On Supervised and Unsupervised Methodologies for Mining of Text Data

Tanmay Basu

Machine Intelligence Unit
Indian Statistical Institute
Kolkata - 700 108, India

A thesis submitted to Indian Statistical Institute
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy
2014

Dedicated to the Lotus Feet of Maa

Acknowledgements

The contents of the thesis are visible to all, but it is not possible to visualize the encouragement of all the well wishers behind this thesis. Few words in this section can not explain all those immeasurable and valuable supports, without which this work would have been futile. Thanksgiving is just a formality, and I do not know how to express my gratitude to all my beloved well wishers.

I would like to express my gratitude and indebtedness to my supervisor Prof. C. A. Murthy. I have got invaluable suggestions from him to organize this thesis. I can recall the evening discussions with him which enriched my knowledge a lot. Above all I want to mention his behavior and gentleness, the very first things which attracted me to select him as my PhD supervisor. The other significant characteristics in him is to maintain the morality, which has thoroughly been followed in this thesis.

I want to say a few words about my parents who insisted me to carry on higher studies in spite of several inconveniences faced by them. I could not have reached this height, if they had not suggested me to pursue research by leaving the prestigious jobs that I had already bagged. I am also indebted to my dearest younger brother Tathamay, elder sister Tanusri and brother-in-law Aninda for their constant support and encouragement. I want to mention the well wishes and supports that I have received all through from the other family members, specially my aunt Sima Sarkar and uncle Kamal Sarkar.

I express my heartfelt gratitude and respect to my teacher cum mentor Swami Sarvottamananda (Shreesh Maharaj). He taught computer science in my undergraduate days. From those days he injected the flavor of knowledge into us and this thesis is just an outcome of it. Other than my parents, he is the only one who suggested me to pursue doctoral study. I want to thank all the teachers of my undergraduate and graduate studies, without those teachings I might not have reached this position.

I owe my sincere gratitude to Dr. Mandar Mitra, who was the mentor of my Master's thesis. I might not have opted for text mining as my research area if I had not worked under him. I also got the very first idea of text and web mining from him. He helped me a lot during my PhD tenure in various academic issues.

I wholeheartedly thank all my friends for their constant encouragement and support, specially Abhirup, Abhisek, Ajoy, Anirban, Apurba, Badri, Bibek, Chinmoy, Chintan, Debashis da, Fatik, Jaydeb, Malay, Manish, Murthy, Partha, Pagla, Prabhu, Prabir, Prasenjit, Rahul, Ranajit da, Sourangshu da, Sourav da, Subrata, Sudeb, Suman da, Swapan and Swarup da.

I want to express my thankfulness to all the faculty members of my department. I would also like to acknowledge the support that I have received throughout from the office staff of our department during the tenure of my PhD. I am indebted to the Dean of Studies of the Institute for providing me fellowship, travel grants and after all a good academic environment. I express my sincere thanks to the authorities of ISI for the facilities extended to carry out research work and for providing me every support during this tenure. I want to thank all the others, whom I might have missed here, for their well wishes and support.

Indian Statistical Institute — Tanmay Basu
July 31, 2014

Abstract

The supervised and unsupervised methodologies of text mining using the plain text data of English language have been discussed. Some new supervised and unsupervised methodologies have been developed for effective mining of the text data after successfully overcoming some limitations of the existing techniques.

The problems of unsupervised techniques of text mining, i.e., document clustering methods are addressed. A new similarity measure between documents has been designed to improve the accuracy of measuring the content similarity between documents. Further, a hierarchical document clustering technique is designed using this similarity measure. The main significance of the clustering algorithm is that the number of clusters can be automatically determined by varying a similarity threshold of the proposed similarity measure. The algorithm experimentally outperforms several other document clustering techniques, but it suffers from computational cost. Therefore another hybrid document clustering technique has been designed using the same similarity measure to overcome the computational burden of the proposed hierarchical algorithm, which performs better than the hierarchical one for most of the corpora.

