Evaluation of Dimensionality Reduction Technique - Principal Feature Analysis

In case of Text Classification Problems

Michael A. Mammo

Department of Computer and System Sciences
Degree Project 15 HE Credits
Computer and Systems Sciences
Degree Project at the Master level
Spring term 2019
Supervisor: Tony Lindgren
Reviewer: Petter Karlström

Stockholm
University

Abstract

One of the common observed phenomena in text classification problems is sparsity of the generated feature set. So far, different feature selection and extraction algorithms have been developed to reduce the feature space to a convenient size that the learner algorithm can infer. Among these, Principal Component Analysis (PCA) is one of the well-developed and used algorithm. Recently, a modified version of PCA is introduced and applied on image database to track important points in human facial motions and in retrieving similar images from an image database. The result shows that the modified algorithm, Principal Feature Analysis (PFA), produced a comparable outcome as PCA. However, it is still unknown if PFA, performs equally well in text classification problems. Using the algorithm, this study conducted an empirical study to experiment it on Reuters-R8 text corpus. The result showed that the algorithm performs well in recognizing text documents which also produced a comparable result with PCA. This suggests that the algorithm is suitable in identifying discriminative features found in text data and can be applied in the domain.

Keywords:
feature extraction algorithms, feature selection algorithms, text categorization, text classification, principal component analysis, PCA, principal feature analysis, PFA, Reuters data set, R8 split

Synopsis

Background

In data mining, text classification is a process that assign classes to text documents. One of the important steps in this text classification task is the extraction of features from documents. This is done by creating a matrix of bag-of-words which is usually very sparse where the individual entries contain zero with only a few non-zero values. This sparsity in the feature creates a curse of dimensionality where it degrades the performance of the learner algorithm. To alleviate this problem feature reduction techniques are applied to the feature set.

Problem

Principal Component Analysis (PCA) is one of the well-established feature reduction techniques used in text mining. Because of its efficacy, other variants of it has been developed such as Latent Semantic Indexing (LSI) and its probabilistic version PLSA, and Principal Feature Analysis (PFA). Though PCA itself and the other variants, i.e., LSA and PLSA, has been tested and used in text classification problems, there are no works done to evaluate if PFA works well in the domain.

Research Question

Compared to PCA how well will its variant PFA perform in text classification problems towards reducing a sparse feature set.

Method

To compare the performance of PFA over PCA in text classification problems an empirical research is performed. For this a standard benchmark text corpus (Reuters-R8) is used and an experimental strategy is applied to analyze the result. This includes passing the raw corpus into a series of pre-processing steps that produces a model at the end being feature reduction is one of its steps. The produced model is then tested against a test data where a recognition rate is observed.

Result

Testing the model produced using PFA and PCA on the corpus resulted a recognition rate of 96.71% and 96.66% respectively. This means PFA which is the variant of PCA have the same performance in reducing a sparse feature set.

Discussion

The performance of PFA as compared to PCA shows that the algorithm performed equally well in identifying discriminative features that exist in text documents. This means that research communities that work with sparse data, particularly in text data, can apply the algorithm safely to avoid the curse of dimensionality. However, to gain more insight to the algorithm, future studies need to test it with larger corpus that contain multi-class samples which the current study did not do. And because of the nature of the study, there are no ethical issues related to the study and its results.

Contents

1. Introduction

1.1. Background

A Huge amount of data is produced every tick of a second. IBM estimates that 2.5 quintillion bytes of data is produced every 24 hours (IBM 2014) most of which is fueled by Internet of things (IoT). The data being produced span from a mere numerical record to a geographical data which by themselves contain a multivariate data. Data mining techniques help to unravel hidden patterns in those data sets. According to (Han and Kamber 2006), data mining is a way of extracting valuable knowledge from massive data sets. Basically, it is the application of different algorithms in data to obtain some meaningful result.

Data mining algorithms can be applied to *structured* and *unstructured* data. In *unstructured* data like *text,* these algorithms can be used to extract useful information from collections of text documents (Feldman and Sanger 2007). Text data contains any forms of words to describe objects. The word descriptions are not usually simple keywords but rather long sentences or paragraphs (Han and Kamber 2006). Examples of text data include electronic publications, World Wide Web, email, summary reports, notes or other documents.

Some areas of interest in text mining domain include Information Extraction, Text Summarization, Question Answering, Clustering, and Text Classification (Gupta and Lehal 2009). Among them, text Classification (or *text categorization*) is a way of assigning some pre-existing labels (or classes) to text documents where a document can have one or more class (Gupta and Lehal 2009). These assigned classes can be a *hard version* where an explicit class is assigned or a *soft version* where a probability is given for a class (Aggarwal and Zhai 2012). Some common application areas of text categorization task include email spam filtering, opinion mining, document organization, and news filtering.

One of the important steps in a text categorization task (in fact in text mining as a whole) is the extraction of features from documents. Feature vectors in a text are first produced by tokenizing the text document into its individual words (*tokens*) then applying optional stemming and stop word removal algorithms on the tokens (Gupta and Lehal 2009). This process produces a *bag-of-words* which are used to represent the underlined text document along with a numerical measure for representing the frequency or appearance of a token in the collection. A collection of tokens for a document represent its entry in the word space of word-by-document matrix (Sahlgren 2005). But since not all words appear in all documents, the produced matrix will be very sparse (Feldman and Sanger 2007). It will contain a high dimensional data with low frequency counts and often zero values for individual tokens (Aggarwal and Zhai 2012). Consequently, this sparsity causes a problem for the data mining task.

The problem with such high dimensional and sparse data is, most of the features will not contain valuable information for understanding the underlying represented document (Fodor 2002). For example, in the case of random forest classifiers, the selected random set of features may contain only zero values which technically does not provide any discriminatory information for the classifier. Also, this high multivariate feature creates a *curse of dimensionality* where it degrades the performance of the underlying learner algorithm. In addition, loading such high dimensional data will take up a lot of space in primary memory.

