain and propose a new approach for representation learning and dimensionality reduction for document classification to achieve high performance and mitigate the problems that we observed during our study on deep learning, such as overfitting and vanishing/exploding gradients.

Chapter 6

Class-Specific Pre-Trained Sparse Autoencoders for Learning Effective Features for Document Classification

6.1 Introduction

Big data analysis through deep learning has attracted much attention in recent years (Nigam et al., 2011; Najafabadi et al., 2015; Xiao et al., 2018). Representation learning is a very important processing step in pattern recognition in high-dimensional feature space (Ng, 2011; Salakhutdinov & Hinton, 2009; Chandrasekaran et al., 2006). Supervised learning is one of the most influential approaches to artificial intelligence in different scientific disciplines such as email classification, text categorization, and information retrieval. It can do very well if a good feature representation is given (Liu et al., 2016; Najafabadi et al., 2015; Ng, 2011).

Unsupervised learning is usually required for feature learning, such as feature learning using the restricted Boltzmann machine (RBM) (Salakhutdinov & Hinton, 2009), sparse autoencoder (Lee et al., 2010, Abdulhussain and Gan, 2016), stacked autoencoder (SAE) (Gehring et al., 2013, Zhou et al., 2015), and hierarchical convolutional sparse autoencoder (Han et al., 2017).

PCA is a very conventional approach for feature extraction and dimensionality reduction (Martínez et al., 2001), which combines features by using orthogonal transformation (Liu et al., 2010; Yu et al., 2009). It has been used in almost all

scientific disciplines. However, it is a linear transformation approach. That means it cannot be applied in the layers of deep network structure because the compositions of linear transformations produce another linear transformation (Ng, 2011; Salakhutdinov and Hinton, 2009). An autoencoder is a symmetrical neural network that is employed to learn the features of a dataset and conduct dimensionality reduction in an unsupervised manner. A non-linear autoencoder transformation approach can discover more complex multi-model structure in the data (Lee et al., 2010; Zhang et al., 2015). Many studies applied sparse autoencoder for intrusion detection (Deng et al., 2010; Javaid et al., 2016; Meng et al., 2017b).

In this chapter, we propose a class-specific (supervised) pre-trained approach by employing sparse autoencoder algorithm (unsupervised) to learn features by encoding training samples from each class respectively. The features learnt for different classes are merged later into a whole feature vector. Experiments were conducted with performance evaluated by comparison with unsupervised training sparse autoencoder.

6.2 Sparse Autoencoder

An autoencoder is an unsupervised neural network trained by using back propagation via gradient descend algorithms, which learns a non-linear approximation of an identity function as described in section 2.6.1 (Abdulhussain & Gan, 2016; Zhang et al., 2015; Ng, 2011). A sparse autoencoder can be obtained by using sparse regularisation in its learning objective function. An autoencoder may have three or more hidden layers, but for simplicity, an autoencoder with just a single hidden layer is described in detail as follows.

The connection weights and bias parameters can be denoted as $w = [vectorised\ W_1; vectorised\ W_2; b_1; b_2]$, where $W_1 \in R^{K \times N}$ is the encoding weight matrix and $W_2 \in R^{N \times K}$ is the decoding weight matrix, $b_1 \in R^K$ is the encoding bias vector, and $b_2 \in R^N$ is the decoding bias vector.

For a training dataset, let the output matrix of the autoencoder be $O = [o^1, o^2, ..., o^m]$; this is intended to be the reconstruction of the input matrix $X = [x^1, x^2, ..., x^m]$, where $o^i \in R^N$ and $x^i \in R^N$ are the output vector and input vector of the autoencoder respectively, and *m* is the number of training samples. Correspondingly, let the hidden output matrix be $H = [h^1, h^2, ..., h^m]$, where $h^i \in R^K$ is the hidden output vector of the autoencoder to be used as a feature vector for feature learning tasks.

For the i^{th} sample, the hidden output vector is defined as

$$h^i = g\left(W_1 x^i + b_1\right) \tag{6.1}$$

And the output is defined by

$$o^i = g\left(W_2 h^i + b_2\right) \tag{6.2}$$

where g(*x*) is a sigmoid logistic function $(1 + exp(-x))^{-1}$.

For a sparse autoencoder, the learning objective function is defined as follows:

$$J_{sparse}(W) = \frac{1}{2m}\sum_{i=1}^{m}\left\|x^i - o^i\right\|^2 + \frac{\lambda}{2}\|W\|^2 + \beta\sum_{j=1}^{K} KL\left(p \parallel \hat{p}_j\right) \tag{6.3}$$

where *p* is the sparsity parameter, and $\hat{P}_j$ is the average output of the j^{th} hidden node, averaged over all the samples, i.e.:

$$\hat{p}_j = \frac{1}{m}\sum_{i=1}^{m} h_j^i \tag{6.4}$$

λ is the coefficient for L_2 regularisation (weight decay), and β is the coefficient for sparsity control, which is defined by the Kullback-Leibler divergence:

$$KL\left(p \parallel \hat{p}_j\right) = p\log\frac{p}{\hat{p}_j} + (1-p)\log\frac{1-p}{1-\hat{p}_j} \tag{6.5}$$

The learning rule for updating the weight vector W (containing W_1, W_2, b_1 and b_2) is error back-propagation based on gradient descent, i.e., $\Delta w = -\eta \cdot wgrad$. The error gradients with respect to W_1, W_2, b_1 and b_2 are derived as follows respectively.

$$W_1 grad = (W_2^T (O - X) + \beta\left(-\frac{p}{\hat{p}_j} + \frac{1-p}{1-\hat{p}_j}\right) I^T)\ .* g'(H) X^T / m + \lambda W_1 \tag{6.6}$$

$$b_1 grad = ((W_2^T (O - X) + \beta\left(-\frac{p}{\hat{p}_j} + \frac{1-p}{1-\hat{p}_j}\right) I^T).* g'(H) I / m \tag{6.7}$$

$$W_2 grad = ((O - X) H^T) / m + \lambda W_2 \tag{6.8}$$

$$b_2 grad = (O - X) I / m \tag{6.9}$$

where $g'(H) = g(H).*[1 - g(H)]$ is the derivative of the sigmoid logistic function, $I = [1,1,\ldots,1]^T$ is a vector of size m and (.*) represents element-wise multiplication

6.3 Proposed Approach

The proposed approach trains a sparse autoencoder for each class separately, aiming to discover interesting structure of data. This offers the opportunity to learn the effective features of each class separately. Figure 6.1 illustrates how it works, in which each

sub-encoder is applied to generate k_2 features from each class, with k_2 equal to K (the number of total features) divided by the number of classes. The features from all classes will be combined to form the input feature vector for pattern classification. Two well-known classifiers were used in this study. They are linear discriminant analysis (LDA) (Gomez et al., 2012), and support vector machine (SVM) (Smola and Schölkopf, 2004; Vapnik, 2013).

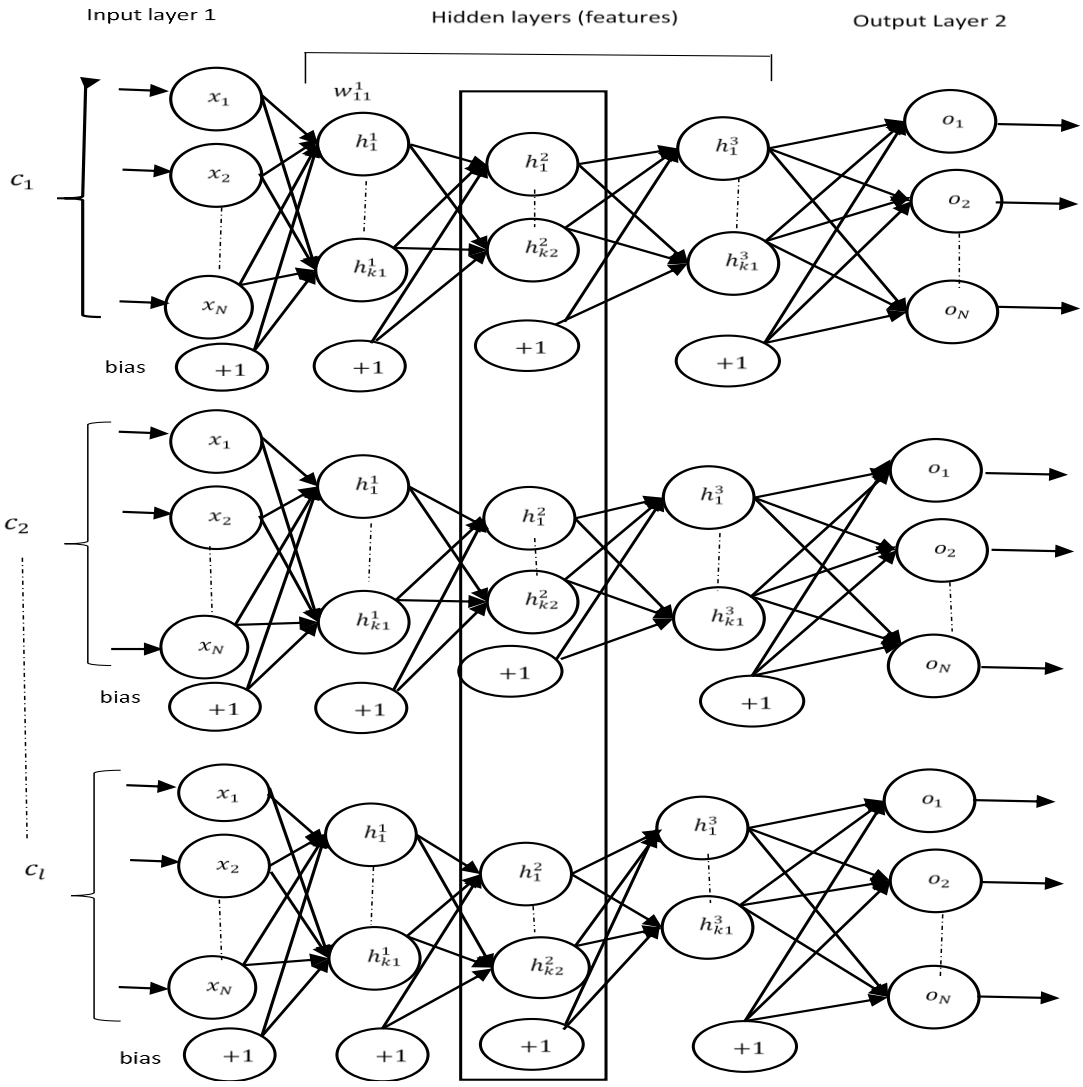

Figure 6.1: Class-specific pre-trained sparse autoencoder network.

6.4 Experimental Results

6.4.1 Datasets

Six datasets were used in the experiments (email_v1, email_v2, Reuters-21578, 20news_v2, Musk, and technical website features). The full descriptions of these datasets are presented in chapter 3.

6.4.2 Experiment Procedure

The full descriptions of the experiment procedure are presented in section 3.3. The proposed approach was used to extract higher-level features or reduce the feature dimensionality and was evaluated by employing classifiers LDA and SVM. For applying the proposed approach, the training set was further divided into an estimation set and a validation set, and cross-validation (5-folds) was adopted for training the autoencoders and classifiers and determining the hyper-parameters such as p, λ, β, K (number of hidden layers and nodes).

6.4.3 Results

Classification accuracy, standard deviation and the corresponding number of required features: Tables 6.1-6.12 show the classification accuracy and standard deviation of LDA and SVM with the specified number of features (100 and 50) achieved by five methods (Full-features, sparse autoencoder (SAE-100, SAE-50), and class-specific pre-trained sparse autoencoder (CSPT_SAE-100 and CSPT_SAE-50). Figures 6.2 and 6.3 show the comparison of the performance among the methods on six datasets. It can be seen that the proposed CSPT_SAE achieved higher test

classification accuracy with the same number of required features than SAE and Full-features.

Table 6.1: Performance of LDA on email_v1 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	86.9%	2.68	1014	85.6%	1.47	1014	85.7%	1.95	1014
SAE	89.7%	1.78	100	89.9%	1.56	100	88.6%	0.77	100
	89.2%	1.69	50	89.9%	0.97	50	87.4%	1.08	50
CSPT_SAE	91.7%	0.85	100	92.7%	0.85	100	90.9%	0.94	100
	92.9%	0.64	50	92.9%	0.64	50	91.9%	0.73	50

Table 6.2: Performance of SVM on email_v1 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	96.9%	2.28	1014	90.6%	1.47	1014	89.7%	1.65	1014
SAE	88.8%	1.98	100	87.9%	1.56	100	87.6%	0.97	100
	87.9%	1.29	50	89.8%	0.97	50	88.4%	1.68	50
CSPT_SAE	93.7%	0.65	100	91.6%	0.85	100	90.2%	0.74	100
	93.9%	0.84	50	91.8%	0.94	50	90.9%	0.89	50

Table 6.3: Performance of LDA on email_v2 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	95.9%	2.35	465	85.6%	1.47	465	85.7%	1.95	465
SAE	91.2%	1.36	100	89.9%	1.56	100	88.6%	0.77	100
	89.3%	1.57	50	89.9%	0.97	50	87.4%	1.08	50
CSPT_SAE	96.9%	0.57	100	92.7%	0.85	100	91.5%	0.94	100
	93.8%	0.54	50	92.9%	0.64	50	91.7%	0.73	50

Table 6.4: Performance of SVM on email_v2 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	96.9%	1.86	465	93.6%	1.67	465	90.4%	1.65	465
SAE	94.6%	2.08	100	93.9%	1.26	100	89.6%	0.87	100
	90.5%	1.38	50	89.9%	0.87	50	89.4%	1.08	50
CSPT_SAE	97.9%	0.65	100	95.7%	0.75	100	94.9%	0.74	100
	96.6%	0.54	50	93.9%	0.54	50	93.5%	0.53	50

Table 6.5: Performance of LDA on 20news_v2 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	84.9%	1.16	2591	79.6%	1.87	2591	70.4%	1.25	2591
SAE	84.6%	3.73	100	79.8%	1.66	100	71.6%	0.77	100
	85.5%	2.25	50	72.7%	0.97	50	70.4%	1.03	50
CSPT_SAE	86.9%	0.75	100	77.8%	0.85	100	75.8%	0.74	100
	85.8%	0.64	50	76.9%	0.64	50	76.5%	0.53	50

Table 6.6: Performance of SVM on 20news_v2 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	89.5%	1.34	2591	81.6%	2.87	2591	79.4%	1.35	2591
SAE	85.6%	3.63	100	76.8%	1.26	100	77.6%	0.97	100
	84.5%	2.45	50	73.5%	1.87	50	70.4%	1.53	50
CSPT_SAE	86.4%	0.95	100	77.9%	1.55	100	79.8%	0.94	100
	84.7%	0.54	50	76.3%	0.84	50	78.5%	0.83	50

Table 6.7: Performance of LDA on Reuters-21578 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	83.9%	4.34	421	83.6%	4.06	421	80.7%	4.75	421
SAE	80.7%	1.24	100	82.8%	3.56	100	82.1%	2.77	100
	81.2%	1.67	50	81.7%	1.76	50	81.4%	3.08	50
CSPT_SAE	81.7%	1.16	100	81.2%	2.87	100	80.8%	2.94	100
	82.9%	1.87	50	82.6%	2.45	50	81.7%	2.73	50

Table 6.8: Performance of SVM on Reuters-21578 dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	87.7%	5.34	421	85.6%	4.36	421	80.7%	3.55	421
SAE	89.6%	2.24	100	85.8%	3.86	100	85.1%	1.57	100
	88.2%	1.57	50	84.7%	2.76	50	84.4%	1.06	50
CSPT_SAE	90.7%	1.46	100	88.2%	2.87	100	87.8%	1.44	100
	89.9%	1.67	50	86.8%	2.45	50	88.7%	1.36	50

Table 6.9: Performance of LDA on musk dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	85.8%	2.02	166	85.6%	3.56	166	81.4%	3.85	166
SAE	95.7%	1.45	100	88.8%	2.78	100	88.6%	2.87	100
	92.2%	2.03	50	86.5%	1.98	50	87.7%	2.08	50
CSPT_SAE	95.7%	0.34	100	94.6%	0.89	100	93.1%	1.25	100
	92.9%	0.45	50	93.5%	0.24	50	92.7%	1.63	50

Table 6.10: Performance of SVM on musk dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	91.8%	5.07	166	89.6%	1.56	166	88.6%	2.65	166
SAE	90.7%	2.25	100	89.8%	2.78	100	88.7%	2.38	100
	90.2%	2.62	50	90.5%	2.98	50	85.6%	3.28	50
CSPT_SAE	91.7%	1.34	100	90.6%	0.89	100	89.1%	1.35	100
	91.1%	1.15	50	90.5%	1.34	50	88.7%	1.43	50

Table 6.11: Performance of LDA on the malicious dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	95.8%	2.18	80	93.8%	1.56	80	92.6%	2.45	80
SAE	97.7%	2.13	60	94.7%	1.78	60	93.8%	2.08	60
	96.5%	2.02	50	93.6%	1.08	50	92.6%	2.01	50
CSPT_SAE	98.9%	1.34	60	95.3%	0.89	60	94.4%	1.53	60
	97.2%	1.13	50	94.7%	0.34	50	93.6%	1.23	50

Table 6.12: Performance of SVM on the malicious dataset.