The limitations of nearest neighbor decision rule for text categorization are discussed. An efficient nearest neighbor decision rule is designed for qualitative improvement of text categorization. The significance of the proposed decision rule is that it does not categorize a document when a decision in not so certain. The method is showing better results than several other classifiers for text categorization. The decision rule is also implemented using the proposed similarity measure instead of traditional similarity measure, which performs better than the same using traditional similarity measure.

The importance of dimensionality reduction for text categorization is also discussed. A supervised term selection technique has been presented to boost the performance of text categorization by removing redundant and unimportant terms. The empirical studies have shown that the proposed method has improved the quality of text categorization.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The conventional form of storing data is text. From the ancient time we communicate and share our views and expressions through letters, articles, books, leaflets, newspapers etc., which are nothing but a collection of texts. In those days it was very difficult to preserve the collection of information for future reference. With the progress of science and technology it has become easy to store a huge amount of information in concise form for ever. But the size of text is growing exponentially after the invention of internet. Now a days most of the information are available on the web, e.g., newspapers, books, articles, etc. People are sharing (or interacting) their views and voices in different blogs and social network sites. In the present days the communication has become very easy by the SMSes and Emails. Technically all of these examples are nothing but a collection of texts. Hence it has become imperative to effectively handle the huge collection of text data in efficient manner.

Text mining refers to a system that identifies useful knowledge from a huge amount of natural language text. The task is challenging as most of the text data sets are unstructured. The enormous amount of information stored in unstructured texts cannot simply be used for further processing by computers, which typically handle text as simple sequences of character strings [62]. Hence it is very difficult to retrieve useful information from a new text data set. Specific pre-processing methods and algorithms are required in order to extract useful patterns. Text mining is an interdisciplinary field that uses the techniques of information retrieval [89], natural language processing [30, 110] and especially data mining [127].

Information retrieval is the activity of obtaining a specific information from a collection of information resources. Suppose there is a collection documents and

a person wants to find a particular information which is available in some of the documents. Information retrieval techniques handle these issues to satisfy the needs of the users efficiently.

The task of natural language processing (nlp) is to understand the significance of a text. The techniques under use are machine learning, statistics and linguistics. The techniques available use either the computational approach or the semantic relationship approach between text segments with the help of a set of grammars and dictionaries (depending on the language) for identifying the actual information. It has a variety of application areas e.g., opinion mining [101], named entity recognition [93], word sense disambiguation [99] etc.

Data mining refers to the task of finding useful patterns from any form of data, whereas text data mining refers to the task of refining the unstructured text to a standard form and then finding useful patterns from that text data.

The two similar emerging fields of text data mining are web data mining and social network mining. The text data in the web is partially structured by the markups and hyperlinks. Several well known script languages for the markups are available e.g., HTML, XML, which follow certain rules to maintain a webpage. The hyperlinks of a webpage provides the relation of the page to another web page, the same is not available in plain text data. The social network data are far more structured than the web data. It is a collection of social web sites connected in a network. In this data each node (a social web page) is mapped to every other node through several inward and outward edges. The idea of network theory can be easily applied to these data sets to retrieve useful information. It may be noted that it is very difficult to simply apply the data mining techniques to the plain unstructured text data. Initially some refinement techniques are required to appropriately represent the texts such that the data mining techniques can be effectively applied on those data.

Text data mining for the plain text data of English language only is discussed in this thesis. Text data mining is referred as text mining throughout the thesis. The main tasks of text mining are;

1) Refinement of text by stopword removal and stemming,

2) Representation of refined plain text, and

3) Extracting useful information from the plain text by applying supervised or

unsupervised methodologies.

1.1 Text Refinement

In order to obtain all terms that are used in a given text, a tokenization process is required, i.e. a text document is split into a stream of terms by removing all punctuation marks and by replacing tabs and other non-text characters by single white spaces. This tokenized representation is then used for further processing. The set of different terms obtained by merging all text documents of a collection is called the vocabulary of a document collection [62].