To alleviate this problem, different techniques are proposed with the goal of transforming the high dimensional data into a compact form. The proposed methods are *mainly* categorized into two main streams: *Feature Selection* and *Feature Extraction* (Feldman and Sanger 2007*). Feature selection* techniques principally take optimal subsets of the original features without making any modification on the features (Zareapoor and K. R 2015). The approach primarily evaluates individual features with respect to their assigned classes to identify their corresponding relevance. The most common methods employed in this category include Chi-Square and Information Gain (Zareapoor and K. R 2015). On the other hand, *Feature Extraction* methods take all features at once and compresses them into few features by projecting them into a new lower dimensional space. However, unlike feature selection techniques, the information contained in the original features will be lost since this is a complete transformation of features. Some of the well-known techniques included in this category are Principal Component Analysis (PCA) and Latent Semantic Analysis (LSA).

1.2. Research Problem

Principal Component Analysis (PCA) is a well-established feature extraction technique used for reducing dimensionality in domains like image processing and text mining (Wold, Esbensen, and Geladi 1987). The algorithm works by constructing the variances the features have in relation to each other and then projecting them into a lower dimension where the variability among them is at most. According to (Engel, Hüttenberger, and Hamann 2011) the algorithm is efficient and results in a genuine undistorted view of the data. As a result, variants of the algorithm has been developed and used in text mining tasks such as Latent Semantic Indexing (LSI) and its probabilistic version PLSA (Aggarwal and Zhai 2012).

However, (Lu et al. 2007) mentions that the algorithm suffers from a drawback of including all the original features to the computations of the projected space. This leads to an increased computational time and space use. In addition, the feature set might contain unnecessary features and noisy fields that might have been represented by other attributes (Lu et al. 2007). To avoid these drawbacks of PCA, different feature set preparation methods are proposed with the aim of selecting the relevant variables in different researches (Lu et al. 2007)(Krzanowski 1986)(Krzanowski 1996)(McCabe 1982).

Particularly (Lu et al. 2007), come up with a computationally efficient method for obtaining feature sets that are useful in finding the relevant attributes. The proposed method, Principal Feature Analysis (PFA), uses PCA to identify the relevant features but instead of projecting the features into a new dimensional space, it picks them from their original dimension. This lessens the computation involved in the projection of the features as well as the reconstruction error involved while trying to recover the original feature set. The algorithm was tested against two applications, Content Based Image Retrieval and Face Tracking Problems, and demonstrated a comparable performance with PCA (Lu et al. 2007).

To this end, although PCA is used as a dimensionality reduction technique in text categorization problems, there are no works done to evaluate if its modified version, PFA, works well in the domain. Thus, an empirical study is required to know whether the algorithm will work well in text categorization tasks. By doing so, the study will benefit research communities that are dealing with sparse data, particularly those who are working with text mining.

1.3. Research Question

To address the knowledge gap described in section 1.2, this study will try to answer the following important question:

“Compared to PCA algorithm, how well will PFA perform in text categorization problems?”

2. Extended Background

2.1. Text Categorization

A massive volume of text data is currently available due to an increased size of connected devices. The data flow comes through chat applications, electronic mail, social media posts, World Wide Web and more. To effectively use this extensive text information for building a knowledge base, assigning classes (categories) is vital as organizing and searching of texts become easier for categorized documents.

Here, text categorization plays an essential role in classifying (tagging) these text data. Text categorization in simple terms is the activity of labeling natural language texts from a set of available classes (Sebastiani 2002). Some of its application areas include sorting electronic mails/documents, categorizing articles in news feeds, websites categorization for information retrieval purpose, etc.

According to (Sebastiani 2002), text categorization has been around for a while and was used to be performed manually before the end of the '80s. A Knowledge Engineer will sit around and defines a rule where the Domain Expert applies the rules to classify documents based on some predefined categories. However, starting from the early years of the next decade, this labor-intensive task was replaced and lost its popularity in favor of statistical tools. It was because an accuracy comparable to human experts was reached using these tools (Sebastiani 2002). The statistical approach makes use of machine learning tools to construct a classifier model that eventually will help to tag a given text document.

2.1.1. Statistical Text Categorization

The statistical approach to text categorization process principally uses a constructed *model* to categorize documents into different classes. The model is built automatically by a *learner algorithm* using the attributes of *training* documents. The learning process can be *supervised* where known categories of pre-classified training documents are used, or *unsupervised* where the training documents assigned into clusters. In case of a supervised learning process, the algorithm dually used to recognize un-classified *(unseen)* document using the model.

Since the learner algorithm is a statistical tool it needs a quantitative input to build the model. Thus, a text document should be represented quantitatively such that the learner algorithm can recognize, as text documents (a collection of words) are qualitative data. According to (Fuka and Hanka 2001), the most general and widely used technique to represent a text document into its quantitative form in the field of text mining is through *bag-of-words*. The representation uses a collection of basic terms, like nouns, found in the collection of documents *(corpus)* with some associated weight with them. Mathematically, let $t_1, t_2, t_3, \dots, t_n$ denote the distinct words *(terms)* occurred in the corpus of D_m documents. Then a document D_i is represented by a term vector as:

$$D_i \ = (a_{i1}, a_{i2}, a_{i3}, \dots, a_{in})^T$$

where a_{ij} is the weight of the term t_j in document D_i, and T is the transpose of the row vector.

This *vector-space-model* creates a term-by-document matrix where each column represents the documents found in the corpus, and rows that represent terms occurred in any of the documents, and the associated weight of the term for the document. Because not all terms exist in most of the documents and each document might contain new term that is not included in another document, usually the vector-space-model for text data will be very sparse with zero weights (Aggarwal and Zhai 2013).

According to (Han and Kamber 2006), there are basically three ways where the weights of terms can be calculated for a document, i.e., the weight a_{ij} for term t_j in document D_i. The first method is to assign either 0 or 1 by merely looking at the existence of a term in a document. Thus, if a term t_j is observed in document D_i then a_{ij} will be 1 otherwise 0.

The other way is to use the frequency of the term (TF) in the document, i.e. if t_j occurs k times in document D_i then a_{ij} will take the value of k. This method can also be modified to use the relative term frequency where it is normalized by dividing it to the total number of occurrences of all terms observed in the document.