Methods	**Train**			**CV**			**Test**		
	Av.	**Std.**	**F**	**Av.**	**Std.**	**F**	**Av.**	**Std.**	**F**
Full features	93.8%	2.77	80	91.6%	2.56	80	91.6%	2.65	80
SAE	94.7%	2.05	60	93.8%	2.78	60	92.7%	2.38	60
	93.2%	2.02	50	93.5%	1.98	50	93.7%	1.28	50
CSPT_SAE	96.7%	0.34	60	95.7%	0.89	60	94.1%	0.35	60
	95.1%	0.15	50	93.6%	0.64	50	93.5%	0.43	50

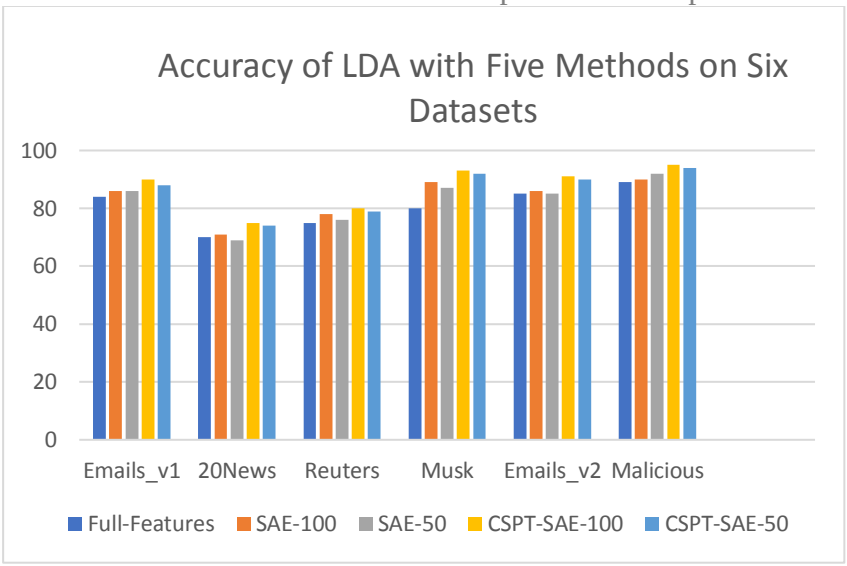

Figure 6.2: Comparison of five methods on six datasets in terms of LDA accuracy.

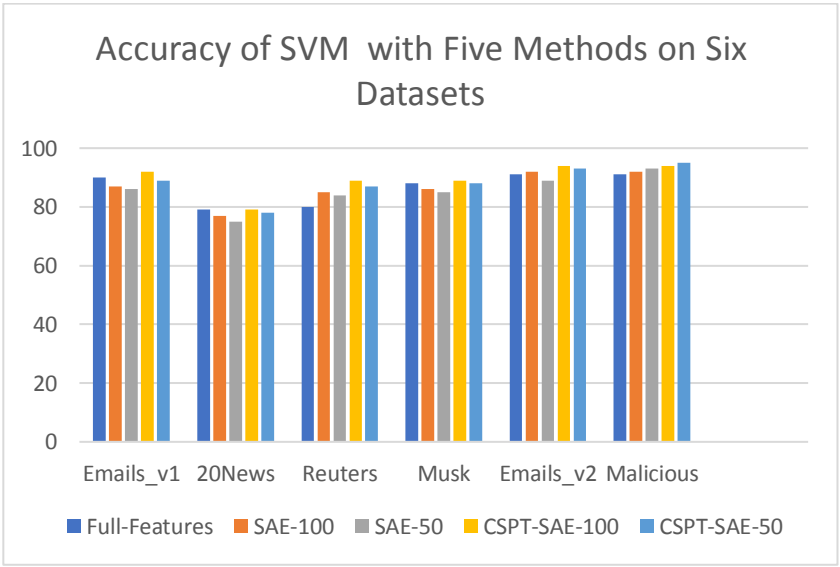

Figure 6.3: Comparison of five methods on six datasets in terms of SVM Accuracy.

Statistical Test and P-values: In order to assess whether the performance differences between the two methods are statistically significant, we applied T-test and Wilcoxon's rank-sum test to determine whether two sets of performance data are significantly different from each other. The statistical test was conducted to compare CSPT-SAE against SAE and CSPT-SAE against Full-features in terms of classification

accuracy on test dataset. Tables 6.13 and 6.14 show the P-values for these pair comparisons, which demonstrate that, in terms of the achieved best classification performance, CSPT-SAE significantly outperformed SAE and Full-features as well.

Table 6.13: Statistical test results (t-test).

Methods for comparison	p-value
CSPT-SAE vs. SAE	6.0471e-10
CSPT-SAE vs. Full-features	0.0024

Table 6.14: Statistical test results (Rank-Sum).

Methods for comparison	p-value
CSPT-SAE vs. SAE	0.0265
CSPT-SAE vs. Full-features	0.0102

6.5 Summary and Discussion

Sparse autoencoder is a deep learning method used for learning feature representation. This chapter proposes using sparse autoencoder algorithm for learning feature representation in a supervised manner, which promises to focus on the effective features of each class. This approach was used to extract higher-level features or reduce the feature dimensionality and was evaluated by employing classifiers LDA and SVM to classify emails/documents/news and numerical data as well. For applying the proposed approach, cross-validation was adopted for training the autoencoders and classifiers and determining the hyper-parameters such as p, λ, β, K (number of hidden layers and nodes). It was tested in a phishing email detection application and other document classification tasks using six datasets. This thesis presents preliminary experimental results which have demonstrated the advantages of the proposed method over the standard SAE and Full-features in terms of the classification performance

with the same number of required features. Further tests in other applications would be conducted in future investigations. Next chapter presents a three-stage learning approach for deep multilayer perceptron with effective weight initialisation based on stacked autoencoder.

Chapter 7

Deep Classifier Structures with Autoencoder for Higher-level Feature Extraction

7.1 Introduction

Deep learning for feature extraction has attracted much attention in different areas such as computer vision, speech recognition, social media analysis, fraud detection, and medical informatics (Ravì et al., 2017; Shen et al., 2017; LeCun et al., 2015; Najafabadi et al., 2015; Chen & Lin, 2014; Hinton et al., 2012; Krizhevsky et al., 2012; Hinton & Salakhutdinov, 2006). One of the main advantages of deep learning due to the use of deep neural network structures is that it can learn feature representation, without separate feature extraction process that is a very significant processing step in pattern recognition (Jiang et al., 2018; Bengio, 2013a; Bengio et al., 2013b).

Unsupervised learning is usually required for feature learning such as feature learning using the restricted Boltzmann machine (RBM) (Salakhutdinov & Hinton, 2009), sparse autoencoder (Lee, 2010, Abdulhussain & Gan, 2016), stacked autoencoder (Gehring et al. 2013, Zhou et al., 2015), denoising autoencoder (Vincent et al., 2008, Vincent et al., 2010), and contractive autoencoder (Rifai et al., 2011). For classification tasks, supervised learning is more desirable using support vector machines or feedforward neural networks as classifiers. How to effectively combine

supervised learning with unsupervised learning is a critical issue to the success of deep learning for traditional pattern classification (Glorot & Bengio, 2011).

Other major issues in deep learning include the overfitting problem and vanishing/exploding gradients during error back-propagation due to adopting deep neural network structures such as deep multilayer perceptron (DMLP) (Geman et al., 1992; Glorot and Bengio, 2010b; Ravì et al., 2017). Many techniques have been proposed to solve the problems in training deep neural networks. Hinton et al. (2006) introduced the idea of greedy layer-wise pre-training. Bengio et al. (2007) proposed to train the layers of a deep neural network in a sequence using an auxiliary objective and then "fine-tune" the entire network with standard optimization methods such as stochastic gradient descent. Martens (2010) showed that truncated-Newton method has the ability to train deep neural networks from certain random initialisation without pre-training; however, it is still inadequate to resolve the challenges in training deep neural networks. It is obvious that most deep learning models are incapable with random initialisation (Martens, 2010; Chapelle & Erhan, 2011; Mohamed et al., 2012; Glorot & Bengio, 2010b). Sutskever et al. (2013) found that both the initialization and the momentum are crucial since poorly initialized networks cannot be trained with momentum and well-initialized networks perform markedly worse when the momentum is absent or poorly tuned.

Effective weight initialisation or pre-training has been widely explored for avoiding vanishing/exploding gradients (Yam & Chow, 2000; Fernández-Redondo & Hernandez-Espinosa, 2001; Sutskever et al., 2013; Sodhi et al., 2014; DeSousa, 2016). Yam & Chow (2000) concluded that with the best initial weights determined, the

initial error is substantially smaller, and the number of iterations required to achieve the error criterion is significantly reduced.

Using a huge amount of training data can overcome overfitting to some extent (Geman et al., 1992). However, in many applications, there is no large amount of training data available or there is insufficient computer power to handle huge amount of training, and regularisation techniques such as sparse structure and dropout technique are widely used for combatting overfitting (Shu & Fyshe, 2013; Srivastava et al., 2014; Zhang et al., 2015).

This chapter investigates deep classifier structures with stacked autoencoder, aiming to overcome difficulties in training deep neural networks with limited training data in high-dimensional feature space. Experiments were conducted with the performance of the proposed method evaluated by comparison with existing methods on six datasets.

7.2 Stacked Autoencoder

An autoencoder is an unsupervised neural network trained by using stochastic gradient descent algorithms, which learns a non-linear approximation of an identity function (Shen et al., 2017; Shu & Fyshe, 2013; Zhang et al., 2015; Yang et al., 2017).

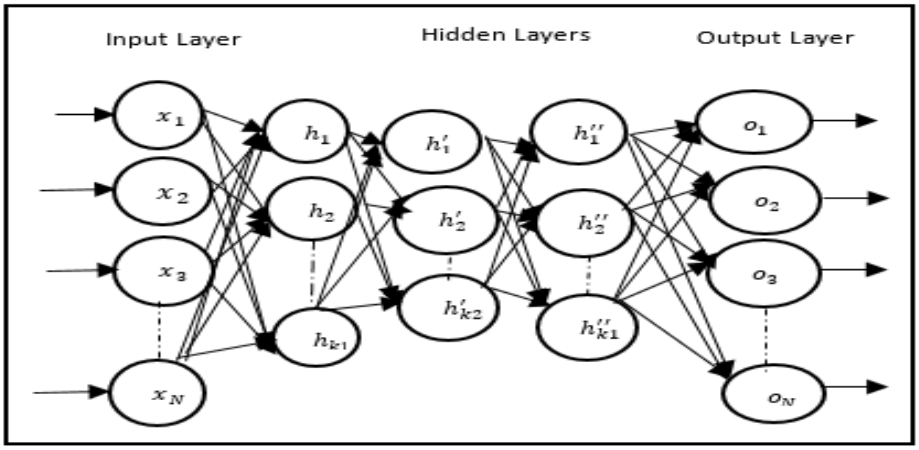

Figure 7.1: Multilayer autoencoder.

Figure 7.1 illustrates a non-linear multilayer autoencoder network, which can be implemented by stacking two autoencoders, each with one hidden layer. A stacked autoencoder may have three or more hidden layers, but for simplicity, an autoencoder with just a single hidden layer is described in detail in chapter 6.

7.3 Deep Multilayer Perceptron (DMLP)

A deep multilayer perceptron is a supervised feedforward neural network with multiple hidden layers. For simplicity, Figure 7.2 illustrates a DMLP with 2 hidden layers only (There are usually more than 2 hidden layers). A deep multilayers perceptron is described in detail in section 2.6.2.

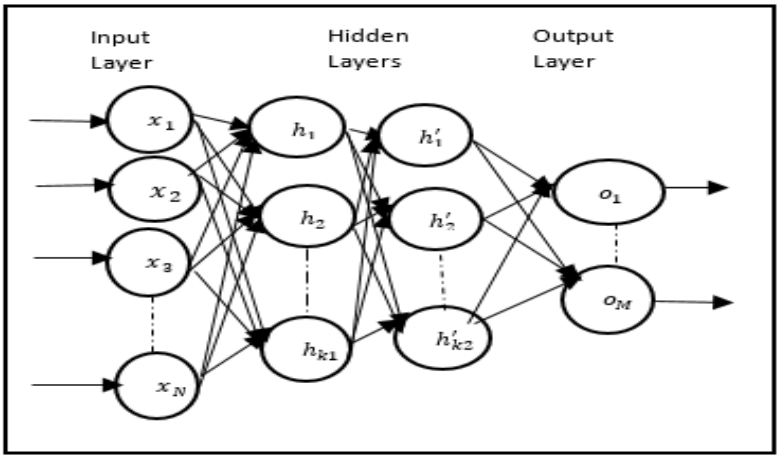

Figure 7.2: Deep multilayer perceptron (DMLP).

7.4 Proposed Approach

Training deep neural networks usually needs a huge amount of training data, especially in high-dimensional input space. Otherwise, overfitting would be a serious problem due to the high complexity of the neural network model. However, in many applications the required huge amount of training data may be unavailable or the computer power available is insufficient to handle a huge amount of training data. With deep neural network training, there may also be local minimum and vanishing/exploding gradient problems without proper weight initialisation. A three-stage learning algorithm for DMLP with effective weight initialisation based on stacked autoencoder is proposed in this chapter to combat these problems, which consists of the following three stages:

1) At the first stage, unsupervised learning is adopted to train a stacked autoencoder with random initial weights to obtain the initial weights of the feature extraction layers of the DMLP. The autoencoder consists of *N* input units, an encoder with two layers of

K1 and *K2* neurons in each hidden layer respectively, a symmetric decoder, and *N* output units. Figure 7.3 illustrates how it works.