1.1.1 Stopword Removal

A standard filtering method for English text data is stopword removal. The idea of stopword removal is to remove terms that bear little or no content information, like articles, conjunctions, prepositions, etc. Furthermore, terms that occur extremely often can be said to be of little information content to distinguish between documents, and also terms that occur very seldom are likely to be of no particular statistical relevance and can be removed from the vocabulary [50]. In order to further reduce the number of terms from the vocabulary, term selection methods can be used [62].

1.1.2 Stemming

In a language like English, a typical term contains a stem which refers to some central idea or meaning. Generally certain prefixes or suffixes are added to modify the meaning to fit the term for its syntactic role [100]. The purpose of stemming is to relate the morphological variants of a term, e.g., *statistic, statistics and statistical* will be mapped to the stem *statis*. A stem is the portion of a term that is left after removing its prefixes or suffixes. Consider a query containing the term statistics, which may be relevant to the document consisting of the term statistic or statistical. It would not make the query and the documents relevant to each other by simple matching (i.e., without stemming). Thus stemming has become famous in information retrieval literature [88]. Several stemming algorithm have been proposed over the years, but most popular and robust stemmer for English text is the Martin Porter's stemming algorithm [70, 85]. The algorithm follows a set of rules

for stemming which can be found in the article by Ali et al. [3, 104]. The method has five steps, and within each step, rules are applied until one of them passes the conditions to a different step [70]. The suffix is removed accordingly, if a rule is accepted and the next step is performed. The resultant stem at the end of the fifth step is returned. The detailed description of the porter stemmer algorithm can be viewed from appendix A.

1.2 Representation of Text Data

The length of different documents in a corpus are different. Note that here length means the number of terms in a document. It is very difficult to find the similarity between two document vectors of different dimensions (length). Therefore it is necessary to maintain the uniform length of all the documents in the corpus. Several models have been introduced in the information retrieval literature to represent the document data sets in the same frame. Probabilistic model, language model and vector space model are three well known techniques among several other techniques for text representation [89].

The *vector space model* enables very efficient analysis of huge document collections in spite of its simple data structure without using any explicit semantic information [62]. It was originally introduced for indexing and information retrieval, but is now used in several text mining approaches as well as in most of the currently available document retrieval systems [111]. The vector space model is used in the entire thesis to represent a document vector.

The vector space model represents documents as vectors in n-dimensional space. Note that the number of documents in the corpus throughout this thesis is denoted by N. The number of terms in a corpus is denoted by n. The i^{th} term is represented by t_i. Number of times the term t_i occurs in the j^{th} document is denoted by $tf_{ij},\ i = 1, 2, ..., n;\ j = 1, 2, ..., N$. Document frequency df_i is the number of documents in which t_i occurs. Inverse document frequency $idf_i = log(\frac{N}{df_i})$, determines how frequently a term occurs in the document collection. The weight of t_i in the j^{th} document, denoted by w_{ij}, is determined by combining the term frequency with the inverse document frequency [111] as follows:

$$w_{ij} = tf_{ij} \times idf_i = tf_{ij} \times log(\frac{N}{df_i}),\quad \forall\, i = 1, 2, ..., n \text{ and } \forall\, j = 1, 2, ..., N$$

The documents can be efficiently represented using the vector space model in most of the text mining algorithms [62]. In this model each document d_j is considered to be a vector $\vec{d_j}$, where the i^{th} component of the vector is w_{ij}, i.e., $\vec{d_j} = (w_{1j}, w_{2j}, ..., w_{nj})$.