The last method that is used for calculating weights is the Inverse Document Frequency (IDF) where it uses a scaling factor, that lowers the weight of a term if it is observed in several documents. The rationally for lowering weights of recurring terms is it eventually loses its discriminative power to identify a given document if it is seen in multiple places. The formula for calculating IDF is:

$$IDF(t) = log\frac{1 + |D|}{|D_t|}$$

where $|D|$ is the number of documents in the corpus, and $|D_t|$ is the number of documents containing the term *t*.

It's also common to use the combinations of the last two methods to calculate weights of terms. The function, TF-IDF, takes the product of the normalized weight of the term over the document to its IDF scale.

2.1.2. Categorization Process

Text documents go through a pre-processing stage to build the vector-space-model representation of a corpus. Once bag-of-words are constructed for the documents, feature selection algorithms are applied to reduce the vector space. The selected features are then fed into a learner algorithm to build a model that is used as a classifier (Ikonomakis, Kotsiantis, and Tampakas 2005). Figure 1 shows the overall process involved in text categorization process, and the next sections describe each steps involved in the process.

2.1.2.1. Tokenization

In the context of text mining, tokenization refers to the process of extracting lexical tokens (words, terms, phrases or any other meaningful element) from text data. According to (Fuka and Hanka 2001) tokenization is generally done at word level that is separated by white space. Thus, tokens are any contiguous string or alphanumeric strings that are separated by whitespace characters or punctuation characters like comma, semi-colon etc. But this process does not always work and becomes complicated for text documents written in *scriptio continua* style where there is no spaces and punctuation marks are used between words or sentences, like Chinese, Thai languages (Fuka and Hanka 2001).

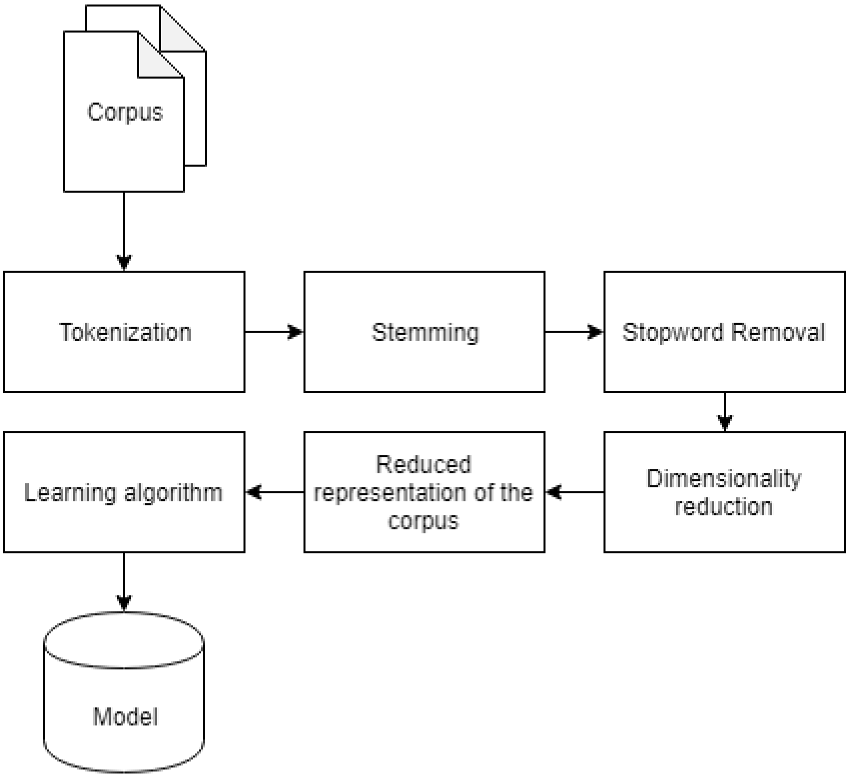

Figure 1: Text categorization process

2.1.2.2. Stemming

Stemming is one of the essential steps in text mining where a given term reduced into its stem or root form (Zareapoor and K. R 2015). By doing so, this stage also helps to diminish the size of the vector-space-model by reducing the number of occurred terms (Ikonomakis, Kotsiantis, and Tampakas 2005). For example, the terms ‘train’, ‘training’, ‘trainer’, and ‘trains’ can be represented by a single term ‘train’.

According to (Anjali 2011), stemming algorithms can be broadly classified into three main groups in the way they extract the stems of words: truncating methods, statistical methods, and mixed methods. The first group, truncating methods, try to remove affixes, i.e., prefix and suffix, that are included in the word using some simple pre-set rules. The most basic stemmer in this group S-stemmer, for example, removes the ‘s’ suffix in a word. The other, Lovins stemmer, looks up a rule table to strip an affix. Porter Stemmer, another popular member in the group, uses five consecutive steps for applying rules in the word.

The second group, statistical methods, use some analytical procedures to reach the stem word. Algorithms included in this group are, N-Gram stemmer (a language independent stemmer algorithm), HMM stemmer which is based on Hidden Markov Model, and YASS Stemmer. The last group of stemming methods use non-trivial mixed steps to reach at the steam word. Some of the algorithm in the group includes are Korvetz stemmer, corpus based stemmer, and context sensitive stemmer algorithm (Anjali 2011).

2.1.2.3. Stopword removal

Stopwords are terms contained in a document that does not provide any significant importance in discriminating the document among the corpus. It is because they frequently appear in multiple documents. Usually, those words are auxiliary verbs like be, do, can, may, might etc., conjunctions like and, or, if etc., and articles like a, the etc. (Ikonomakis, Kotsiantis, and Tampakas 2005). Also, a word that occurs extremely infrequent in a document can also be considered as stopword (Aggarwal and Zhai 2013). This stage of the text categorization process helps to reduce the dimension of the vector-space-model considerably.

2.1.2.4. Dimensionality reduction

After stemming is applied and stopwords removed from the token set of the corpus, a dimensionality reduction algorithm will be applied to reduce the dimension of the vector-space-model further. Dimensionality reduction is an essential step in text mining which transforms the data representation into a short and compact form that will eventually improve efficiency for the learner algorithm (Zareapoor and K. R 2015). However, the criteria of the dimensionality reduction technique differ based on the learner algorithm that is going to be used in the next stage of the process. In cases of unsupervised learning algorithms, dimensionality reduction should tend to minimize information losses, whereas in supervised learning setting the techniques should work to maximize class discrimination.