2) At the second stage, error back-propagation is employed to pre-train the DMLP by fixing the weights obtained at the first stage for its feature extraction layers (W_1 and W_2). The weights of higher hidden layers and output layer for feature classification (W_3, W_4 and W_5) are trained with random initial weights. Figure 7.4 illustrates how it works.

3) At the third stage, all the weights of the DMLP obtained at the second stage are refined by error back-propagation, without random weight initialisation. Figure 7.5 illustrates how it works.

In our experiment, the pre-trained DMLP at the second stage, denoted as M1, and the refined DMLP obtained at the third stage, denoted as M2, are compared on several datasets. They are also compared with the DMLP trained using random initial weights for all layers, which is denoted as M3, and SVM with the output of the SAE encoder as input, which is denoted as M4.

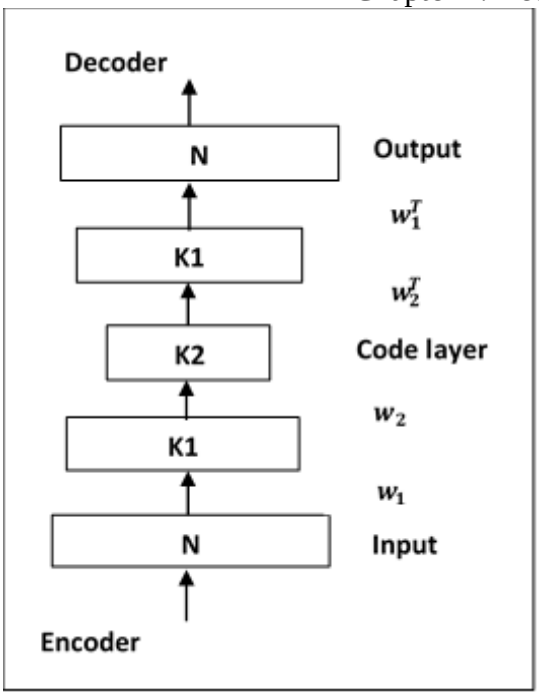

Figure 7.3: Training stacked autoencoder (SAE).

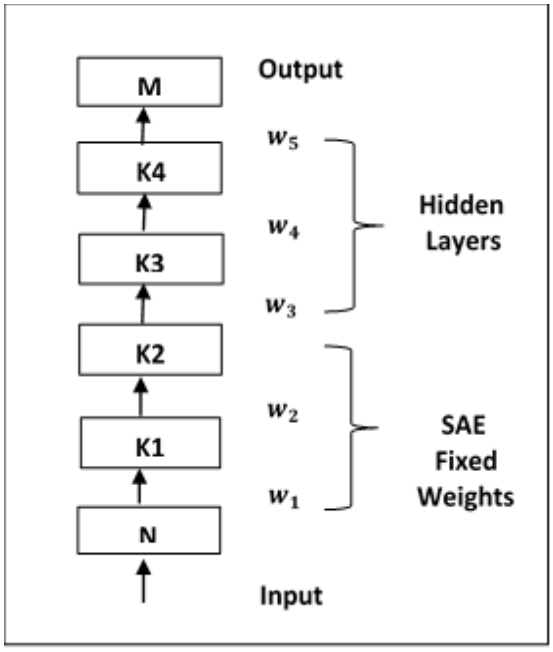

Figure 7.4: Pre-training DMLP with fixed W_1 and W_2 from SAE.

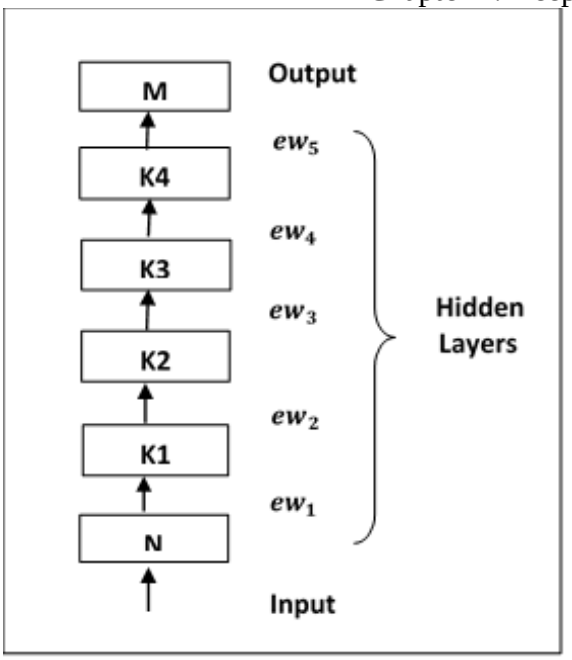

Figure 7.5: Refined-training of the DMLP with initial weights from the pre-trained DMLP.

7.5 Experimental Results and Discussion

7.5.1 Datasets

Six datasets were used in the experiments (email_v1, email_v2, Reuters-21578, 20news_v2, Musk, and technical website features). The full descriptions of these datasets are given in chapter 3.

7.5.2 Experiment Procedure

Each dataset was partitioned into a training set and a testing set. The training set was further partitioned into estimation set and validation set for cross-validation (5-folds) to determine the best or appropriate network structure and hyper-parameter values (λ, β, p). The proposed method was evaluated with different number of hidden layers and

different number of hidden neurons by cross-validation, and each testing set was only used once to evaluate the performance of the proposed method with the network structure trained using the hyper-parameter values chosen by the cross-validation. For each dataset, the experiment was repeated five times in order to assess the consistency of the results, with different data partition obtained by shuffling the data using different

random seeds for each run as described in chapter 3.

The methods for comparison were named as Mi-jHL, where i=1, 2, 3, or 4 representing one of the four methods described in Section 7.4 and j=1, 2, 3, or 4 representing the number of hidden layers. As a typical DMLP with 4 hidden layers, the numbers of hidden neurons in the first stage are K1 and K2 respectively, and the numbers of hidden neurons in the second or third stage are K1, K2, K3, and K4 respectively. For training the stacked autoencoder, 8 sets of hyper-parameters (λ, β, p) were validated, as shown in Table 7.1, which are around the suggested default values.

Table 7.1: Hyper-parameters for training the SAE.

Hyper-Parameters	HP1	HP2	HP3	HP4	HP5	HP6	HP7	HP8
L2W (λ)	0.001	0.001	0.001	0.001	0.001	0.01	0.1	0.5
Sp. Re. (β)	2	4	1.6	0.5	0	0	0	0
Sp. Pr. (p)	0.0005	0.005	0.05	0.5	1	1	1	1

7.5.3 Results

Classification Accuracy: Tables 7.2-7.7 show the cross-validation classification accuracies of the four methods with different hyper-parameter values, different number of hidden layers (M1-4HL, M1-3HL, M1-2HL, M2-4HL, M2-3HL, M2-2HL, M3-4HL

and M3-1HL, M4) and different number of hidden neurons (K1 K2 K3 K4), on the six datasets respectively. Tables 7.8-7.13 show the corresponding training and testing accuracies and standard deviation of the methods with the appropriate network structure trained using the hyper parameter values chosen by the cross-validation. Figure 7.6 compares the four methods (M1, M2, M3 and M4) in terms of average training and testing accuracy. It can be seen from Figure 7.6 that the proposed three-stage learning approach, M2-jHL, achieved the highest accuracy, which can be proved statistically significantly better than other methods evaluated in the experiment. Also, it can be seen that the proposed method (M2) has much smaller difference between testing accuracy and training accuracy than methods M1, M3, and M4, which can be regarded as evidence of less serious overfitting in the proposed method. It can be concluded that the DMLP with effective weight initialisation can achieve significantly much better performance than the standard MLP, and it is evident that the three-stage learning algorithm for DMLP can reduce overfitting.

Table 7.2: Cross validation accuracy on email_v1 dataset.

Methods\ Network Structure / Hyper-Param.		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (750 -25-20-10-5-2)	**M1-4HL**	88.9	68.9	86.4	88.9	88.7	88.9	87.1	88.4	85.7	88.9
	M1-3HL	87.2	68.4	85.1	86.1	86.1	87.5	87.2	87.1	84.4	87.5
	M1-2HL	87.1	68.1	86.8	86.0	86.0	87.1	87.1	87.0	84.5	87.1
	M2-4HL	90.2	87.9	89.3	90.7	88.5	90.2	88.9	88.5	89.2	90.7
	M2-3HL	89.2	86.2	88.3	88.4	88.4	89.2	87.2	88.1	88.3	89.2
	M2-2HL	89.1	86.3	88.9	87.3	88.1	89.1	88.1	88.1	88.1	89.1
	M3-4HL									87.9	88.0
	M3-1HL									87.1	87.8
SVM,20F	**M4**	56.5	55.9	57.2	49.8	88.9	54.7	57.2	54.8	59.3	88.9
Cross -Valid. Acc. (750-50-25-15-10-2)	**M1-4HL**	89.7	78.1	87.4	89.2	89.4	**91.6**	84.1	89.0	**87.3**	**91.6**
	M1-3HL	89.4	80.4	86.2	83.6	**91.2**	89.8	83.5	88.4	**86.6**	**91.2**
	M1-2HL	88.7	79.3	86.8	**91.5**	88.6	89.5	84.0	92.4	**87.1**	**91.5**
	M2-4HL	91.0	92.8	87.5	92.6	91.7	90.5	90.9	**90.2**	**91.1**	**92.8**
	M2-3H	**91.2**	90.3	87.3	90.1	91.6	89.1	88.2	90.2	**89.7**	**91.9**
	M2-2HL	89.8	88.3	**90.1**	90.9	88.8	88.9	89.4	90.0	**89.5**	**90.6**
	M3-4HL									**88.1**	**88.8**
	M3-1HL									**87.5**	**87.9**
SVM,25F	**M4**	53.5	77.5	**88.9**	61.3	48.7	79.9	56.5	48.6	**64.3**	**88.9**
Cross -Valid. Acc. (750-75-35-30-20-2)	**M1-4HL**	88.5	86.2	85.6	87.9	88.1	88.4	88.9	89.2	87.8	89.2
	M1-3HL	87.8	86.8	84.3	87.1	88.0	87.7	87.6	88.8	87.2	88.8
	M1-2HL	87.2	85.3	85.3	86.2	87.2	87.3	87.2	88.3	86.7	88.3
	M2-4HL	88.9	89.9	87.3	88.9	89.2	89.3	90.2	90.2	89.2	90.2
	M2-3HL	87.4	89.2	86.6	87.3	89.1	89.2	89.1	89.3	88.4	89.3
	M2-2HL	88.3	87.1	87.1	87.1	88.2	89.2	89.2	89.2	88.1	89.2
	M3-4HL									86.7	88.3
	M3-1HL									86.1	87.1
SVM,35F	**M4**	72.6	54.7	52.8	84.5	68.1	55.7	53.9	63.6	63.2	84.5

Table 7.3: Cross validation accuracy on email_v2 dataset.

Methods\ Network Structure / Hyper-Param.		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (465 -20-15-10-5-2)	**M1-4HL**	85.7	87.7	86.9	87.1	89.1	87.6	88.2	89.3	87.7	89.3
	M1-3HL	87.9	85.4	86.1	86.3	88.3	88.2	87.7	88.5	87.3	88.5
	M1-2HL	83.8	86.3	85.8	85.4	87.8	88.1	87.3	87.6	86.5	88.1
	M2-4HL	91.5	88.9	87.8	90.2	89.3	91.2	89.2	89.2	89.6	91.5
	M2-3HL	87.9	87.6	85.6	87.3	88.8	89.8	88.9	89.4	88.1	89.9
	M2-2HL	86.0	87.9	84.9	84.3	88.0	88.2	88.3	89.1	87.0	89.1
	M3-4HL									86.5	87.2.
	M3-1HL									86.1	87.0
SVM,15F	**M4**	59.9	57.6	58.2	59.5	88.7	59.7	69.2	64.4	64.6	88.7
Cross -Valid. Acc. (465-45-20-10-5-2)	**M1-4HL**	90.1	88.8	90.7	**93.5**	91.4	89.9	85.6	89.6	**89.9**	**93.5**
	M1-3HL	88.3	87.3	89.4	88.8	**92.5**	89.8	83.5	89.9	**88.6**	**92.5**
	M1-2HL	86.9	88.5	**90.8**	87.4	89.7	88.9	85.1	88.7	**88.2**	**90.8**
	M2-4HL	90.9	89.8	88.7	92.3	93.5	91.9	92.7	**94.4**	**91.7**	**94.4**
	M2-3HL	89.3	**93.9**	90.6	91.0	92.5	89.1	88.1	90.2	**90.5**	**93.9**
	M2-2HL	86.9	88.6	**91.7**	87.1	87.2	87.4	88.9	90.1	**88.4**	**91.7**
	M3-4HL									**88.9**	**89.1**
	M3-1HL									**87.8**	**87.8**
SVM,20F	**M4**	53.6	77.2	**89.9**	64.4	48.7	78.9	56.3	49.8	**64.8**	**89.9**
Cross -Valid. Acc. (465-65-30-25-15-2)	**M1-4HL**	86.8	84.8	85.6	88.2	85.2	86.4	83.9	82.2	85.3	88.2
	M1-3HL	84.9	83.2	84.5	85.4	84.0	83.7	87.6	86.8	85.0	87.6
	M1-2HL	85.6	82.1	83.2	83.3	89.2	83.3	87.2	88.3	85.2	89.2
	M2-4HL	87.8	87.7	86.8	89.0	88.2	90.3	91.2	89.2	88.7	91.2
	M2-3HL	86.4	91.7	86.2	84.3	88.1	89.5	89.1	88.6	87.9	91.7
	M2-2HL	87.8	88.9	84.6	86.1	87.2	89.5	89.0	86..2	87.5	89.5
	M3-4HL									85.9	86.1
	M3-1HL									85.3	85.8
SVM,30F	**M4**	76.1	55.1	59.3	69.1	84.9	59.1	54.6	76.1	66.7	84.9

Table 7.4: Cross validation accuracy on Reuters-21578 dataset.