The similarity between two documents is achieved through some distance function. Given two document vectors $\vec{d_i}$ and $\vec{d_j}$, it is required to find the degree of similarity (or dissimilarity) between them. Various similarity measures are available in the literature but the commonly used measure is cosine similarity between two document vectors [112], which is given by

$$\cos(\vec{d_i}, \vec{d_j}) = \frac{\vec{d_i}.\vec{d_j}}{|\vec{d_i}|\,|\vec{d_j}|} = \frac{\sum_{k=1}^{n}(w_{ik} \times w_{jk})}{\sqrt{\sum_{k=1}^{n} w_{ik}^2 \times \sum_{k=1}^{n} w_{jk}^2}}, \quad \forall i, j \tag{1.1}$$

The weight of each term in a document is non negative. As a result the cosine similarity is non negative and bounded between 0 and 1, both inclusive. $cos(\vec{d_i}, \vec{d_j}) = 1$ means the documents are exactly similar and the similarity decreases as the value decreases to 0. An important property of the cosine similarity is its independence of document length. Thus cosine similarity has become popular as a similarity measure in the vector space model [64].

1.3 Supervised and Unsupervised Methodologies for Text Data

The task of the supervised or the unsupervised learning methodologies is to group the given objects into some particular categories by applying prior knowledge or by finding the relationships between the objects. The process in supervised or unsupervised methodologies for mining plain text consists of the following stages: a) feature selection and b) development of supervised or unsupervised learning methodologies. All these stages are very much application specific. Note that the main contributions of the thesis are the development of new supervised and unsupervised techniques to find useful patterns from the plain text data.

1.3.1 Feature Selection Methods

The task of feature subset selection is to select a number of important features from the set of all features without sacrificing the quality of the learning process [67]. Consider F as the given set of features and the task is to select γ number of features from F. Let the criterion function for feature selection be $Cr(F)$. Let us assume that a higher value of Cr indicates a better feature subset. The task is to find an optimum subset f for which $Cr(f)$ is maximum among all subsets of cardinality γ. Different methods are available for feature subset selection in statistics, machine learning and pattern recognition using several search strategies and evaluation functions. Mainly there are two types of search strategies - optimal search and sub optimal search, for feature subset selection [39].

Any optimal subset search approach initially examines all $\binom{|F|}{\gamma}$ subsets of size γ, and then selects the subset with the largest value of Cr. In this search strategy the number of possible subsets grows combinatorially, which makes the exhaustive search impractical for even moderate values of $|F|$ and γ [67].

1.3.1.1 Suboptimal Search

In practical scenario the optimal search for feature subset selection is not feasible due to computational cost, though the optimal search techniques can guarantee an optimal solution. The sub optimal feature subset selection techniques are used to manage the burden of computational cost and so these techniques are useful for the data sets with large number of features. Note that the sub optimal search techniques never guarantee an optimal solution [39]. Some sub optimal search techniques are discussed below.

- **Feature Ranking**: The simple and most effective sub optimal technique is feature ranking [39]. The method is extremely dependent on the criterion function and so it does not always possess a realistic solution. The method ranks each feature $t_i,\ i = 1, ..., |F|$ according to their $Cr(t_i)$ values and selects the top γ number of features from F.

- **Sequential Forward Selection**: The method is a bottom up search procedure and it selects the best single feature and adds it to the already selected feature subset [67]. The main drawback of this method is that once a feature is added to the feature subset, it is never removed in a later stage [39].

- **Sequential Backward Selection**: This technique is a top down search technique. It starts with a feature set containing all features and greedily removes one feature at a time until $|F| - \gamma$ features have been deleted [39].

- **Generalized Sequential Forward Selection**: This technique is a forward selection technique. Instead of adding one feature at a time, l best features are added at each stage of the algorithm. The value of l is to be chosen in such a way that ultimately γ features are obtained.

- **Generalized Sequential Backward Selection**: The method is a backward selection technique. Instead of discarding one feature at a time, this method removes r worst features at each stage. The value of r is to be chosen in such a way that ultimately γ features are obtained.