As pointed out in (Feldman and Sanger 2007), in the context of supervised learning, dimensionality (*feature*) reduction, techniques can be broadly classified into two: feature selection and feature extraction methods.

Feature selection is a technique where a subset of the original features selected by evaluating them with respect to the document category. The selected features are then used to *train* the underlined supervised learner algorithm (Zareapoor and K. R 2015). The most common algorithms that belong to this group include, Chi-square where it evaluates each terms chi-square score against its class and keeps the one with the highest score, and Information gain where it uses a threshold value to keep a feature by evaluating the number of information bits obtained or lost by keeping or discarding the term. These class of methods is generally easy to use and to compute.

The other class of method, feature extraction, transform all original features into a new reduced space by projecting the original elements into a smaller space (Zareapoor and K. R 2015). Two of the well known techniques in this category include Principal component analysis (PCA) where it computes the variances between the term vectors and projects it to a lower dimension which the variability among them is at most, and Latent Semantic Analysis (LSA) where tries to find patterns between terms and concepts described in the text so that the terms will be used as a feature set (Zareapoor and K. R 2015).

2.1.2.5. Learning algorithm

The learning algorithm is a function that takes in feature set from the previous stage and produces a model which consequently is used to categorize a text document. In the context of machine learning, learning algorithms are divided into two: Supervised learning and unsupervised learning.

The supervised learning algorithms take in features along with a known class label. The algorithms will then try to create a link between the features to their classes. This processing step is called *training*. Some of the well-known learning algorithms in this group include decision trees, neural networks, support vector machines etc.

In the case of unsupervised learning, the algorithms try to associate (or cluster) the documents based on some similarity observed among them. The algorithms, unlike supervised learning, do not need to know the classes of the documents. K-means, K-medoids, Partitioning Around Medoids, Hierarchical clustering are some of the algorithms that belong to this group.

2.2. Principal Component Analysis

Principal component analysis (PCA) is a popular statistical method used in machine learning to reduce the feature space of a sparse data. The method has been used extensively in the areas where high dimensional data is seen, like in the case of text data. The technique was first formulated by Karl Pearson in 1901 but further developed by H. Hotelling in the 1930's to its present stage (Wold, Esbensen, and Geladi 1987).

PCA constructs a *covariance* matrix to compute *eigen vector* and *eigen values* which consequently uses them to select the optimal feature with maximum spread.

Covariance is statistical measure that is used to calculate the difference of two dimensions with respect to their average *(mean)* value. In the context of text mining, a vector-space-model of n terms can be represented in covariance matrix as:

$$C = \begin{bmatrix} cov(t_1, t_1) & \cdots & cov(t_1, t_n) \\ \vdots & \ddots & \vdots \\ cov(t_n, t_1) & \cdots & cov(t_n, t_n) \end{bmatrix}$$

where $cov(t_i, t_j) = \frac{\sum_{q=1,}^{m}(a_{i,q}-\widehat{A_I})(a_{j,q}-\widehat{A_J})}{n-1}$, and m is the number of documents in the corpus, $a_{i,q}$ is the weight of the i^{th} term in the q^{th} document, $\widehat{A_\iota}$ is the mean of the i^{th} term.

Eigen vector v is special vector that can be derived from a square matrix which holds the equation:

$$Cv = \lambda v$$

where C is the covariance matrix, λ is a scalar eigen value.

A given square matrix of $n \times n$ at most will have n number of eigen vectors of size $1 \times n$ that hold the above equation. These vectors can be represented collectively in *feature vector* as:

$$V = [v_1 v_2 \ldots v_n]$$

where each column vector represents eigen vector of C.

The eigen values, λ, are scalar quantities that indicate the projection of the eigen vectors for the transformation matrix C.

The eigen vectors are orthonormal, i.e. orthogonal and unit, to each other and the ones with highest eigen values correspond to the dimension where the data is spread at most. PCA make use of this important property to choose the optimal features among the feature set. To do this, retained variability (RV) of the data is calculated using an ordered set of eigen values as:

$$RV = \frac{\sum_{i=1}^{q} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \times 100\%$$

where q is the number of eigen values that are to be retained, and n is the number of dimensions for the data set.

PCA then takes the transpose of original vector-space-model, which is going to be a document-by-term matrix $m \times n$, and multiplies it against the first q columns of the feature vector V matrix, which is $n \times q$. The computation produces $m \times q$ document-by-term matrix which now only contains q features.

2.3. Principal Feature Analysis

Principal feature analysis (PFA) is a variant of Principal Component Analysis (PCA) that was introduced in (Lu et al. 2007). As PCA, the technique can be used to reduce feature space of the vector-space-model. The technique utilizes ability of PCA to identify the dominant eigen vectors but instead of choosing the q vectors it tries to cluster them to find the ones that are most representative.

According to (Lu et al. 2007), features that are highly correlated will have the same absolute value weight vectors in their eigen value entry. In other words, two variables are correlated if they have the same absolute weight vectors and separated if they do not. To identify the correlation, the technique clusters the feature vector rows, since the rows can be considered as datapoints for the corresponding features. Thus, a term t_1 in the vector-space-model will be highly correlated to term t_2 if their corresponding feature vector row entries end up in the same cluster.

The steps the algorithm follow in the context of text categorization can be summarized as:

1. Compute the covariance or correlation matrix of the vector-space-model. This produces $n \times n$ matrix where n is the number of terms in the corpus.
2. Calculate the eigen values and eigen vectors of the constructed matrix. This produces an eigen vectors with size $n \times 1$.
3. Construct a feature vector where the first column represents the eigen vector of highest eigen values and so on.
4. Select q columns from the feature vector constructed in step 3 by calculating the retained variability that is desired. This step produces a matrix with size of $n \times q$, where each row can be considered as a datapoint for a feature in the original data set.
5. Taking each row of the matrix built in step 4 as a datapoint, construct p clusters using K-Means Euclidean distance algorithm. The number of clusters chosen should not be less than the retained variability computed in the previous step, since variability needs to be retained $q \leq p$. According to (Lu et al. 2007) it is usually enough if it is between 1 to 5 but not greater than the number of terms, i.e. n.
6. For each cluster produced, find the datapoint that is closest to the mean and select the term that correspond to the datapoint by tracing back to feature vector and then to the vector-space-model. This step produces p features that can be used as a diminished representation of the vector-space-model.