Hyper-Param. / Methods\ Network Structure		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (421-35-25-20-15-10)	M1-4HL	78.2	75.5	69.4	76.9	67.2	78.5	69.3	62.5	72.1	78.5
	M1-3HL	76.5	71.3	67.3	76.3	66.1	67.5	77.2	61.9	70.5	77.2
	M1-2HL	76.2	70.1	64.6	74.5	60.3	69.9	68.5	76.9	70.1	76.9
	M2-4HL	77.1	78.5	75.7	77.9	71.5	77.8	77.2	76.5	76.5	78.5
	M2-3HL	74.7	76.2	73.7	77.2	70.9	79.3	76.4	77.4	75.7	79.3
	M2-2HL	73.8	73.6	71.8	75.8	70.1	77.4	76.3	76.2	74.3	77.4
	M3-4HL									68.7	77.1
	M3-1HL									68.9	75.8
SVM,25F	M4	61.2	62.8	62.9	64.8	64.7	68.8	69.3	76.1	66.3	76.1
Cross -Valid. Acc. (421-40-35-25-20-10)	M1-4HL	75.2	84.7	85.9	**89.9**	83.4	77.9	73.0	70.6	**80.0**	**89.9**
	M1-3HL	73.9	82.3	82.0	**83.7**	79.2	74.1	82.8	76.9	**79.3**	**83.7**
	M1-2HL	72.7	**83.2**	81.4	82.5	74.3	74.0	73.0	72.1	**76.6**	**83.2**
	M2-4HL	85.3	83.8	86.9	86.1	86.9	85.7	85.9	**90.3**	**86.3**	**90.3**
	M2-3H	82.9	82.3	81.6	84.8	83.6	**86.9**	83.5	83.9	**83.6**	**86.9**
	M2-2HL	82.1	82.1	80.0	83.9	83.1	85.2	**86.3**	83.2	**83.6**	**86.3**
	M3-4HL									**81.2**	**82.4**
	M3-1HL									**80.3**	**82.2**
SVM,35F	M4	79.4	76.7	76.1	75.1	78.2	78.6	**81.3**	79.4	**78.1**	**81.3**
Cross -Valid. Acc. (421-65-55-45-30-10)	M1-4HL	79.5	67.2	77.4	76.8	60.4	70.7	69.1	69.2	71.2	79.5
	M1-3HL	76.8	65.6	76.7	74.1	60.2	73.9	76.7	67.8	71.4	76.8
	M1-2HL	74.6	64.9	67.5	75.7	60.1	70.4	69.9	66.3	68.6	76.7
	M2-4HL	78.3	77.3	78.2	78.2	77.1	75.4	79.5	73.9	77.2	79.5
	M2-3HL	76.8	75.2	77.8	77.1	76.3	76.6	78.9	73.5	76.5	78.9
	M2-2HL	75.8	73.9	76.8	74.6	73.3	76.7	76.8	72.2	75.0	76.8
	M3-4HL									68.6	74.3
	M3-1HL									66.8	73.9
SVM,55F	M4	67.5	62.2	61.4	68.4	62.7	68.6	70.3	66.1	65.9	70.3

Table 7.5: Cross validation accuracy on musk dataset

Hyper-Param. / Methods\ Network Structure		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (166-8-6-4-3-2)	M1-4HL	82.2	77.5	79.4	88.3	77.0	86.1	96.1	77.1	83.0	96.1
	M1-3HL	77.2	74.2	76.5	77.1	75.2	78.7	78.1	77.4	76.9	78.7
	M1-2HL	79.3	74.1	75.5	75.6	75.1	78.5	78.0	77.0	76.8	79.3
	M2-4HL	96.6	83.2	77.0	93.4	65.9	66.1	82.9	81.1	80.6	96.6
	M2-3HL	95.1	76.9	76.3	82.2	77.0	89.9	79.5	78.3	81.6	95.1
	M2-2HL	88.5	69.9	78.3	80.2	67.8	78.9	77.8	78.1	77.3	88.5
	M3-4HL									72.8	77.4
	M3-1HL									72.3	77.2
SVM,6F	M4	66.8	75.2	75.6	73.8	73.6	74.9	74.7	74.6	73.6	75.6
Cross -Valid. Acc. (166-10-7-5-3-2)	M1-4HL	88.8	85.2	87.8	88.1	75.1	77.5	**93.8**	79.9	**84.5**	**93.8**
	M1-3HL	81.5	83.2	91.1	84.2	77.7	67.5	75.3	**93.9**	**81.8**	**93.9**
	M1-2HL	80.4	82.4	77.0	76.3	76.7	**93.2**	75.2	75.2	**79.5**	**93.2**
	M2-4HL	99.2	99.6	**99.8**	99.2	83.0	76.7	98.9	79.7	**92.0**	**99.8**
	M2-3H	**97.5**	88.2	87.9	92.1	67.7	94.6	76.4	76.3	**85.0**	**97.5**
	M2-2HL	83.1	87.5	**94.7**	89.8	67.2	75.2	76.2	80.1	**81.7**	**94.7**
	M3-4HL									**76.8**	**81.9**
	M3-1HL									**76.1**	**80.8**
SVM,7F	M4	**75.5**	73.8	59.5	73.1	74.6	75.2	73.7	62.9	**71.0**	**75.5**
Cross -Valid. Acc. (166-15-10-8-6-2)	M1-4HL	92.6	80.2	79.5	92.4	77.7	77.8	92.1	80.8	84.1	92.6
	M1-3HL	78.5	79.2	91.8	79.6	78.3	79.5	78.2	77.5	80.3	91.8
	M1-2HL	77.8	77.3	78.3	76.8	77.1	87.9	78.1	77.4	78.8	87.9
	M2-4HL	96.2	82.8	88.2	90.1	77.1	76.4	82.9	81.5	84.4	96.2
	M2-3HL	86.6	80.3	74.8	83.2	77.4	84.3	82.6	85.9	81.8	86.6
	M2-2HL	84.7	79.4	72.6	83.1	85.4	75.4	82.4	81.2	80.5	85.4
	M3-4HL									77.3	80.1
	M3-1HL									76.9	78.3
SVM,10F	M4	75.6	59.7	71.8	71.7	75.8	75.6	62.1	73.8	70.7	75.8

Table 7.6: Cross validation accuracy on 20news_v2 dataset.

Hyper-Param. / Methods\ Network Structure		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (2000-50-35-25-20-4)	M1-4HL	67.8	67.4	65.2	67.9	66.2	62.4	72.5	73.5	67.6	73.5
	M1-3HL	66.2	66.8	66.7	64.1	66.4	60.3	61.5	62.7	64.4	66.8
	M1-2HL	64.7	65.3	65.7	63.3	65.5	66.8	69.5	67.3	66.0	69.5
	M2-4HL	77.8	79.2	77.4	79.3	79.7	77.5	72.4	73.8	77.1	79.7
	M2-3HL	76.4	78.6	75.	75.1	78.2	70.4	70.6	72.1	74.5	78.6
	M2-2HL	73.8	78.4	73.6	74.3	77.2	70.1	70.3	72.5	73.7	78.4
	M3-4HL									68.8	73.1
	M3-1HL									66.2	70.3
SVM,35F	M4	57.9	61.8	65.7	64.1	62.2	70.1	67.9	67.5	64.6	70.1
Cross -Valid. Acc. (2000-100-75-50-25-4)	M1-4HL	85.3	85.4	82.1	83.1	84.5	**87.9**	81.4	82.5	**84.0**	**87.9**
	M1-3HL	83.8	84.8	**86.6**	79.0	84.1	80.3	81.5	82.3	**82.8**	**86.6**
	M1-2HL	83.2	84.2	84.7	83.7	**85.2**	76.9	79.5	77.7	**81.8**	**85.2**
	M2-4HL	87.6	88.9	89.0	88.1	**90.4**	78.9	82.4	83.5	**86.1**	**90.4**
	M2-3H	84.4	85.6	**89.8**	86.2	88.9	80.4	80.4	82.2	**84.7**	**89.8**
	M2-2HL	83.7	**89.1**	84.1	84.3	86.1	80.1	80.3	82.1	**84.7**	**89.1**
	M3-4HL									**78.3**	**80.9**
	M3-1HL									**74.1**	**80.3**
SVM,75F	M4	78.1	77.4	**78.8**	78.1	72.4	73.9	78.5	75.6	**76.6**	**78.8**
Cross -Valid. Acc. (2000-150-125-75-50-5)	M1-4HL	73.4	76.8	79.8	79.1	79.3	74.2	75.6	72.9	76.3	79.8
	M1-3HL	74.6	75.6	75.7	76.4	74.6	70.7	72.7	71.9	74.0	76.4
	M1-2HL	70.1	74.7	74.6	73.6	72.6	76.7	79.8	72.2	74.2	79.8
	M2-4HL	77.5	78.8	79.9	78.5	79.9	73.2	73.4	74.6	76.9	79..9
	M2-3HL	76.2	77.9	78.2	76.3	77.1	70.2	72.3	71.7	74.9	78.2
	M2-2HL	74.2	76.8	77.2	76.7	77.2	70.1	70.3	72.5	74.3	77.2
	M3-4HL									70.9	75.9
	M3-1HL									70.2	74.6
SVM,125F	M4	67.6	66.7	68.3	69.6	76.8	72.8	73.6	73.8	71.1	76.8

Table 7.7: Cross validation accuracy on technical website features dataset.

Hyper-Param. / Methods\ Network Structure		HP 1	HP 2	HP 3	HP 4	HP 5	HP 6	HP 7	HP 8	Ave Acc.	Max Acc.
Cross-Valid. Acc. (47-8-6-4-3)	M1-4HL	67.2	69.7	67.6	68.5	64.0	50.3	62.9	62.7	64.1	69.7
	M1-3HL	68.6	66.7	66.7	65.8	63.8	55.3	60.3	62.5	63.7	68.6
	M1-2HL	62.7	64.8	65.8	63.3	63.5	66.9	62.5	62.2	63.9	66.9
	M2-4HL	96.9	95.2	87.8	92.1	97.0	96.7	96.8	96.8	94.9	97.0
	M2-3HL	95.7	94.7	86.1	88.9	96.2	92.1	93.3	94.2	92.6	96.2
	M2-2HL	88.9	88.2	88.9	87.6	89.1	89.1	92.3	89.5	89.2	92.3
	M3-4HL									60.7	65.8
	M3-1HL									60.9	65.3
SVM,6F	M4	54.4	56.1	58.5	56.1	58.4	61.3	61.7	56.8	57.9	61.7
Cross -Valid. Acc. (47-15-8-7-5-2)	M1-4HL	67.0	77.8	78.5	73.3	66.0	59.9	**90.5**	62.8	**71.9**	**90.5**
	M1-3HL	64.6	70.9	72.4	72.9	65.8	**85.4**	66.9	61.8	**70.0**	**85.4**
	M1-2HL	**81.9**	65.2	74.2	63.2	65.3	58.7	66.3	61.5	**67.0**	**81.9**
	M2-4HL	99.4	**99.7**	99.3	98.6	99.1	99.4	99.6	99.0	**99.2**	**99.7**
	M2-3H	**99.3**	98.9	98.5	97.2	97.8	98.2	98.8	98.3	**98.3**	**99.3**
	M2-2HL	98.9	98.7	98.2	97.0	97.9	86.4	97.5	**99.1**	**96.7**	**99.1**
	M3-4HL									**77.2**	**82.3**
	M3-1HL									**76.5**	**80.4**
SVM,8F	M4	57.7	60.2	62.1	58.9	**77.9**	61.2	61.3	60.9	**62.5**	**77.9**
Cross -Valid. Acc. (47-30-15-10-6-2	M1-4HL	69.7	68.4	67.5	69.5	66.0	59.9	90.5	62.8	69.2	90.5
	M1-3HL	67.8	67.6	66.8	68.2	85.9	59.8	67.9	60.3	68.0	85.9
	M1-2HL	65.4	66.3	62.7	65.6	65.3	75.9	70.9	61.5	66.7	75.9
	M2-4HL	71.4	98.1	96.4	98.4	99.4	84.3	99.3	99.1	93.3	99.4
	M2-3HL	94.8	96.7	97.4	96.3	98.3	80.9	76.8	56.1	87.1	98.3
	M2-2HL	86.8	97.3	95.8	95.2	78.1	80.7	67.6	86.5	86.0	97.3
	M3-4HL									65.3	72.4
	M3-1HL									65.1	70.6
SVM,15F	M4	58.1	59.7	59.7	52.3	58.3	57.2	57.8	59.8	57.8	59.8

Table 7.8: Performance comparison on email_v1 dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	750-50-25-10-5-2 / HP 6	89.5%	1.43	83.3%	1.47
M1-3HL	750-50-25-10-2 / HP 5	90.5%	0.81	82.7%	0.59
M1-2HL	750-50-25-2 / HP 4	91.5%	1.39	80.3%	1.74
M2-4HL	750-50-25-10-5-2 / HP 2	92.9%	0.29	89.9%	0.28
M2-3HL	750-50-25-10-2 / HP 5	92.1%	0.36	88.2%	0.45
M2-2HL	750-50-25-2 / HP 4	92.9%	0.52	86.7%	0.42
M3-4HL	750-50-25-10-5-2	91.7%	1.52	81.7%	1.47
M3-1HL	750-50-2	89.6%	1.49	80.9%	1.52
M4	SVM,25 features	87.1%	1.07	77.7%	1.63

Table 7.9: Performance comparison on email_v2 dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	465-45-20-10-5-2/ HP 4	93.2%	1.46	86.2%	1.16
M1-3HL	465-45-20-10-2 / HP 5	93.1%	1.78	85.1%	0.96
M1-2HL	465-45-20-2/ HP 3	93.0%	1.95	84.3%	1.02
M2-4HL	465-45-20-10-5-2 / HP 8	96.5%	0.17	93.7%	0.09
M2-3HL	465-45-20-10-2 / HP 2	96.1%	0.44	93.6%	0.38
M2-2HL	465-45-20-2 / HP 3	94.9%	0.87	91.2%	0.84
M3-4HL	465-45-20-10-5-2	93.1%	1.25	85.1%	1.46
M3-1HL	465-45-2	91.1%	1.32	84.7%	1.34
M4	SVM, 20 features	87.5%	1.43	82.7%	1.57

Table 7.10: Performance comparison on Reuters-21578 dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	421-40-35-25-20-10/HP 4	93.9%	4.92	79.1%	4.89
M1-3HL	421-40-35-25-10/ HP 4	84.3%	2.84	75.5%	3.67
M1-2HL	421-40-35-10/ HP 2	83.9%	2.86	75.1%	3.52
M2-4HL	421-40-35-25-20-10/ HP 8	88.3%	1.71	86.2%	0.56
M2-3HL	421-40-35-25-10/ HP 6	85.3%	1.85	85.2%	0.65
M2-2HL	421-40-35-10/ HP 7	85.2%	1.78	84.1%	0.95
M3-4HL	421-40-35-25-20-10	83.9%	2.03	74.8%	2.76
M3-1HL	421-40-10	83.4%	2.34	74.4%	2.03
M4	SVM, 25 features	83.1%	2.98	71.4%	2.56

Table 7.11: Performance comparison on musk dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	166-10-7-5-3-2/ HP 5	94.8%	2.99	82.1%	2.18
M1-3HL	166-10-7-5-2/ HP 8	94.7%	3.21	81.8%	2.78
M1-2HL	166-10-7-2/ HP 6	92.2%	3.32	81.1%	2.82
M2-4HL	166-10-7-5-3-2/ HP 7	96.2%	0.24	90.2%	0.17
M2-3HL	166-10-7-5-3-2/ HP 6	96.8%	0.26	87.5%	0.21
M2-2HL	166-10-7-5-3-2/ HP 3	94.3%	0.32	86.6%	0.25
M3-4HL	166-10-7-5-3-2	97.5%	2.17	83.6%	1.25
M3-1HL	166-10-7-5-3-2	96.2%	2.28	80.6%	1.34
M4	SVM, 7 features	90.1%	3.59	77.2%	1.15

Table 7.12: Performance comparison on 20news_v2 dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	2000-100-75-50-25-4/ HP6	87.1%	2.54	81.8%	2.67
M1-3HL	2000-100-75-50-4 / HP 3	86.9%	2.76	81.3%	2.35
M1-2HL	2000-100-75-4 / HP 5	85.9%	2.88	80.3%	2.63
M2-4HL	2000-100-75-50-25-4/ HP 5	94.4%	1.42	93.2%	1.36
M2-3HL	2000-100-75-50-4 / HP 3	89.8%	1.56	89.1%	1.53
M2-2HL	2000-100-75-4 / HP 2	89.1%	1.24	86.4%	1.23
M3-4HL	2000-100-75-50-25-4	86.4%	2.05	81.6%	2.85
M3-1HL	2000-25-4	86.7%	2.56	80.3%	2.65
M4	SVM, 75	83.1%	3.21	78.4%	3.71

Table 7.13: Performance comparison on technical website features dataset.