- **Plus l take away r Selection**: This is a split and merge feature selection technique, and this is better known as (l, r) *algorithm*. If the method starts with empty set of features, then in each iteration, l best features are added, and r worst features are removed. That is, $l > r$. On the other hand, if the method starts with the complete set of features F, then, in each iteration, r worst features are removed and l best features are added. That is, $r > l$ [39]. l and r are to be chosen appropriately so that the required number of features γ can be obtained.

In the text data each unique term of the vocabulary is considered to be a feature of that data set. The number of features (or terms) of a text data set may be a large number (of the order of several thousands) and the number of features may exceed the number of documents of the corpus by more than an order of magnitude in general [49]. The techniques available in machine learning for feature subset selection are generally not designed for the data sets with large number of features [92]. Hence the term selection methods for text data are very simple compared to the methods available in the literature of machine learning [92]. A number of research works have been done on term selection for text data [77]. There are two types of term selection techniques for text data - supervised and unsupervised. Some well known supervised and unsupervised term selection methods for dimensionality reduction of the text data will be discussed now. It may be noted that the task of stopword removal can be thought of as a feature removal task.

1.3.1.2 Supervised Term Selection Methods

Supervised methods for text data mining assume that the number of categories present in the data set is known. Generally, prior knowledge is available about every category present in the data set. Usually, the knowledge available is in the form of a few documents belonging to each category. The problem to be tackled is to classify an unknown document to one of the existing categories in the data set.

It may be noted that, usually, the experimenter has the knowledge of all the categories existing in the data set, but there are some cases where an experimenter does not possess the knowledge of all the existing categories [84]. In such a case, the formulation of the problem, as well as the analysis is different from the usual methodologies. Here, in this thesis, it is assumed that the experimenter has the knowledge of all the existing categories in the data set. For each existing category, the number of documents belonging to that category is known.

Each document contains some terms and the category of the document is known for the task of text categorization. Thus the category of each term is known for the supervised term subset selection methods for text categorization. Note that text categorization groups the documents into some predefined categories. The supervised term selection methods rank the terms based on a criterion function and then use the best subset of terms for text categorization. These methods use an evaluation function that is applied to every term, and all the terms are independently evaluated. Subsequently, a score is assigned to each of the terms [92]. The terms are then sorted according to those scores and a predefined number of best terms form the resultant subset of terms. Various techniques for ranking the terms for text categorization are available in the literature. Some well known criterion functions for term ranking in text categorization are discussed here. Several notations which have been used throughout the thesis are given below initially before the definitions of the criterion functions.

Let us assume that t is a term, $C = \{c_1, c_2, ..., c_m\}$ is the set of m categories, $(m > 1)$ and N is the number of documents in the training set. Let us consider the following functions for a term t and a category $c_i,\ i = 1, 2, ..., m$ to describe some criterion functions to rank the terms for text categorization. Note that the conventions $log(0) = -\infty$ and $0 \log(0) = 0$ are assumed throughout the thesis. It is also assumed that $|c_i| > 0,\ \forall i$.

$u(t, c_i)$: number of documents containing term t and belonging to c_i.

$v(t, c_i)$: number of documents containing the term t but not belonging to c_i.

$w(t, c_i)$: number of documents not containing the term t, but belonging to c_i i.e, $w(t, c_i) = |c_i| - u(t, c_i)$.

$x(t, c_i)$: number of documents that neither contain the term t nor belong to c_i, i.e., $x(t, c_i) = N - |c_i| - v(t, c_i)$.