The described algorithm was tested in content-based image retrieval application, where a given image is searched against an image database to retrieve similar images, and in human face tracking problem where it selects important points in face motions like smiling, talking, and frowning; and resulted in a comparable performance with PCA. In addition, the algorithm also runs in the same order of complexity as PCA since the added K-Means algorithm is applied on n features with q dimension (Lu et al. 2007). Thus, it would be interesting to know how this modified version will perform in text categorization problems as PCA has been used extensively in text mining problems.

3. Method

3.1. Research Method and Strategy

The approaches for scientific studies can be broadly classified as *Design Research* and *Empirical research.* According to (Johannesson and Perjons 2012), Design research tries to develop and create a new artifact to study and use them effectively for the good of society. In the areas of Information technology (IT), design research termed as design science where it tries to create innovations in the form of models, ideas and methods. The other category, Empirical research aims at developing a new knowledge by describing, explaining and predicting entities that already exist so that everyone can have a common understanding of them regardless of their backgrounds and believed assumptions.

Empirical research needs data from the real world to base its investigation. The data to be used can be *quantitative* where the individual data types are numeric types like the number of clicks, the number of downloads, etc. and *qualitative* data like text messages, sound recordings, an image captured in devices, etc. According to (Johannesson and Perjons 2012), five basic techniques can be used to collect these data categories. The first way, Interviews, involves communication between humans to obtain the required information. Group discussion and Questionnaires are other ways of communication between peoples where a researcher set up a meeting with interest group and make use of guided written question to gather data respectively. The other method, observation, involves the researcher to be present at the context of the subject to gather the information. The final technique, document study, consists of examining existing documents and collecting data from them.

Once data is collected, a *Research Strategy* is required to analyze the gathered data. According to (Johannesson and Perjons 2012), some of the most common and well-established research strategies include but not limited to surveys, experiments, case studies, and action research. Survey is a strategy that is used usually in quantitative data with the goal of finding some generalizations among the data. An experiment is another strategy where the researcher sets up an artificial environment to observe a correlation or cause-effect relationship. A case study is yet another strategy where a single object is selected among a phenomenon and analyzed to get a deep understanding of the general matter. Action research involves introducing some stimuli to an environment and observing the change produced to deduct some knowledge.

This study is concerned with evaluating pre-existing feature extraction techniques in text categorization problems. The feature extraction technique to be assessed has been fully developed and used in other domains other than text mining. Thus, among the two groups of scientific research, this study best suits itself in the empirical research group. For data collection method a standardize benchmark corpus will be used as most researches on the field use this method. Once an appropriate corpus is selected, a controlled experimental strategy is a suitable way to analyze the results of the applied feature extraction techniques.

3.2. Data Collection Method

This study requires qualitative data to answer the research question. The collected data will be used to evaluate the feature reduction techniques. Specifically, a text dataset is needed to know the performances of the algorithms in selecting the optimal features. Two ways can be followed towards building this test dataset.

The first way is to collect several documents from different sources and assigning class labels manually. This approach helps to include specific texts that are needed to be covered in a study. However, the method is very tedious and time taking. Also, the constructed corpus might not contain enough number of individual samples for each class that leads to class imbalance problem. The second way is to use a standardize benchmark dataset that is available in public repositories. These datasets allow researchers to experiment their algorithms on a number of data items and compare their results.

According to (Debole and Sebastiani 2004), the most widely used benchmark corpus in the field of text categorization is the Reuters-21578. The Corpus has a collection of 21,578 text news articles that are classified under 135 categories which appeared in Reuters newswire in 1987 on the areas of business and economy. However, the creators of the corpus later identified that out of the total 21,578 text articles only 12,902 should be used as the remaining 8,676 text articles are not indexed manually and unreliable. Thus, researchers did a standard split, 'ModApte', in which 9,603 documents are selected for training and the remaining 3,299 text documents are used testing purpose (Debole and Sebastiani 2004).

However, the Reuters-12902 corpus contains documents that are multi-labeled where a document might belong to multiple categories. Also it includes highly skewed categories where one class contains more training samples than the other class which eventually makes it hard for the learner algorithm to infer classes correctly. As a result, several subsets of the corpus are constructed by researchers for experimentation purposes. According to (Debole and Sebastiani 2004), the most popular subsets that are used include:

- R(10) – a collection of documents that include the top 10 categories with a highest number of training example.
- R(90) – a collection of documents with the top 90 categories where each category at least contains one training example and testing example.
- R(115) – a collection of document with 115 category where each class has at least one training example.

This study uses a purposive sampling strategy since most qualitative research adopt this method (Creswell 2007). As a result, the Reuters - R(10) will be used for conducting the experiment. The main reasons for choosing this dataset include:

- Other studies have been carried out using the dataset which helps to cross-check the output of this study.
- The corpus spans texts that came from different authors with enough training examples.
- It contains enough text samples per category avoiding class imbalance problem.
- Availability of the corpus for other researchers to work on similar studies.

3.3. Experimental Design

To compare the feature reductions techniques described in this study, text data (corpus) must be processed in such a way that the learner algorithm transcribes and produces a model that can be tested. In text categorization problem, creating a model out of a corpus involves tokenization, stemming, stopword removal, application of dimensionality reduction, and training steps. The following sections describe what will be done in this study for each of these steps.

Step 1: Tokenization

Tokenization is the first step in text processing where words (or terms) are extracted from text data. For the tokenization process, there are different tools that are available for use. According to (Vijayarani and Janani 2016), some of the most common tools include:

- Nlpdotnet Tokenizer is an open source tokenization tool implemented in Python library which is based on neural networks. The tools can also be used for other operation like part-of-speech tagging.
- MILA Tokenizer is a Hebrew language-based tokenizer produced by Israel Ministry of Science and Technology that can also be used for other languages such as English.
- NLTK Word Tokenizer is also another popular open source tokenization tool in Python that can be used for stemming purpose as well.
- MBSP Word Tokenizer is a text analysis system that provides tools for common text processes like tokenization, part-of-speech tagging, sentence splitting etc.