Methods	Network Structure/ Hyper-Parameters	Training Acc.		Testing Acc.	
		Av. Acc.	Std.	Av. Acc.	Std.
M1-4HL	47-15-8-7-5-2/ HP 7	92.8%	4.30	67.6%	5.82
M1-3HL	47-15-8-7-2/ HP 6	92.3%	4.29	65.6%	5.27
M1-2HL	47-15-8-2 / HP 1	92.1%	4.32	64.9%	5.25
M2-4HL	47-15-8-7-5-2/ HP 2	99.7%	0.13	96.8%	0.41
M2-3HL	47-15-8-7-2/ HP 1	97.3%	0.16	95.9%	0.46
M2-2HL	47-15-8-2/ HP 8	97.9%	0.18	94.7%	0.47
M3-4HL	47-15-8-7-5-2	98.6%	3.42	61.5%	4.41
M3-1HL	47-15-2	97.4%	4.36	61.5%	4.54
M4	SVM, 8 features	85.9%	4.40	61.1%	4.62

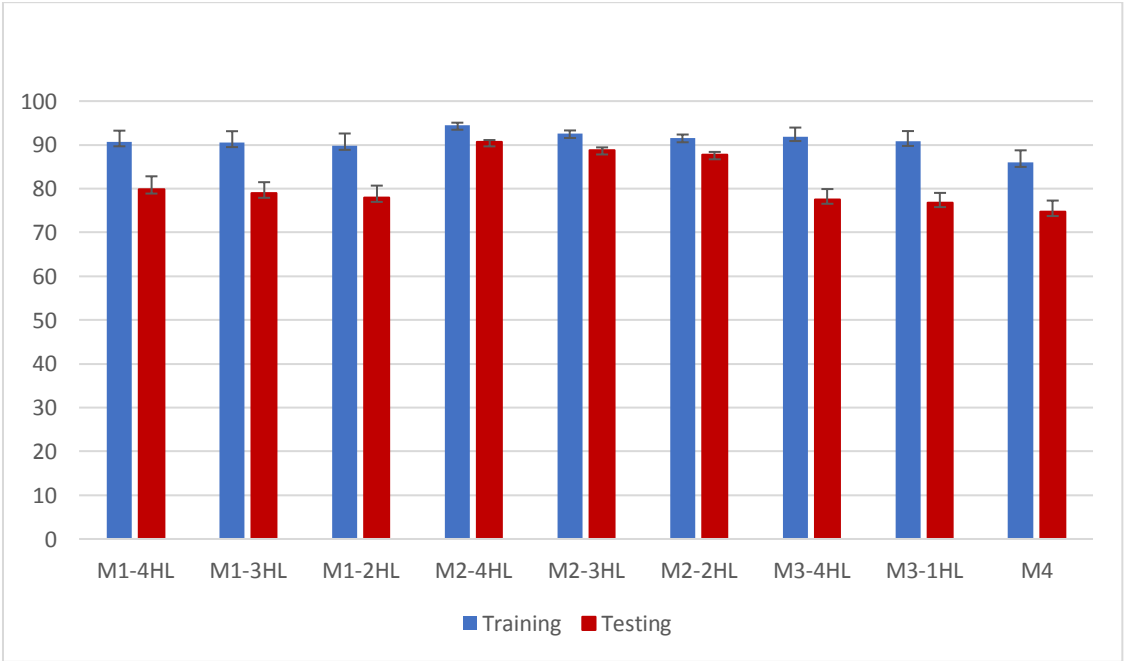

Figure 7.6: Comparison of four methods in terms of average training and testing accuracy.

Statistical Significance Test: In order to assess whether the performance differences among the methods are statistically significant, we applied T-test and the Wilcoxon's rank-sum test to determine whether two sets of accuracy data are significantly different from each other. The statistical tests were conducted on five paired methods (M2 vs M1, M2 vs M3, M2 vs M4, M1 vs M3, and M1 vs M4) in terms of testing classification accuracy. Tables 7.14 and 7.15 show the p-values from these tests, which demonstrate that, in terms of classification performance, M2 significantly outperformed M1, M3 and M4, and M1 significantly outperformed M3 and M4.

Table 7.14: Statistical test results (t-test).

Methods for comparison	p-value
M2 vs. M1	2.7431e-08
M2 vs. M3	2.8853e-06
M2 vs. M4	2.0631e-08
M1 vs. M3	0.0026
M1 vs. M4	0.0012

Table 7.15: Statistical test results (rank-sum).

Methods for comparison	p-value
M2 vs. M1	0.0037
M2 vs. M3	0.0046
M2 vs. M4	0.0025
M1 vs. M3	0.0452
M1 vs. M4	0.0132

7.6 Summary and Discussion

Representation learning is very crucial step to extract useful information when building classifiers. This chapter investigates deep classifier structures with autoencoder for higher-level feature extraction. The proposed approach was tested on six datasets and evaluated by comparison with existing methods.

The three-stage learning approach for deep multilayer perceptron with effective weight initialisation based on stacked autoencoder can combat potential overfitting and vanishing/exploding gradient problems in deep learning with limited training data.

It is evident from the experimental results that the deep multilayer perceptron trained using the proposed algorithm significantly outperformed the standard multilayer perceptron and its combination with stacked autoencoder as well. The experimental results have demonstrated the advantages of the proposed method. Further tests in other applications would be conducted in future investigations.

Chapter 8

Conclusion and Future Work

8.1 Summary of Contributions

In this chapter, we summarise how the work for this thesis has achieved the research objectives set up in Section 1.2.

First, an improved PCA method has been proposed for improving the performance of feature dimensionality deduction for text categorisation. This proposed method is based on cosine similarity and correlations between pairs of feature vectors. For initial feature extraction, four term weighting schemes were applied to the document datasets: term frequency (TF), term presence (TP), term frequency and inverse document frequency (TF-IDF), and term presence and class-specific document frequency (TP-CSDF). Experiments were conducted in order to evaluate the performance of the proposed method, which was based on correlation (via a correlation matrix) and cosine similarity, against that of a standard PCA which used covariance. Then, the proposed approach was evaluated by employing three well-known classifiers SVM, KNN and LDA to classify emails/documents/news-items using the best number of principal components or features that were found to achieve the highest performance. It is evident that the proposed PCA using the cosine similarity criterion achieved competitive accuracy with a smaller number of required features compared to the PCA based on covariance and correlation, across different datasets and different classifiers.

Second, two hybrid methods for feature subset selection have been proposed, aiming to further reduce feature dimensionality and thus improve classification accuracy and

interpretability as well. Both methods use a two-stage process for selecting a subset of relevant features. The first stage selects feature subsets based on either the union or the intersection of features selected according to distance or similarity measures (unsupervised approach) and mutual information measures (supervised approach). The second stage employs a wrapper approach on the selected features at the first stage. Experiments were conducted and the performance of the proposed methods was evaluated via comparisons with the performance of individual filter approaches and the full wrapper approach. It has been demonstrated that the proposed methods have advantages over individual filter approaches and the full wrapper approach in terms of classification accuracy, the number of required features, consumed time, and interpretability. Furthermore, the first hybrid approach H1 is better than the second approach H2 in terms of time consumed.

Third, a new scheme was proposed for employing sparse autoencoder to extract features, which uses a class-specific (supervised) pre-trained approach to learn extracting features for each class separately. The performance of this method was evaluated via comparisons with the performance of an unsupervised-training sparse autoencoder. It has been demonstrated that the proposed method is advantageous over the standard SAE and Full-features in terms of the classification performance with the same number of required features.

Finally, deep classifier structures using SAE and MLP for higher-level feature extraction have been investigated, leading to a three-stage learning algorithm that can overcome the difficulties encountered when training deep neural networks with limited training data in high-dimensional feature space. The performance of the proposed

three-stage learning algorithm for DMLP was evaluated via comparisons with the performance of support vector machines combined with SAEs and DMLP trained with random weight initialisation. The experimental results demonstrated the advantages and effectiveness of the proposed method.

To sum up, the work for this Ph.D. thesis has resulted in new methods for dimensionality reduction, feature extraction and selection, and training of deep neural networks with evident performance improvement.

8.2 Future Work

The methods and solutions presented in this thesis raise some issues which may well be worthy of further investigation. This section discusses some of these issues.

1) In this work, we employed vector space model or bag-of-words for document representation. There are two major drawbacks with the bag-of-words. First, it counts word occurrences but neglects the fact that a word may have different meanings in different documents or even in the same document. Second, in some cases related documents may not share the same keywords, which means that two related documents may not be classified as belonging to the same class. Future work should focus on employing other methods for document representation such as semantic-based or graph-based models.
2) In this work, we employed traditional classification accuracy measures for evaluation. Some machine learning methods can achieve relatively high classification accuracy, but nevertheless their false positive rates and false negative rates are too high to be acceptable, especially when they are to be used for online

applications. For example, phishing detection methods should achieve low false negative and false positive rates. More work should be conducted in order to overcome this limitation.

3) This research proposed two hybrid methods which combine both supervised and unsupervised filter approaches with a wrapper approach for best feature subset selection. It may be worthwhile to use a greater variety of filters and a number of different wrappers.
4) This research used a number of different document datasets, such as phishing emails, websites technical features and 20newsgroup datasets, with different numbers of samples and features. Also, the document datasets chosen entailed the use of both binary-class and multi-class classifications. However, the approaches proposed here might be capable of being extended to work with other kinds of dataset such as those involved with image processing.
5) This research investigated deep learning algorithms, such as sparse autoencoder and stacked autoencoder algorithms, for learning feature representation, and it also investigated a deep multilayer perceptron for classification. The goal was to obtain effective low-dimensional features which can achieve high classification accuracy and overcome the difficulties often encountered in training deep neural networks with limited training data in high-dimensional feature space: overfitting and vanishing/exploding gradients. Future work should focus on employing other deep learning approaches to address these issues (overfitting and vanishing/exploding gradients).

6) Imbalanced data problem: one major technical challenge for document classification is the problem of imbalanced data. Throughout the thesis this was always a major challenge for most datasets. I have not proposed a real solution for this problem but depended on simple solutions like down-sampling the majority class. More investigation is necessary in this area.

References

Abdulhussain, M. I. & Gan, J. Q. (2016). Class-specific pre-trained sparse autoencoders for learning effective features for document classification. *Proceedings of the 8th Computer Science and Electronic Engineering (CEEC)*, Colchester, UK, 36-41. [1], [2], [48], [90], [91]

Abraham, A., Muhuri, P. K., Muda, A. K., & Gandhi, N. (2018). Intelligent systems design and applications. *Proceedings of the 17th International Conference on Intelligent Systems Design and Applications,* Delhi, India. [43].

Abu-Nimeh, S., Nappa, D., Wang, X., & Nair, S. (2007). A comparison of machine learning techniques for phishing detection. *Proceedings of the Anti-Phishing Working Groups, 2nd annual eCrime Researchers Summit.*[45], [55].

Aggarwal, A., Rajadesingan, A., & Kumaraguru, P. (2013). PhishAri: Automatic realtime phishing detection on twitter. *arXiv preprint arXiv:1301.6899*. [18], [57].

Aggarwal, C. C., & Zhai, C. (2012). Mining text data. *Springer Science & Business Media.* [49], [76].

Aldehim, G. (2015). Heuristic ensembles of filters for accurate and reliable feature selection, University of East Anglia. [74].

Aldehim, G., & Wang, W. (2017). Determining appropriate approaches for using data in feature selection. *International Journal of Machine Learning and Cybernetics,* 8(3), 915-928. [21].

Alheeti, K., Al-Ani, M., & McDonald-Maier, K. (2018). A hierarchical detection method in external communication for self-driving vehicles based on TDMA. *PloS one, 13*(1), 1-11. [7].

Allahyari, M., Pouriyeh, S., Assefi, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., & Kochut, K. (2017). A brief survey of text mining: Classification, clustering and extraction techniques. *ArXiv preprint arXiv:*1707.02919. [12].

Al-Janabi, M., Quincey, E. D., & Andras, P. (2017). Using supervised machine learning algorithms to detect suspicious URLs in online social networks. *Proceedings of the ACM International Conference on Advances in Social Networks Analysis and Mining*, Sydney, Australia, 1104-1111. [38].

Almomani, A., Gupta, B., Atawneh, S., Meulenberg, A., & Almomani, E. (2013). A survey of phishing email filtering techniques. *IEEE Communications Surveys & Tutorials*, *15*(4), 2070-2090. [14], [56].

Almomani, A., Wan, T.-C., Manasrah, A., Altaher, A., Almomani, E., Al-Saedi, K., Ramadass, S. (2012). A survey of learning based techniques of phishing email filtering. *International Journal of Digital Content Technology & its Applications (JDCTA), 6*(18).119-129. [42].

Almusallam, N.Y., Tari, Z., Bertok, P. and Zomaya, A.Y. (2017). Dimensionality reduction for intrusion detection systems in multi-data streams: A review and proposal of unsupervised feature selection scheme. *Emergent Computation,* Springer, 467-487. [7], [21], [61].

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician, 46*(3), 175-185. [36], [65], [77].

Altidor, W., Khoshgoftaar, T. M., Van Hulse, J., & Napolitano, A. (2011). Ensemble feature ranking methods for data intensive computing applications. *Handbook of Data Intensive Computing, S*pringer, New York, 349-376. [25].

Amorim, R. C. (2016). A survey on feature weighting based K-Means algorithms. *Journal of Classification,* 33(2), 210-242. [21].

Andrew, A. M. (2000). An Introduction to support vector machines and other kernel-based learning methods by Nello Christianini and John Shawe-Taylor, Cambridge University Press, Cambridge, 103-105. [16], [28], [33], [39].

Aït-Sahalia, Y., & Xiu, D. (2017). Using principal component analysis to estimate a high dimensional factor model with high-frequency data. *Journal of Econometrics,* 201(2), 384-399. [33].

Antonakakis, M., Perdisci, R., Lee, W. & Nikolaos, V.I. (2018). Method and system for detecting DGA-based malware. *Patent and Trademark Office*, U.S. ,922-190. [58].

Arguello, J., Elsas, J. L., Callan, J., & Carbonell, J. G. (2008). Document representation and query expansion models for blog recommendation. *ICWSM*, 10-18. [11].

Arras, L., Horn, F., Montavon, G., Müller, K. R., & Samek, W. (2017). What is relevant in a text document?: An interpretable machine learning approach. *PloS one,* 12(8),1-13. [1]

Awty-Carroll, D., Clifton-Brown, J., & Robson, P. (2018). Using k-NN to analyse images of diverse germination phenotypes and detect single seed germination in Miscanthus sinensis. *Plant Methods,* 14(1), 1-7. [38].

Babaie, M., Kalra, S., Sriram, A., Mitcheltree, C., Zhu, S., Khatami, A., Tizhoosh, H. R. (2017). Classification and retrieval of digital pathology scans: A new dataset. *Proceedings of the CVMI Workshop.*[11].

Badadhe, M. N., More, M. S., & Puri, M. N. (2014). An Efficient approach to detecting phishing a web using K-means and Naïve-Bayes algorithms with results. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 3*(5), 1106-1111. [58].