$P(t)$: Probability of a document that contains term t. It is assumed that the term t occurs in at least one document, i.e., $u(t, c_i) + v(t, c_i) > 0$.

$$P(t) = \frac{u(t, c_i) + v(t, c_i)}{N}$$

$P(\bar{t})$: Probability of a document that does not contain term t.

$$P(\bar{t}) = 1 - P(t) = \frac{w(t, c_i) + x(t, c_i)}{N}$$

$P(c_i)$: Probability of a document that belongs to c_i.

$$P(c_i) = \frac{u(t, c_i) + w(t, c_i)}{N}$$

$P(\bar{c_i})$: Probability of a document that does not belong to c_i.

$$P(\bar{c_i}) = 1 - P(c_i) = \frac{v(t, c_i) + x(t, c_i)}{N}$$

$P(t, c_i)$: Joint probability that a document contains term t and also belongs to c_i.

$$P(t, c_i) = \frac{u(t, c_i)}{N}$$

$P(\bar{t}, c_i)$: Joint probability that a document does not contain term t but belongs to c_i.

$$P(\bar{t}, c_i) = \frac{w(t, c_i)}{N}$$

$P(t, \bar{c_i})$: Joint probability that a document contains term t but does not belong to

c_i.

$$P(t, \bar{c_i}) = \frac{v(t, c_i)}{N}$$

$P(\bar{t}, \bar{c_i})$: Joint probability that a document neither contains term t nor belongs to c_i.

$$P(\bar{t}, \bar{c_i}) = \frac{x(t, c_i)}{N}$$

$P(t|c_i)$: Conditional probability of a document that contains term t given that it belongs to c_i.

$$P(t|c_i) = \frac{u(t, c_i)}{u(t, c_i) + w(t, c_i)}$$

$P(t|\bar{c_i})$: Conditional probability of a document that contains term t given that it does not belong to c_i.

$$P(t|\bar{c_i}) = \frac{v(t, c_i)}{v(t, c_i) + x(t, c_i)}$$

$P(c_i|t)$: Conditional probability of a document that belongs to c_i given that it contains the term t.

$$P(c_i|t) = \frac{u(t, c_i)}{u(t, c_i) + v(t, c_i)}$$

$P(c_i|\bar{t})$: Conditional probability of a document that belongs to c_i given that it does not contain term t.

$$P(c_i|\bar{t}) = \frac{w(t, c_i)}{w(t, c_i) + x(t, c_i)}$$

The *Mutual Information* (MI) [92] between the term t and category c_i is defined (under the assumption that $P(t) > 0$) as

$$MI(t, c_i) = log\frac{P(t|c_i)}{P(t)}$$

This method assumes that the term with higher category ratio is more effective for categorization. On the other hand the method is biased towards low frequency terms

as can be seen from the following form.

$$MI(t, c_i) = logP(t|c_i) - logP(t)$$

Rare terms will have a higher score than common terms for those terms with equal conditional probability $P(t|c_i)$. Hence MI might perform badly when a classifier gives stress on common terms. The MI of a term over all categories may be computed in the following way:

$$MI_{max}(t) = \max\{MI(t, c_i) : \ i = 1, 2, ..., m\}$$

For a training corpus those terms whose MI score is less than a predefined threshold are removed from the vocabulary.

Information Gain (IG) measures the number of bits of information obtained for category prediction by knowing the presence or absence of a term in a document [131]. It is defined as

$$IG(t) = P(t) \sum_{i=1}^{m} P(c_i|t) \log \frac{P(c_i|t)}{P(c_i)} + P(\bar{t}) \sum_{i=1}^{m} P(c_i|\bar{t}) \log \frac{P(c_i|\bar{t})}{P(c_i)}$$

This measure gives more weights to common terms rather than the rare terms. Hence IG might perform badly when there is scarcity of common terms between the documents of the training corpus.

The *Cross Entropy* (CE) of a term t over all categories is defined as follows:

$$CE(t) = P(t) \sum_{i=1}^{m} P(c_i|t) \log \frac{P(c_i|t)}{P(c_i)}$$

The cross entropy used in term selection [74] is designed similar to information gain. The difference between IG and CE is that CE does not consider the non occurrence of the terms like IG. The CE values of all the terms are sorted in decreasing order and the best subset of terms is selected.