A comparative study was done among these, and other tokenization tools in (Vijayarani and Janani 2016) suggests that the Nlpdotnet Tokenizer scores better performance than others, and MILA and NLTK comes in second and third places. The performance measurement was based languages used, the maximum number of characters taken for tokenization and how normal will the tokens become after tokenization.

For this stage, this study opted to adopt the third-ranked NLTK Word Tokenizer since the study is not affected by the performance measurements that were used to compare the tokenizers. The main rationally for choosing this tokenizer is its ease of use and accessibility. Besides, the tool can be used for the subsequent stemming operation.

Step 2: Stemming

Once the texts are reduced to the form of terms, a stemming algorithm is applied. This stage helps to identify the unique occurrence of words thereby reducing the number of different terms for a text document. According to (Anjali 2011), one of the most popular stemming algorithm for the English language is Porters stemmer. The algorithm was first proposed in 1980 but made many modification and enhancements since then. The algorithm works in five steps where each step contains a set of rules that can be applied to a word. If a rule is accepted for a word, the next action will be initiated which includes a set of rules for the step. At the end of the fifth step, the resultant stem is returned.

This study opts to use Porters stemmer because of its popularity in English texts, ease of use, and availability. In addition, most text mining researchers use it. The algorithm can be found in NLTK python package along with the set of rules needed for parsing a term.

Step 3: Stopword Removal

The root words produced in the previous step contains auxiliary verbs, conjunctions, and articles words that do not contribute to the classification task. Thus, these terms need to be removed from the collection of words. According to (Aggarwal and Zhai 2013), for the English language, stop words are usually count between 300 to 400, and their lists can be found in several online resources. For instance, the list of words built by Cornell University SMART information retrieval system[1] lists an

[1] http://www.lextek.com/manuals/onix/stopwords2.html

exhaustive list of 571 words with included punctuation marks, for example, you've.

Step 4: Constructing Vector-Space-Model

The previous three stages will be executed for each text document found in the corpus. This produces a bag-of-words with multiple occurrences. The occurrence in a document comes from either the word is appearing multiple times in the document or stemming algorithm delivered similar root words. The bag-of-words are then grouped distinctly, and their occurrences will be counted which consequently used as a weight for the term.

For this study, TF-IDF will be used for calculating the weight of the terms instead of merely taking the frequency of the terms. This is because, the scheme naturally smoothes out very common words that are missed by the Stopword removal step (Aggarwal and Zhai 2013).

After each document in the corpus are represented by the terms it contains and their corresponding TF-IDF weight, they will be passed through a program to create the vector-space-model of term-by-document matrix. For this purpose, a Python code will be developed to execute the following pseudocode.

```
for each Document in Corpus
            unique_terms.addAll (Document.terms)
for each unique_term in unique_terms
      for each Document in Documents
            if unique_term exist in Document
                  put the weight for the term
            else
                  put 0
```

In order to store the values of the execution, a Comma Separated File – CSV will be used that basically represents the vector-space-model.

Step 5: Application of Feature Reduction Algorithms

Feature reduction algorithms are used to minimize the existing features space into few thereby improving the efficiency of the consequent learner algorithm. It should be noted that the algorithm used in the context of supervised learning need to maximize class discrimination.

For this study, two feature reduction techniques, PCA and PFA, are selected to be tested. Thus, the CSV file containing the vector-space-model will be further processed by PCA and PFA algorithms. For this purpose, RStudio will be used.

RStudio is a free open source tool that contains packages for performing statistical analysis. The tool includes a PCA library as well as K-means clustering algorithm which will be needed for PFA. Applying the algorithms in vector-space-model will produce two compact vector-space-models that can be used for training purposes.

Step 6: Training

Training is the step where a model is constructed for the vector-space-model produced in the previous step. To achieve this, a supervised learning algorithm is applied, in which portion of the data *(training set)* is used for producing a model, and the other part *(testing set)* is used for testing the generated model.

The common techniques employed to partition the samples into training and testing sets include holdout, random sub-sampling, cross-validation, and bootstrap (Han and Kamber 2006). Holdout is a method where 2/3 of the training data is used for training and 1/3 for testing purpose. Random sub-sampling is holding out some random portion of data for testing while using the remaining data for training. Cross-validation is partitioning the data set into k equal groups and using $k - 1$ group for training while keeping the last group for testing. The process is performed k times where each group is used for testing. The last method, bootstrap, use a portion of the dataset for training and the other portion for testing for some number of repetitions by replacing the data items. In .632 bootstrap, for example, 63.2% of the data sets will be used for training while the rest 36.8% will be used for testing for K number of repetitions, where each repetition involves sampling the data set with replacement.

For this study, the holdout technique will be used as text data needs heavy computations and will not be feasible to do repeated executions under the other sampling techniques. As well, a supervised learning algorithm, Support Vector Machine-SVM, will be applied to construct the models for both feature spaces. Here it must be noted that the same data sets must be used for producing models under PCA and PFA feature vectors.

3.4. Data Analysis

3.4.1. Method

Text categorization is a two-step process. The first step is to create a model, and the next step is to test the model. The classification algorithm builds the model by first analyzing the feature vectors in the vector-space-model to their associated class labels. Once this operation is completed the next step testing the model with a test data to measure the *recognition rate* will be performed. Recognition rate is a term used in machine learning to tell how well a classifier (or model) is able to correctly recognize unseen data (Han and Kamber 2006). The features vectors obtained from the training document will be used for building a model, and the features vectors delivered from the test documents will be used for testing. By taking these feature vectors and evaluating the resulting model for recognition rate allows to indirectly assess how well the underlined feature reduction technique produce features that are representative enough for the representing document.

3.4.2. Performance Measures

The conventional method used to report accuracy in the field of text categorization is reporting through prediction performance. The prediction performance measures the proportion of correctly classified documents out of the total documents. This helps to compare accuracies of different algorithms quantitatively. The other common techniques used in areas of data mining include Precision and Recall, and Receiver Operating Characteristics (ROC).

Precision and Recall are two measurement values where precision calculates the ratio between the number of correctly classified documents to the number of correctly and incorrectly classified documents, and recall calculates the ratio between the number of correctly classified documents to the number of correctly and incorrectly classified documents.