Basnet, R., Mukkamala, S., & Sung, A. H. (2008). Detection of phishing attacks: A machine learning approach. *Soft Computing Applications in Industry*, Springer, 373-383. [42].

Basnet, R. B., Sung, A. H., & Liu, Q. (2012). Feature selection for improved phishing detection. *Proceedings of the Advanced Research in Applied Artificial Intelligence*, Berlin, 252-261. [17], [27], [30], [42], [49].

Bazarganigilani, M. (2011). Phishing e-Mail detection using ontology concept and Naïve Bayes algorithm. *International Journal of Research and Reviews in Computer Science, 2*(2), 249-252. [8].

Belhumeur, P. N., Hespanha, J. P., & Kriegman, D. J. (1997). Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 19(7), 711-720. [36], [65].

Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and Trends in Machine Learning, 2*(1), 1-127. [46], [50].

Bengio, Y. (2012). Deep learning of representations for unsupervised and transfer learning. *Unsupervised and Transfer Learning Challenges in Machine Learning, 7*, 19. [46].

Bengio, Y. (2013a). Deep learning of representations: Looking forward. *Proceedings of the International Conference on Statistical Language and Speech Processing*, Berlin, Heidelberg, 1-37. [46].

Bengio, Y., Courville, A., & Vincent, P. (2013b). Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(8), 1798-1828. [102].

Bengio, Y., Courville, A. C., & Bergstra, J. S. (2011). Unsupervised models of images by spike-and-slab RBMs. *Proceedings of the Proceedings of the 28th International Conference on Machine Learning* (ICML-11), 1-9. [46].

Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research, 3*, 1137-1155. [47].

Bengio, Y., Lamblin, P., Popovici, D., & Larochelle, H. (2007). Greedy layer-wise training of deep networks. *Advances in Neural Information Processing Systems, 19*, 153-160. [46], [48].

Bergholz, A., Chang, J. H., Paaß, G., Reichartz, F., & Strobel, S. (2008). Improved phishing detection using model-based features. *Proceedings of the CEAS*. CA. [7].

Bergholz, A., De Beer, J., Glahn, S., Moens, M.-F., Paaß, G., & Strobel, S. (2010). New filtering approaches for phishing email. *Journal of Computer Security, 18*(1), 7-35. [17], [28].

Bishop, C. M. (2008). A new framework for machine learning. *Computational Intelligence Research Frontiers*, Berlin, Heidelberg, 1-24. [42].

Bishop, C. M. (2006). Pattern recognition and machine learning. *Springer,128*, 1-58. [35]. [40], [41], [42].

Bolshakova, N., & Azuaje, F. (2003). Cluster validation techniques for genome expression data. *Signal Processing, 83*(4), 825-833. [29].

Bradski, G., & Kaehler, A. (2008). Learning OpenCV: Computer vision with the OpenCV library. *O'Reilly Media, Inc*. [45].

Breiman, L.(2017). Classification and regression trees. *Routledge*.

Buczak, A. L., & Guven, E. (2016). A survey of data mining and machine learning methods for cyber security intrusion detection. *IEEE Communications Surveys & Tutorials, 18*(2), 1153-1176. [7].

Cai, X., Hu, S., & Lin, X. (2012). Feature extraction using restricted Boltzmann machine for stock price prediction. *Processdings of the International Conference on the Computer Science and Automation Engineering (CSAE)*, Zhangjiajie, China, 80-83. [50], [51].

Cambria, E. (2016). Affective computing and sentiment analysis. *IEEE Intelligent Systems, 31*(2), 102-107. [7].

Cha, J.-H. J. (2007). Finding diamonds in the rubble. *Experimental Neurology, 205*(1), 1-4. [30], [62], [74].

Cha, S.-H. (2007). Comprehensive survey on distance/similarity measures between probability density functions. *City, 1*(2), 300-307.

Chandrasekaran, M., Narayanan, K., & Upadhyaya, S. (2006). Phishing email detection based on structural properties. *Processdings of the NYS Cyber Security Conference.*[1], [8] ,[24], [74].

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., & Yuille, A. L. (2014a). Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv Preprint arXiv:1412.7062*. [8], [52], [73], [102]

Chen, M., Mao, S., Zhang, Y., & Leung, V. C. (2018). Big data analysis. *Big Data*, *Springer,* 51-58. [51].

Chen, X.-W., & Lin, X. (2014b). Big data deep learning: challenges and perspectives. *IEEE Access, 2*, 514-525. [51].

Clark, M., Kim, Y., Kruschwitz, U., Song, D., Albakour, D., Dignum, S., De Roeck, A. (2012). Automatically structuring domain knowledge from text: An overview of current research. *Information Processing & Management, 48*(3), 552-568. [12].

Cobern, W. W., & Loving, C. C. (2001). Defining science in a multicultural world: Implications for science education. *Science Education, 85*(1), 50-67. [1]

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. *The Journal of Machine Learning Research, 12*, 2493-2537. [47].

Cong, Y., Chan, Y. B., Phillips, C. A., Langston, M. A., & Ragan, M. A. (2017). Robust inference of genetic exchange communities from microbial genomes using TF-IDF. *Frontiers in Microbiology,* 8, 21-32. [15].

Conneau, A., Kiela, D., Schwenk, H., Barrault, L., & Bordes, A. (2017). Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv*:1705.02364. [39].

Conti, M., Dargahi, T., & Dehghantanha, A. (2018). Cyber threat intelligence: challenges and opportunities. *Cyber Threat Intelligence,* 70, 1-6. [56].

Cordero, A., & Blain, T. (2007). Catching phish: detecting phishing attacks from rendered website images. *Processdings of the 16th Intl.* Conf. *on World Wide Web*, Berkeley, CA. [33], [57].

Dahl, G. E., Yu, D., Deng, L., & Acero, A. (2012). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing, 20*(1), 30-42. [46].

Das, S. (2001). Filters, wrappers and a boosting-based hybrid for feature selection. *Processding of the ICML,* 1, 74-81. [30].

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis, 1*(1-4), 131-156. [73], [75].

DeSousa, C. A. (2016). An overview on weight initialization methods for feedforward neural networks. *Processdings of the International Joint Conference on the Neural Networks (IJCNN)*, Vancouver, Canada, 52-59. [43].

Debole, F., & Sebastiani, F. (2004). Supervised term weighting for automated text categorization. *Processdings of the Text mining and its Applications*, Berlin, Heidelberg, 18-97. [16], [17].

Deng, L., Hinton, G., & Kingsbury, B. (2013). New types of deep neural network learning for speech recognition and related applications: An overview. *Processdings*

of the the Acoustics Conference of the Speech and Signal Processing (ICASSP), Vancouver, Canada. [46].

Deng, L., Seltzer, M. L., Yu, D., Acero, A., Mohamed, A.-r., & Hinton, G. (2010). Binary coding of speech spectrograms using a deep autoencoder. *Processdings of the Eleventh Annual Conference of the International Speech Communication Association,* Chiba, Japan. [91].

Deng, Z.-H., Luo, K.-H., & Yu, H.-L. (2014). A study of supervised term weighting scheme for sentiment analysis. *Expert Systems with Applications, 41*(7), 3506-3513. [16].

Dhote, Y., Agrawal, S., & Deen, A. J. (2015). A Survey on Feature Selection Techniques for Internet Traffic Classification. *Processdings of the International Conference at Computational Intelligence and Communication Networks (CICN)*. [73], [75].

Doak, J. (1992). An evaluation of feature selection methods and their application to computer security, University of California, Computer Science. [32].

Dobša, J. (2014). Algorithm for classification of textula documents represented by tandem analysis. *Processdings of the Proceedings of Conference on Data Mining and Data Warehouses*, 9-12. [13].

Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning, arXiv:1702.08608. [73].

Duda, R. O., Hart, P. E., & Stork, D. G. (2012). Pattern classification. John Wiley & Sons. [35].

Eberle, W., & Holder, L. (2007). Discovering structural anomalies in graph-based data. *Processdings of the Seventh International Conference on Data Mining Workshops*, ICDM Workshops. IEEE. [20].

Elhadad, M. K., Badran, K., & Salama, G. I. (2017). A novel approach for ontology-based dimensionality reduction for web text document classification. *International Journal of Software Innovation (IJSI)*, 4, 44-58. [18].

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise, *Kdd*, 96, 226-231. [29].

Fahad, A., Tari, Z., Khalil, I., Almalawi, A., & Zomaya, A. Y. (2014). An optimal and stable feature selection approach for traffic classification based on multi-criterion fusion. *Future Generation Computer Systems,* 36, 156-169. [74].

Fatemi, M., & Safayani, M. (2017). Joint sentiment/topic modeling on text data using boosted restricted Boltzmann machine. *arXiv preprint arXiv:1711.03736.* [50].

Faust, K., & Wasserman, S. (1992). Centrality and prestige: A review and synthesis. *Journal of Quantitative Anthropology, 4*(1), 23-78. [20].

Fazil, M., & Abulaish, M. (2018). A Hybrid approach for detecting automated spammers in Twitter. *IEEE Transactions on Information Forensics and Security,* 13(11), 2707-2719. [31], [74].

Fernández-Redondo, M., & Hernandez-Espinosa, C. (2001). Weight initialization methods for multilayer feedforward. *Proceedings of the European Symposium on Artificial Neural Networks (ESANN)*, Belgium, 119-124. [103].

Fette, I., Sadeh, N., & Tomasic, A. (2007). Learning to detect phishing emails. *Proceedings of the 16th International Conference on World Wide Web.* [45].

Frank, E., Chui, C., & Witten, I. H. (2000). Text categorization using compression models. *Working Paper,* New Zealand, University of Waikato, Department of Computer Science. [16].

Freeman, L. C. (1979). Centrality in social networks conceptual clarification. *Social Networks, 1*(3), 215-239. [20].

Gan, J. Q., Hasan, B. A. S., & Tsui, C. S. L. (2014). A filter-dominating hybrid sequential forward floating search method for feature subset selection in high-dimensional space. *International Journal of Machine Learning and Cybernetics, 5*(3), 413-423. [22], [29], [63], [73], [77].

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International Journal of Information Management, 35*(2), 137-144. [8], [61].

García, S., Fernández, A., Luengo, J., & Herrera, F. (2009). A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Computing, 13*(10), 959-971. [2]

Garera, S., Provos, N., Chew, M., & Rubin, A. D. (2007). A framework for detection and measurement of phishing attacks. *Proceedings of the ACM Workshop on Recurring Malcode*, ACM, Virginia, USA, 1-8. [38].

Garrett, D., Peterson, D. A., Anderson, C. W., & Thaut, M. H. (2003). Comparison of linear, nonlinear, and feature selection methods for EEG signal classification. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, 11*(2), 141-144. [34].

Gartner, 2012, What is big data, [online] Available at: https://www.gartner.com/it-glossary/big-data. [8].

Gbashi, I. K., Hashem, S. H., & Majeed, S. K. (2014). Proposed vision for Network intrusion detection system using latent semantic analysis and data mining.

Proceedings of the Computer Science and Electronic Engineering Conference (CEEC), Colchester, UK, 36-41. [34], [62].

Gehring, J., Miao, Y., Metze, F., & Waibel, A. (2013). Extracting deep bottleneck features using stacked auto-encoders. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Vancouver, Canada. 3377-3381. [90].

George S., K., & Joseph, S. (2014). Text classification by sugmenting bag of words (BOW) representation with co-occurrence feature. *IOSR Journal of Computer Engineering (IOSR-JCE)*, 16, 34-38. [12].

Geman, S., Bienenstock, E., & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation, 4*(1), 1-58. [48], [49].

Girshick, R., Kokkinos, I., Laptev, I., Malik, J., Papandreou, G., Vedaldi, A., & Yuille, A. (2017). Computer vision and image understanding. *Deep Learning for Computer Vision,* 164, 1–2. [45].

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics.* [48].

Glorot, X., Bordes, A., & Bengio, Y. (2011). Domain adaptation for large-scale sentiment classification: A deep learning approach. *Proceedings of the 28th International Conference on Machine Learning* (ICML-11). [47], [102].

Gomez, J. C., Boiy, E., & Moens, M. F. (2012). Highly discriminative statistical features for email classification. *Knowledge and Information Systems, 31*(1), 23-53. [34], [62], [74].

Goodfellow, I. J., Courville, A., & Bengio, Y. (2012). Spike-and-slab sparse coding for unsupervised feature discovery. *arXiv preprint arXiv:1201.3382*. [46].

Goodfellow, I., Bengio, Y., Courville, A., & Bengio, Y. (2016). Deep learning. *Cambridge: MIT Press,* 1. [2], [45].

Group, A.-P. W. (2017). APWG Phishing trends reports. Anti-Phishing Working Group, http://www. antiphishing. org/phishReportsArchive. [55].

Gu, Q., Li, Z., & Han, J. (2012). Generalized fisher score for feature selection. *arXiv preprint arXiv:1202.3725*. [29].

Gupta, B. B., Tewari, A., Jain, A. K., & Agrawal, D. P. (2017). Fighting against phishing attacks: state of the art and future challenges. *Neural Computing and Applications*, 28(12), 3629-3654. [56].

Gurney, K. (2014). *An introduction to neural networks.* CRC press.[43].

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 1157-1182. [22], [27], [64], [73], [74].

Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A. (2008). Feature extraction*:* foundations and applications. *Studies in Fuzziness and Soft Computing,* Springer, *207*. [33].

Guzella, T. S., & Caminhas, W. M. (2009). A review of machine learning approaches to spam filtering. *Expert Systems with Applications, 36*(7), 10206-10222. [14].

Hao, H., Liu, C.-L., & Sako, H. (2003). Comparison of genetic algorithm and sequential search methods for classifier subset selection. *Proceeding of the Seventh International Conference in Document Analysis and Recognition.* [33].

Han, J., Pei, J., & Kamber, M. (2011). Data mining: concepts and techniques. *Elsevier*. [7].

Han, X., Zhong, Y., Zhao, B., & Zhang, L. (2017). Scene classification based on a hierarchical convolutional sparse autoencoder for high spatial resolution imagery. *International Journal of Remote Sensing,* 38(2), 514-536. [90].

Harish, B., Kumar, S. A., & Manjunath, S. (2014). Classifying text documents using unconventional representation*. Proceeding of the International Conference on the Big Data and Smart Computing (BIGCOMP)*, IEEE, 210-216. [13].

Hasan, B. A. S., Gan, J. Q., & Zhang, Q. (2010). Multi-objective evolutionary methods for channel selection in brain-computer interfaces: some preliminary experimental results. *Proceeding of the IEEE Congress on Evolutionary Computation (CEC),* Barcelona, Spain, 1-6. [29], [30], [63].

Hawkins, D. M. (2004). The problem of overfitting. *Journal of Chemical Information and Computer Sciences, 44*(1), 1-12. [2].

Haykin, S. S. (2001). Neural networks: a comprehensive foundation. Prentice Hall. [43].

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Sainath, T. N. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine, 29*(6), 82-97. [45].

Hinton, G. E. (1986). Learning distributed representations of concepts. *Proceeding of the Eighth Annual Conference of the Cognitive Science Society*,1-12. [47].

Hinton, G. E., Osindero, S., & Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural Computation, 18*(7), 1527-1554. [46], [48].

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science, 313*(5786), 504-507. [49].

Hira, Z. M., & Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics,*1-13. [21].

Hofmann, T. (1999). Probabilistic latent semantic indexing. *Proceedings of the 22nd annual international ACM SIGIR Conference on Research and Development in Information Retrieval.* [18].

Hu, R., Zhu, X., Cheng, D., He, W., Yan, Y., Song, J., & Zhang, S. (2017). Graph self-representation method for unsupervised feature selection. *Neurocomputing,* 220, 130-137. [31].

Huh, J. H., & Kim, H. (2012). Phishing detection with popular search engines: Simple and effective. *International Symposium on Foundations and Practice of Security*, Springer, Berlin, 194-207. [37], [38].

Islam, R., & Abawajy, J. (2013). A multi-tier phishing detection and filtering approach. *Journal of Network and Computer Applications, 36*(1), 324-335. [59].

Jain, A. K., Duin, R. P. W., & Mao, J. (2000). Statistical pattern recognition: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 22*(1), 4-37. [28].

Jain, A. K., & Yu, B. (1998). Document representation and its application to page decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 20*(3), 294-308. [11].

Jain, A. K., & Gupta, B. B. (2018). PHISH-SAFE: URL features-based phishing detection system using machine learning. *Proceedings of the CSI* in *Cyber Security,* Singapore. 467-474. [55].

Jakobsson, M., & Myers, S. (2006). Phishing and countermeasures: understanding the increasing problem of electronic identity theft. John Wiley & Sons. [55].

Jalali, A., Jang, G., Kang, J.-S., & Lee, M. (2015). Convolutional neural networks considering robustness improvement and its application to face recognition. *Proceedings of the International Conference on Neural Information Processing,* Springer. [51].

Jameel, N. G. M., & George, L. E. (2013). Detection of phishing emails using feed forward neural network. *International journal of Computer Applications, 77*(7), 10-15. [45].

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A deep learning approach for network intrusion detection system. *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, 21-26. [55].

Jiang, J., Chen, J., Choo, K. K. R., Liu, C., Liu, K., Yu, M., & Wang, Y. (2017). A deep learning based online malicious URL and DNS detection scheme. *Proceedings of the International Conference on Security and Privacy in Communication Systems,* Cham , 438-448. [46].

Jiang, M., Liang, Y., Feng, X., Fan, X., Pei, Z., Xue, Y., & Guan, R. (2018). Text classification based on deep belief network and softmax regression. *Neural Computing and Applications,* 29(1), 61-70. [12], [47].

Juang, B.-H., & Katagiri, S. (1992). Discriminative learning for minimum error classification (pattern recognition). *IEEE Transactions on Signal Processing, 40*(12), 3043-3054. [35].

Kabir, M. M., Islam, M. M., & Murase, K. (2010). A new wrapper feature selection approach using neural network. *Neurocomputing,* 73(16-18), 3273-3283. [31].

Kalbhor, M., Shrivastava, S., & Ujjainiya, B. (2013). An artificial immune system with local feature selection classifier for spam filtering. *Procesdings of the Fourth International Conference at the Computing, Communications and Networking Technologies (ICCCNT), IEEE*, Tiruchengode, India, 1-7. [24], [76].

Khammassi, C., & Krichen, S. (2017). A GA-LR wrapper approach for feature selection in network intrusion detection. Computers & Security, 70, 255-277. [30].

Karegowda, A. G., Jayaram, M., & Manjunath, A. (2010). Feature subset selection problem using wrapper approach in supervised learning. *International Journal of Computer Applications, 1*(7), 13-17. [25], [76].

Katuwal, G. J., & Chen, R. (2016). Machine learning model interpretability for precision medicine. *ArXiv preprint arXiv:1610.09045*. [73].

Khonji, M., Iraqi, Y., & Jones, A. (2013). Phishing detection: a literature survey. *Communications Surveys & Tutorials, 15*(4), 2091-2121. [51].

Kim, K. I., Jung, K., & Kim, H. J. (2002). Face recognition using kernel principal component analysis. *IEEE Signal Processing Letters, 9*(2), 40-42. [62].

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*(1), 273-324. [30].

Kohavi, R., & Sommerfield, D. (1995). Feature subset selection using the wrapper method: overfitting and dynamic search space topology. *KDD,* 192-197. [30].

Koller, D., Sahami, M., & Jorquera, D. (2007). Toward Optimal Feature Selection. Stanford InfoLab. [25].

Kowsari, K., Brown, D. E., Heidarysafa, M., Meimandi, K. J., Gerber, M. S., & Barnes, L. E. (2017). Hdltex: Hierarchical deep learning for text classification. *ArXiv Preprint arXiv:*1709.08267. [47].

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances Neural Information Processing Systems, 1097-1105.* [102].

Kumar, V., & Minz, S. (2014). Feature selection. *SmartCR, 4*(3), 211-229. [32], [33].

Kumar, B. S., & Ravi, V. (2017). Text document classification with PCA and One-Class SVM. *Proceedings of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications*, Singapore, 107-115. [62].

Kwak, N., & Choi, C.-H. (2002). Input feature selection for classification problems. *IEEE Transactions on Neural Networks, 13*(1), 143-159. [21].

Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). Recurrent convolutional neural networks for text classification. *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence.* [47].

Lakshmi, V. S., & Vijaya, M. (2012). Efficient prediction of phishing websites using supervised learning algorithms. *Procedia Engineering, 30*, 798-805. [35].

Lan, M., Tan, C. L., Su, J., & Lu, Y. (2009). Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 31*(4), 721-735. [16].

Lang, K. J., Waibel, A. H., & Hinton, G. E. (1990). A time-delay neural network architecture for isolated word recognition. *Neural Networks, 3*(1), 23-43. [46].

Längkvist, M., Karlsson, L., & Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters, 42*, 11-24. [45].

Larsson, K., Baker, S., Silins, I., Guo, Y., Stenius, U., Korhonen, A., & Berglund, M. (2017). Text mining for improved exposure assessment. *PloS one,* 12(3), 1-21. [32].

LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., & Kruschwitz, N. (2011). Big data, analytics and the path from insights to value. *MIT Sloan Management Review, 52*(2), 21. [8], [11].

Le, H., Pham, Q., Sahoo, D., & Hoi, S. C. (2018). URLNet: Learning a URL representation with deep learning for malicious URL detection. *arXiv preprint arXiv:*1802.03162. [48].

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*(7553), 436-444. [102].

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278-2324. [46].

Lee, B., Amaresh, S., Green, C., & Engels, D. (2018). Comparative study of deep learning models for network intrusion detection. *SMU Data Science Review,* 1(1), 8.

Lee, H., Ng, A. H.-s., Koller, D., & Shenoy, K. V. (2010). Unsupervised feature learning via sparse hierarchical representations. Stanford University. [48], [90]

Lee, H., Pham, P., Largman, Y., & Ng, A. Y. (2009). Unsupervised feature learning for audio classification using convolutional deep belief networks *Proceedings of the Advances in Neural Information Processing Systems*, 1096-1104. [46], [47].

Lee, J. A., & Verleysen, M. (2007). Nonlinear dimensionality reduction. *Springer Science & Business Media.* [34], [91].

Levin, D. A., & Peres, Y. (2017). Markov chains and mixing times. *American Mathematical Soc.* [16].

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research, 5*(Apr), 361-397. [1], [38].

Li, J., Tang, J., & Liu, H. (2017). Reconstruction-based unsupervised feature selection: an embedded approach. *Proceedings of the 26th International Joint Conference on Artificial Intelligence. IJCAI/AAAI*, 2159-2165.[32].

Lin, Y. S., Jiang, J. Y., & Lee, S. J. (2014). A similarity measure for text classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 26(7), 1575-1590. [37].

Liu, H., Li, J., & Wong, L. (2002). A comparative study on feature selection and classification methods using gene expression profiles and proteomic patterns. *Genome Informatics, 13*, 51-60. [22].

Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering,* 17(4), 491-502. [35].

Liu, H., & Motoda, H. (2007). Computational methods of feature selection. *CRC Press*. [21], [22], [29].

Liu, Y. (2009). On document representation and term weights in text classification. *Handbook of Research on Text and Web Mining Technologies*, IGI Global, 1-22. [7].

Liu, Y., Wang, Y., Feng, L., & Zhu, X. (2016). Term frequency combined hybrid feature selection method for spam filtering. *Pattern Analysis and Applications, 19*(2), 369-383. [13], [90].

Liu, Y., Zhang, L., & Guan, Y. (2010). Sketch-based streaming PCA algorithm for network-wide traffic anomaly detection. *Proceedings of the IEEE 30th International*

Conference *at the Distributed Computing Systems (ICDCS)*, Genova, Italy, 807-816. [33], [62], [91].

Loo, L.-H., Roberts, S., Hrebien, L., & Kam, M. (2005). New filter-based feature selection criteria for identifying differentially expressed genes. *Proceedings of the* Fourth International Conference *on Machine Learning and Applications*, Los Angeles, CA, USA, 10-20. [28].

Ma, J., Saul, L. K., Savage, S., & Voelker, G. M. (2009). Beyond blacklists: learning to detect malicious web sites from suspicious URLs. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining,* Paris, France. [55].

Ma, L., Li, M., Gao, Y., Chen, T., Ma, X., & Qu, L. (2017). A novel wrapper approach for feature selection in object-based image classification using polygon-based cross-validation. *IEEE Geoscience and Remote Sensing Letters*, 14(3), 409-413. [30].

Markov, A., Last, M., & Kandel, A. (2008). The hybrid representation model for web document classification. *International Journal of Intelligent Systems, 23*(6), 654-679. [19].

Martens, J. (2010). Deep learning via Hessian-free optimization. *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 27, 735-742. [48].

Martens, J., & Sutskever, I. (2012). Training deep and recurrent networks with hessian-free optimization.*Neural Networks: Tricks of the Rade*, Springer, 479-535. [49].

Martínez, A. M., & Kak, A. C. (2001). Pca versus lda. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(2), 228-233. [90].

Martín-Smith, P., Ortega, J., Asensio-Cubero, J., Gan, J. Q., & Ortiz, A. (2017). A supervised filter method for multi-objective feature selection in EEG classification based on multi-resolution analysis for BCI. *Neurocomputing,* 250, 45-56. [2], [23], [77].

Meng, J., Lin, H., & Yu, Y. (2011a). A two-stage feature selection method for text categorization. *Computers & Mathematics with Applications,* 62(7), 2793-2800. [21].

Meng, L., Ding, S., & Xue, Y. (2017b). Research on denoising sparse autoencoder. *International Journal of Machine Learning and Cybernetics,* 8(5), 1719-1729. [91].

McCrohan, K., & Harvey, J. W. (2011). Security in internet commerec:emerging threats to customr trustd. *Proceedings of the 6th International Conference at the American Institute of Higher Education*, USA, 262-405. [56].

Mesnil, G., Dauphin, Y., Glorot, X., Rifai, S., Bengio, Y., Goodfellow, I. J.,Warde-Farley, D. (2012). Unsupervised and transfer learning challenge: A deep learning approach. *ICML Unsupervised and Transfer Learning, 27*, 97-110. [46].

Mihalcea, R. (2004). Graph-based ranking algorithms for sentence extraction, applied to text summarization. *Proceedings of the ACL on Interactive Poster and Demonstration Sessions.* [19].

Mihalcea, R., & Tarau, P. (2004). TextRank: Bringing order into texts. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. [19].

Mohamed, A.-r., Dahl, G. E., & Hinton, G. (2012). Acoustic modeling using deep belief networks. *IEEE Transactions on Audio, Speech, and Language Processing, 20*(1), 14-22. [49].

Mohammad, R. M., Thabtah, F., & McCluskey, L. (2014). Intelligent rule-based phishing websites classification. *Information Security, IET, 8*(3), 153-160. [13].

Molina, L. C., Belanche, L., & Nebot, À. (2002). Feature selection algorithms: A survey and experimental evaluation. *Proceedings of the IEEE International Conference at the Data Mining ICDM*, Japan, 306-313. [32].

Müller, K.-R. (2015). Machine learning and BCI. *Proceedings of the* 3rd *International Winter Conference at the Brain-Computer Interface (BCI),* South Korea. [40].

Müller, K.-R., Tangermann, M., Dornhege, G., Krauledat, M., Curio, G., & Blankertz, B. (2008). Machine learning for real-time single-trial EEG-analysis: from brain–computer interfacing to mental state monitoring. *Journal of Neuroscience Methods, 167*(1), 82-90. [40].

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. *Journal of Big Data, 2*(1), 1-31. [90], [102].

Nam, L. N. H., & Quoc, H. B. (2015). A Combined approach for filter feature selection in document classification. *Proceedings of the IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, Italy. [73].

Nemati, S., Basiri, M. E., Ghasem-Aghaee, N., & Aghdam, M. H. (2009). A novel ACO–GA hybrid algorithm for feature selection in protein function prediction. *Expert systems with applications,* 36(10), 12086-12094. [21], [33].

Ng, A. (2011). Sparse autoencoder. *CS294A Lecture Notes, 72*(2011), 1-19. [1], [50], [90], [91].

Ngiam, J., Chen, Z., Chia, D., Koh, P. W., Le, Q. V., & Ng, A. Y. (2010). Tiled convolutional neural networks. *Proceedings on the Advances in Neural Information Processing Systems.* [46].

Ngiam, J., Coates, A., Lahiri, A., Prochnow, B., Le, Q. V., & Ng, A. Y. (2011). On optimization methods for deep learning. *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. [90].

Nguyen, M., Nguyen, T., & Nguyen, T. H. (2018). A Deep learning model with hierarchical LSTMs and supervised attention for anti-phishing. *arXiv preprint arXiv:*1805.01554. [48].

Ortega, J., Asensio-Cubero, J., Gan, J. Q., & Ortiz, A. (2016). Classification of motor imagery tasks for BCI with multiresolution analysis and multiobjective feature selection. *Biomedical Engineering,* 15(1), 73. [21].

Oskoei, M. A., & Hu, H. (2006). GA-based feature subset selection for myoelectric classification. *Proceedings on the IEEE International Conference on the Robotics and Biomimetics*, *ROBIO'06,* Kunming, China. [33].

Panday, D., de Amorim, R. C., & Lane, P. (2018). Feature weighting as a tool for unsupervised feature selection. *Information Processing Letters,* 129, 44-52. [8].

Peng, H., Long, F., & Ding, C. (2005). Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 27*(8), 1226-1238. [26], [74].

Peng, T., Harris, I., & Sawa, Y. (2018). Detecting phishing attacks using natural language processing and machine learning. *Proceedings on the IEEE 12th International Conference on Semantic Computing (ICSC)*, Laguna Hills, CA, USA300-301. [36], [39], [42], [56].

Peng, X., & Choi, B. (2005). Document classifications based on word semantic hierarchies. *Artificial Intelligence and Applications*, 5, 362-367. [11], [18].

Piramuthu, S. (2004). Evaluating feature selection methods for learning in data mining applications. *European Journal of Operational Research,* 156(2), 483-494. [76].

Phukon, K. K. (2012). A composite graph model for web document and the mcs technique. *International Journal of CognitiveResearch in Science*, 1. [19].

Plansangket, S., & Gan, J. Q. (2015a). A new term weighting scheme based on class specific document frequency for document representation and classification. *Proceedings on the Computer Science and Electronic Engineering Conference (CEEC).*Colchester, UK. [15], [82].

Plansangket, S., & Gan, J. Q. (2015b). A query suggestion method combining TF-IDF and jaccard coefficient for interactive web search. *Artificial Intelligence Research,* 4(2). [13].