The other measure, ROC, plots a curve under specificity vs. sensitivity axis where specificity is calculated by taking the ratio of False-Positive to False-Positive and True-Negative, and sensitivity is measured by taking the ratio of True-Positive to True-Positive and False-Negative. The True-Positive are documents that are classified correctly, the False-Negatives are the ones that are wrongly ignored, the False-Positives are the documents that are wrongly categorized, and the True-Negatives are the one that are correctly ignored.

3.5. Research Ethics

The study uses publicly available data that does not involve any apparent risks as it works on text files that do not reveal any private information. Similarly, as the data is public, it does not contain any confidential data. Thus, it does not affect any vulnerable population and children. As well, there are no issues related to the anonymity of the collected data.

4. Result

4.1. Experimental Setup

Empirical research with controlled experimental strategy is followed to compare the performances of the feature extraction techniques that are described in this study. The following sections explain what is done for experimenting as they are applied in text categorization problem.

4.1.1. Data Collection

This study uses purposive sampling strategy as most empirical researches adopt this method (Creswell 2007). As a result, this research opts to use R10 of Reuters-12902 ModApté split as described in section 3.2. The dataset contains ten classes with the highest number of training examples. But since the dataset contains multiple classes for some training examples, researches did another standard split, R8, where the instances have single classes. Table 1, shows the content or R8 split.

Class	**Description**	**Train docs**	**Test docs**	**Total docs**
acq	Acquisition related texts	1596	696	2292
crude	Texts about crude oil	253	121	374
earn	Earnings about companies	2840	1083	3923
grain	Grain seed related news	41	10	51
interest	Interest rate related texts	190	81	271
money-fx	Texts about foreign exchange	206	87	293
ship	Logistics related texts	108	36	144
trade	Texts about trade	251	75	326
	TOTAL	**5485**	**2189**	**7674**

Table 1: content of R8 split

Since this study is concerned with evaluating the classification of single-labeled classes, it opts for the R8 split. Thus, the split is downloaded from an online resource.[2]

4.1.2. Data Processing

The downloaded corpus comes with two text files for training and testing. The text files contain a list of documents where the beginning of each line represent the class and the rest of the line contains the content of the document., The pre-processing techniques described in section 3.3 are applied to convert these files into a vector-space-model (a term-by-document matrix).

For the first step, tokenization, a python function `word_tokenize` found in `nltk` package is used. This step produces a comma-separated form of the files, i.e., csv files, where the first term represents the class, and the rest of the line represents words occurred in the document. After that, stemming operation is applied to the csv file. For this operation, Porter stemmer found in the same package is used.

Next, the Stopword removal algorithm executed against the csv files. The algorithm goes through each term, except the class labels, to remove any word that is listed in Cornell University SMART stop word collection.

Finally, a python program is developed to calculate the TF-IDF of terms for a document according to the pseudocode shown in section 3.3- Step 4. This stage produces term-by-document matrix, i.e., the

[2] https://www.cs.umb.edu/~smimarog/textmining/datasets/

vector-space-model, where the first row contains the classes and the rest of the rows contain the unique terms existed in the corpus along with the TF-IDF value of the terms for the respective document. The program is executed against a single file constructed by concatenating the test document at the end of the training document and produced 23109 X 7674 matrix.

4.2. Experimentation

As discussed in the introduction chapter of this study, the vector-space-model constructed for text data is very sparse and usually contains zero values, as all terms do not appear in all documents. Thus, feature reduction techniques are applied, so that learner algorithms build a viable model using them. Among the many feature extraction techniques, this study compares two feature extraction algorithms namely, PCA and PFA. As a result, an R script is developed to implement these algorithms[3].

For preparing PCA feature, the vector-space-model (23109 X 7674 matrix) is taken and run against the PCA algorithm found in the R package to produce the eigen vector. The operation produced a 23109 X 23109 squared matrix where out of the 23109 principal components, the first 7514 principal components are enough to describe the data with a retained variability (RV) of 100%. Thus, the first 7514 columns of the eigen vector are taken and performed dot product against the transpose of the vector-space-model. This produced a 7674 X 7514 document-by-term matrix.

For PFA, the first 7514 columns are taken from the original eigen vector and produced a 23109 X 7519 matrix. Then a k-means clustering algorithm run against the matrix taking each row as a data point to construct 7519 clusters (adding 5 more columns in addition to 7514 as described in section 2.3) with maximum iteration of 10 million. The nearest points to the cluster centroid are then collected and traced back to the original transposed model where they are used to pick features from the original vector-space-model matrix. The operation produced 7674 X 7519 document-by-term matrix.

Finally, the features produced for both algorithms are saved into a csv file for subsequent operation.

4.3. Outcome

To evaluate the feature extraction techniques, the csv files produced for PCA and PFA are loaded to a linear SVM model found in R library individually with a cost function of 10. For both feature sets, the first 5485 rows used for constructing the model and the rest 2189 used for testing. Table 2 and Table 3 shows the confusion matrix gained for the PCA and PFA models when they are tested against their respective test sets.

	acq	crude	earn	grain	interest	money.fx	ship	trade
acq	679	6	8	0	1	2	7	1
crude	3	114	1	0	0	0	2	0
earn	12	1	1074	0	0	1	2	0
grain	0	0	0	9	0	0	0	0
interest	0	0	0	1	65	5	0	0
money.fx	0	0	0	0	13	77	0	1
ship	0	0	0	0	0	0	25	0
trade	2	0	0	0	2	2	0	73

Table 2: confusion matrix produced using PCA feature extraction

[3] The R-code can be found at https://github.com/MichaelAbebaw/Principal-Feature-Analysis-PFA

	acq	crude	earn	grain	interest	money.fx	ship	trade
acq	683	7	11	0	1	2	10	1
crude	3	110	0	0	0	0	1	0
earn	8	1	1072	0	0	1	0	0
grain	0	0	0	10	0	0	0	0
interest	0	0	0	0	64	3	0	0
money.fx	0	0	0	0	15	79	0	0
ship	0	2	0	0	0	0	26	0
trade	2	1	0	0	1	2	0	74

Table 3: confusion matrix produced using PFA feature extraction algorithm

From the confusion matrixes the recognition rate can be calculated by taking the ratio of correctly classified documents to the total number of documents which gives 96.66% and 96.71% for PCA and PFA respectively.