Plansangket, S. (2017). New weighting schemes for document ranking and ranked query suggestion, Doctoral dissertation, University of Essex. [7].

Quoc, H.B. (2015). A combined approach for filter feature selection in document classification. *IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI),* 317-324. [75].

Rifai, S., Vincent, P., Muller, X., Glorot, X., & Bengio, Y. (2011). Contractive auto-encoders: Explicit invariance during feature extraction. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*. 833-840. [48].

Raghupathi, W., & Raghupathi, V. (2014). Big data analytics in healthcare: promise and potential. *Health Information Science and Systems, 2*(1), 3. [8].

Ramanathan, V., & Wechsler, H. (2012). phishGILLNET—phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training. *EURASIP Journal on Information Security,*(1), 1-22. [13], [15].

Ranzato, M., Taylor, P. E., House, J. M., Flagan, R. C., LeCun, Y., & Perona, P. (2007). Automatic recognition of biological particles in microscopic images. *Pattern Recognition Letters,* 28(1), 31-39. [46].

Raudys, S. J., & Jain, A. K. (1991). Small sample size effects in statistical pattern recognition: Recommendations for practitioners. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 13(3), 252-264. [35].

Ravì, D., Wong, C., Deligianni, F., Berthelot, M., Andreu-Perez, J., Lo, B., & Yang, G.-Z. (2017). Deep learning for health informatics. *IEEE Journal of Biomedical and Health Informatics, 21*(1), 4-21. [103].

Rodrigues, F., Lourenco, M., Ribeiro, B., & Pereira, F. C. (2017). Learning supervised topic models for classification and regression from crowds. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 39(12), 2409-2422. [38].

Rogati, M., & Yang, Y. (2002). High-performing feature selection for text classification. *Proceedings of the Eleventh International Conference on Information and Knowledge management.* [14].

Roiger, R. J. (2017). Data mining: a tutorial-based primer. *CRC Press.* [1], [7].

Rosiello, A. P., Kirda, E., Kruegel, C., & Ferrandi, F. (2007). A layout-similarity-based approach for detecting phishing pages. *Proceedings of the Third International Conference at the Security and Privacy in Communications Networks and the Workshops*, SecureComm. [20].

Rui, Y., Huang, T. S., & Chang, S. F. (1999). Image retrieval: Current techniques, promising directions, and open issues. *Journal of Visual Communication and Image Representation, 10*(1), 39-62. [21].

Russom, P. (2011). Big data analytics. *TDWI Best Practices Report*, fourth quarter, 19, 40-45. [7], [8].

Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics, 23*(19), 2507-2517. [23], [30], [62], [74].

Sachan, D. S., Zaheer, M., & Salakhutdinov, R. (2018). Investigating the working of text classifiers, *arXiv preprint arXiv*:1801.06261. [45].

Sainath, T. N., Mohamed, A.-r., Kingsbury, B., & Ramabhadran, B. (2013). Deep convolutional neural networks for LVCSR. *Proceedings of the IEEE International Conference at the Acoustics, Speech and Signal Processing (ICASSP)*. [51].

Salakhutdinov, R., & Hinton, G. (2009). Deep boltzmann machines. *Proceedings of the Artificial Intelligence and Statistics.* [48], [90], [91].

Saxe, J., Harang, R., Wild, C., & Sanders, H. (2018). A deep learning approach to fast, format agnostic detection of malicious web content. *arXiv preprint arXiv:*1804.05020. [45], [48], [50].

Saxena, A., Gupta, A., & Mukerjee, A. (2004). Non-linear dimensionality reduction by locally linear isomaps. *Proceedings of the International Conference on Neural Information Processing*. Springer, Berlin, Heidelberg, 1038-1043. [34].

Sebban, M., & Nock, R. (2002). A hybrid filter/wrapper approach of feature selection using information theory. *Pattern Recognition,* 35(4), 835-846. [31].

Schielzeth, H. (2010). Simple means to improve the interpretability of regression coefficients. *Methods in Ecology and Evolution, 1*(2), 103-113. [2].

Schölkopf, B., Smola, A., & Müller, K.-R. (1998). Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation, 10*(5), 1299-1319. [34].

Scott, J. (2017). Social network analysis: Sage. [7].

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys (CSUR), 34*(1), 1-47. [2].

Seide, F., Li, G., & Yu, D. (2011). Conversational speech transcription using context-dependent deep neural networks. *Proceedings of the Twelfth annual conference of the international speech communication association.* [47].

Sergienko, R., Gasanova, T., Semenkin, E. and Minker, W., (2014). Text categorization methods application for natural language call routing. *Proceedings*

of the 11th International Conference Informatics in Control Automation and Robotics (ICINCO), *2, 827-831.* [16].

Sergienko, R., Shan, M. and Schmitt, A., (2017). A Comparative study of text preprocessing techniques for natural language call routing. *In Dialogues with Social Robots,* 23-37. [17].

Shen, D., Wu, G., & Suk, H. I. (2017). Deep learning in medical image analysis. *Annual Review of Biomedical Engineering,* 19, 221-248. [50].

Shu, M., & Fyshe, A. (2013). Sparse autoencoders for word decoding from magnetoencephalography. *Proceedings of the Third NIPS Workshop on Machine Learning and Interpretation in NeuroImaging (MLINI).* [103].

Siemens, G., Gasevic, D., Haythornthwaite, C., Dawson, S., Shum, S. B., Ferguson, R., Baker, R. (2011). Open learning analytics: an integrated & modularized platform. Open University Press Doctoral dissertation. [45].

Singhal, A. (2001). Modern information retrieval: A brief overview. *IEEE Data Eng. Bull., 24*(4), 35-43. [30], [63].

Smola, A. J., & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing, 14*(3), 199-222. [41], [42], [65], [77].

Sodhi, S. S., Chandra, P., & Tanwar, S. (2014). A new weight initialization method for sigmoidal feedforward artificial neural networks. *Proceedings of the International Joint Conference at the Neural Networks (IJCNN),*.

Solomatine, D., See, L. M., & Abrahart, R. (2009). Data-driven modelling: concepts, approaches and experiences. *Practical Hydroinformatics* , Springer, 17-30. [11].

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research, 15*(1), 1929-1958. [103].

Stephens, S., Rung, J., & Lopez, X. (2004). Graph data representation in oracle database 10g: case studies in life sciences. *IEEE Data Eng. Bull., 27*(4), 61-66. [13], [15].

Sun, S., Luo, C., & Chen, J. (2017). A review of natural language processing techniques for opinion mining systems. *Information Fusion,* 36, 10-25. [10].

Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). On the importance of initialization and momentum in deep learning. *Proceedings of the International Conference on Machine Learning*, USA, 1139-1147. [103].

Tang, J., Alelyani, S., & Liu, H. (2014). Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, 37. [73].

Tapia, J. E., & Pérez Flores, C. A. (2013). Gender classification based on fusion of different spatial scale features selected by mutual information from histogram of LBP, intensity, and shape. *IEEE Transactions on Information Forensics and Security,* 8(3), 488-49. [25].

Thanh, N. C., & Yamada, K. (2011). Document representation and clustering with wordnet based similarity rough set model. *International Journal of Computer Science Issues (IJCSI),* 8(5). [14].

Thaseen, I. S., & Kumar, C. A. (2017). Intrusion detection model using fusion of chi-square feature selection and multi class SVM. *Journal of King Saud University-Computer and Information Sciences,* 29(4), 462-472. [28].

Tian, X.-P., Geng, G.-G., & Li, H.-T. (2010). A framework for multi-features based web harmful information identification. *Proceedings of the International Conference at the Computer Application and System Modeling (ICCASM).* [62].

Todorovski, L., & Džeroski, S. (2006). Integrating knowledge-driven and data-driven approaches to modeling. *Ecological Modelling, 194*(1), 3-13. [11].

Touretzky, D. S., Mozer, M. C., & Hasselmo, M. E. (1996). Advances in neural information processing systems. *Proceedings of the Mit Press Conference*. [46]. [46].

Triguero, I., Peralta, D., Bacardit, J., García, S., & Herrera, F. (2015). MRPR: A MapReduce solution for prototype reduction in big data classification. *Neurocomputing, 150*, 331-345. [8].

Trstenjak, B., Mikac, S., & Donko, D. (2014). KNN with TF-IDF based framework for text categorization. *Procedia Engineering, 69*, 1356-1364. [14], [37].

Valle, K., & Ozturk, P. (2011). Graph-based representation for text classification. *Proceedings of the India-Norway Workshop on Web Concepts and Technologies.* [19].

Van Der Maaten, L., Postma, E., & Van den Herik, J. (2009). Dimensionality reduction: a comparative. *Journal of Mach Learn Res, 10*, 66-71. [34].

Vapnik, V. (2013). The nature of statistical learning theory.*Springer Science & Business Media.* [41], [65], [77].

Vincent, P., Larochelle, H., Bengio, Y., & Manzagol, P. A. (2008). Extracting and composing robust features with denoising autoencoders. *Proceedings of the 25th international conference on Machine learning*, 1096-1103. [48].

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., & Manzagol, P. A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a

local denoising criterion. *Journal of machine learning research,* 3371-3408. [48], [102].

Walker, S.J. (2014). Big data: A revolution that will transform how we live, work, and think. *International Journal of Avertising*, 33(1), 181-183. [8].

Wang, H., Khoshgoftaar, T. M., & Napolitano, A. (2012). Software measurement data reduction using ensemble techniques. *Neurocomputing, 92*, 124-132. [74].

Weston, J., Bengio, S., & Usunier, N. (2010). Large scale image annotation: learning to rank with joint word-image embeddings. *Machine Learning, 81*(1), 21-35. [47].

Wilbur, W. J., & Sirotkin, K. (1992). The automatic identification of stop words. *Journal of Information Science, 18*(1), 45-55. [14].

Xing, E. P., & Karp, R. M. (2001). CLIFF: clustering of high-dimensional microarray data via iterative feature filtering using normalized cuts. *Bioinformatics, 17* (suppl 1), S306-S315. [21].

Xiao, L., Jiang, D., Xu, D., & An, N. (2018). Secure Mobile Crowdsensing with Deep Learning. *arXiv preprint arXiv:1801.07379.* [90].

Xu, J., Tang, B., He, H., & Man, H. (2017). Semisupervised feature selection based on relevance and redundancy criteria. *IEEE Transactions on Neural Networks and Learning Systems,* 28(9), 1974-1984. [27].

Xu, K., Li, Y., Deng, R. H., & Chen, K. (2018). DeepRefiner: Multi-layer Android Malware Detection System Applying Deep Neural Networks. *IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE. [48].

Yam, J. Y., & Chow, T. W. (2000). A weight initialization method for improving training speed in feedforward neural network. *Neurocomputing, 30*(1), 219-232. [103].

Yang, Z., Hu, Z., Salakhutdinov, R., & Berg-Kirkpatrick, T. (2017). Improved variational autoencoders for text modeling using dilated convolutions. *ArXiv preprint arXiv:1702.08139*. [104].

Yang, Y., & Pedersen, J. O. (1997). A comparative study on feature selection in text categorization. *Proceedings of the ICMl*, 97, 412-420. [21], [23], [75].

Yang, Y., & Wilbur, J. (1996). Using corpus statistics to remove redundant words in text categorization. *JASIS, 47*(5), 357-369. [14].

Young, T., Hazarika, D., Poria, S., & Cambria, E. (2017). Recent trends in deep learning based natural language processing. *ArXiv preprint arXiv:*1708.02709. [47].

Yu, C., Jian, Z., Bo, Y., & Deyun, C. (2009). A novel principal component analysis neural network algorithm for fingerprint recognition in online examination system.

Proceedings of the Information Processing,. APCIP. Asia-Pacific Conference. [34], [62], [91].

Yu, L., & Liu, H. (2003). Feature selection for high-dimensional data: A fast correlation-based filter solution. *Paper Presented at the ICML*. [28], [61].

Zhai, C. (2017). Probabilistic topic models for text data retrieval and analysis. *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval,* Tokyo, Japan, 1399-1401. [18].

Zhang, F., Du, B., & Zhang, L. (2015). Saliency-guided unsupervised feature learning for scene classification. *IEEE Transactions on Geoscience and Remote Sensing, 53*(4), 2175-2184. [91], [103], [104], [50].

Zhang, H., Shang, X., Luan, H., Wang, M., & Chua, T. S. (2017a). Learning from collective intelligence: Feature learning using social images and tags. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 13(1), 1.* [45].

Zhang, N., & Yuan, Y. (2012). Phishing detection using neural network. *CS229 Lecture Notes.* [45].

Zhang, Y., Chan, W., & Jaitly, N. (2017b). Very deep convolutional networks for end-to-end speech recognition. *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 4845-4849. [45], [46].

Zhang, Y., Hong, J. I., & Cranor, L. F. (2007). Cantina: a content-based approach to detecting phishing web sites. *Proceedings of the 16th International Conference on World Wide Web,*ACM. [57].

Zhang, Y., Pezeshki, M., Brakel, P., Zhang, S., Bengio, C. L. Y., & Courville, A. (2017c). Towards end-to-end speech recognition with deep convolutional neural networks. *ArXiv preprint arXiv:1701.02720*. [52].

Zhen, Z., Wang, H., Xing, Y., & Han, L. (2016). Text feature selection approach by means of class difference. *Proceedings of the IEEE International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD)*. [73].

Zheng, Y., Li, Y., Wang, G., Chen, Y., Xu, Q., Fan, J., & Cui, X. (2018). A novel hybrid algorithm for feature selection. *Personal and Ubiquitous Computing*, 1-15. [21], [74].

Zhou, S.-M., & Gan, J. Q. (2008). Low-level interpretability and high-level interpretability: a unified view of data-driven interpretable fuzzy system modelling. *Fuzzy Sets and Systems, 159*(23), 3091-3131. [73].

Zhou, Y., Hu, X., & Zhang, B. (2015). Interlinked convolutional neural networks for face parsing. *The International Symposium on Neural Networks.*[48], [90].

Zins, C. (2007). Conceptual approaches for defining data, information, and knowledge. *Journal of the Association for Information Science and Technology, 58*(4), 479-493. [1].

Appendix A

Publications

1. Journal Papers

Abdulhussain, Maysa I. and John Q. Gan. "Hybrid approaches to feature subset selection for document classification in high-dimensional feature space." *International Journal of Pattern Recognition and Artificial Intelligence*. (Under review).

Abdulhussain, Maysa I. and John Q. Gan. "A three-Stage learning algorithm for deep multilayer perceptron with effective weight initialisation based on sparse auto-encoder." *Artificial Intelligence Research,* 8(1), 41-50, 2019.

2. Conference Papers

Abdulhussain, Maysa I. and John Q. Gan. "An experimental investigation on PCA based on cosine similarity and correlation for text feature dimensionality reduction." *Proceedings of the 7th Computer Science and Electronic Engineering Conference (CEEC)*, Colchester, UK, 2015.

Abdulhussain, Maysa I. and John Q. Gan. "Class-specific pre-trained sparse autoencoders for learning effective features for document classification." *Proceedings of the 8th Computer Science and Electronic Engineering (CEEC),* Colchester, UK, 2016.

Abdulhussain, Maysa I. and John Q. Gan. "Deep classifier structures with autoencoder for higher-level feature extraction." *Proceedings of International Joint Conference on Computational Intelligence*, Seville, Spain, 2018.