5. Discussion

The concern of this study is to evaluate a feature reduction technique, PFA, that is introduced by (Lu et al. 2007) in text classification problem. The algorithm is a modified version of PCA where instead of projecting the whole dataset to a new space, it picks features from the original space that correspond to the cluster centroids of the principal components. The algorithm is tested against a benchmark text corpus, the Reuters R8 split, and produced a recognition rate of 96.71% which is a comparable result to PCA that produced a recognition rate of 96.66%.

The results gained from this study go in line with the findings obtained by (Lu et al. 2007) where the algorithm was tested against image database to track important points in facial motions and content-based image retrieval problems where feature computation is an expensive task. There, the algorithm is able to pick the optimal features from the original feature set and produced similar results to PCA.

The above results suggest that PFA can be applied in sparse and computationally expensive feature sets and is able to produce an optimal feature set. In addition, since PFA takes its features from the original feature set, it can also be used as a tool for automatically indexing newly observed documents as the optimal features, i.e., terms, are already known.

To further know about the feature reduction technique in text classification problems the algorithm needs to be tested on larger corpora where multi-classes exit which this study did not cover because of the time frame given for the study. In addition, it would be interesting to explore its performance in constructing Realtime Indexing Systems for text data.

In conclusion, this study tried to evaluate PFA in text categorization problems in contrast to a well-known technique in the field PCA. Both algorithms are applied in benchmark text corpus and produced a comparable performance. From the results, it is inferred that the algorithm is performs well in identifying discriminative terms that exist in documents and can be safely applied in feature reduction problems where feature sparsity is a problem. However, more studies needs to be done to know the performance in multi-class problems.

References

Aggarwal, Charu C., and ChengXiang Zhai. 2012. "A Survey of Text Classification Algorithms." In *Mining Text Data*, 9781461432:163–222. doi:10.1007/978-1-4614-3223-4_6.

———. 2013. "Mining Text Data." In *Mining Text Data*, 1–522. doi:10.1007/978-1-4614-3223-4.

Anjali, Ganesh Jivani. 2011. "A Comparative Study of Stemming Algorithms." *IJCTA* 2 (2004): 1930–38. doi:10.1.1.642.7100.

Creswell, John W. 2007. *Qualitative Inquiry & Research Design*. *Sage Publications, Inc.* 2nded. Thousand Oaks: Sage Publications, Inc. doi:10.1111/1467-9299.00177.

Debole, Franca, and Fabrizio Sebastiani. 2004. "An Analysis of the Relative Difficulty of Reuters-21578 Subsets." *Proceedings of LREC-04, 4th International Conference on Language Resources and Evaluation* . http://www.math.unipd.it/~fabseb60/Publications/LREC04.pdf.

Engel, Daniel, Lars Hüttenberger, and Bernd Hamann. 2011. "A Survey of Dimension Reduction Methods for High-Dimensional Data Analysis and Visualization." In *OASIcs-OpenAccess Series in Informatics*, 27:135–49.

Feldman, Ronen, and James Sanger. 2007. *Hand Book of Text Mining*. *Cambridge University Press*. New York: Cambridge University Press. doi:10.1017/CBO9781107415324.004.

Fodor, Imola K. 2002. "A Survey of Dimension Reduction Techniques." *Lawrence Livermore National Laboratory*. https://e-reports-ext.llnl.gov/pdf/240921.pdf.

Fuka, Karel, and Rudolf Hanka. 2001. "Feature Set Reduction for Document Classification Problems." *IJCAI-01 Workshop: Text Learning: Beyond Supervision*.

Gupta, Vishal, and Gurpreet S. Lehal. 2009. "A Survey of Text Mining Techniques and Applications." *Journal of Emerging Technologies in Web Intelligence* 1 (1): 60–76. doi:10.4304/jetwi.1.1.60-76.

Han, Jiawei, and Micheline Kamber. 2006. *Data Mining: Concepts and Techniques*. 2nded. San Francisco: Morgan Kaufmann Publishers. doi:10.1128/AAC.03728-14.

IBM. 2014. "What Is Big Data?" http://www-01.ibm.com/software/data/bigdata/what-is-big-data.html.

Ikonomakis, M, S Kotsiantis, and V. Tampakas. 2005. "Text Classification Using Machine Learning Techniques." *WSEAS TRANSACTIONS on COMPUTERS* 4 (8): 966–74.

Johannesson, Paul, and Erik Perjons. 2012. *A Design Science Primer*. Edited by 1. 1sted. Stockholm.

Krzanowski, W. J. 1996. "A Stopping Rule for Structure-Preserving Variable Selection." *Statistics and Computing* 6 (1): 51–56. doi:10.1007/BF00161573.

Krzanowski, W.J. 1986. "Royal Statistical Society." *Applied Statistics- Journal of the Royal Statistical Society Series* 36 (1): 22–33.

Lu, Yijuan, Ira Cohen, XS Zhou, and Q Tian. 2007. "Feature Selection Using Principal Feature Analysis." *Proc. Int. Conf. on Multimedia*, 301–4. doi:10.1145/1291233.1291297.

McCabe, George P. 1982. "Principle Variables."

Sahlgren, Magnus. 2005. "An Introduction to Random Sets." In *Proceedings of the Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005 (2005)*, 1–9. Copenhagen. doi:10.1201/9781420010619.

Sebastiani, Fabrizio. 2002. "Machine Learning in Automated Text Categorization" 34 (1): 1–47.

Vijayarani, S, and R Janani. 2016. "Text Mining: Open Source Tokenization Tools – An Analysis." *Advanced Computational Intelligence: An International Journal (ACII)* 3 (1): 37–47. doi:10.5121/acii.2016.3104.

Wold, Svante, Kim Esbensen, and Paul Geladi. 1987. "Principal Component Analysis." *Chemometrics and Intelligent Laboratory Systems* 2 (1–3): 37–52. doi:10.1016/0169-7439(87)80084-9.

Zareapoor, Masoumeh, and Seeja K. R. 2015. "Feature Extraction or Feature Selection for Text Classification: A Case Study on Phishing Email Detection." *International Journal of Information Engineering and Electronic Business* 7 (2): 60–65. doi:10.5815/ijieeb.2015.02.08.