Gain Ratio (GR) is defined as the ratio between the information gain of a term

t and the entropy of the system of categories, i.e.,

$$GR(t, c_i) = \frac{\sum\limits_{c \in \{c_i, \bar{c_i}\}} P(t)P(c|t) \log \frac{P(c|t)}{P(c)} + \sum\limits_{c \in \{c_i, \bar{c_i}\}} P(\bar{t})P(c|\bar{t}) \log \frac{P(c|\bar{t})}{P(c)}}{- \sum\limits_{c \in \{c_i, \bar{c_i}\}} P(c) \log P(c)}$$

The value of GR of a term t and category c_i lies between 0 and 1. The larger the value of GR, the more important the term is for category prediction [38]. The maximum GR value among all the categories is selected as the gain ratio for a term t. The GR values of all terms are sorted in decreasing order and the best subset of terms is selected accordingly.

The χ^2 *statistic* (CHI) measures the association between a term t and a category c_i [55]. CHI score is defined as

$$\chi^2(t, c_i) = \frac{N \times \left[P(t, c_i)P(\bar{t}, \bar{c_i}) - P(t, \bar{c_i})P(\bar{t}, c_i)\right]^2}{P(t) \times P(\bar{t}) \times P(c_i) \times P(\bar{c_i})}$$

The χ^2 statistic of a term over all categories can be defined in the following two ways:

$$\begin{aligned} \chi^2_{avg}(t) &= \sum_{i=1}^{m} P(c_i)\chi^2(t, c_i) \\ \chi^2_{max}(t) &= \max\{\chi^2(t, c_i) : \ i = 1, 2, ..., m\} \end{aligned}$$

Bi-Normal Separation (BNS) is proposed by Forman [49] as

$$BNS(t, c_i) = \left| F^{-1}\left(\frac{P(t, c_i)}{P(c_i)}\right) - F^{-1}\left(\frac{P(t, \bar{c_i})}{P(\bar{c_i})}\right) \right|$$

where F is the distribution function for standard normal distribution and F^{-1} is the inverse of F. For $F^{-1}(0), 0$ is substituted by 0.0005 to avoid the undefined value $F^{-1}(0)$. The larger the value of BNS, the larger the difference between the prevalence of term t in category c_i and $1 - P(c_i)$. The author [49] made a comparative study of twelve term selection methods on 229 text categorization problem instances and the experiments showed that BNS can perform very well in the evaluation metrics of recall rate and F-measure. But for precision, its performance was often not good as IG.

Odds Ratio (OR) measures the odds of term t occurring in category c_i, $i = 1, 2, ..., m$, divided by the odds of the term t not occurring in category c_i [92]. It is defined as follows:

$$OR(t, c_i) = \frac{P(t|c_i) \times \big(1 - P(t|\bar{c_i})\big)}{P(t|\bar{c_i}) \times \big(1 - P(t|c_i)\big)}$$

$OR(t, c_i) > 1$, when the odds of term t occurring in documents of c_i is greater than the odds of term t not occurring in documents of c_i. $OR(t, c_i) = 1$, when the odds of term t occurring in documents of c_i is the same as the odds of term t not occurring in documents of c_i. $OR(t, c) < 1$, when the odds of term t occurring in documents of c_i is smaller than the odds of term t not occurring in documents in category c_i. The odds ratio is estimated as

$$OR(t, c_i) = \frac{\big(u(t, c_i) + 0.5\big) \times \big(x(t, c_i) + 0.5\big)}{\big(w(t, c_i) + 0.5\big) \times \big(v(t, c_i) + 0.5\big)}$$

Here 0.5 is added to each observed frequency to avoid the extreme cases of a few or all the frequencies becoming zero [81]. The maximum odds ratio among all the categories is selected as the odds ratio value for a term.

A *Gini Index* (GI) based term selection method for text categorization was introduced by Shang. et al. [114]. The original form of the gini index algorithm was used to measure the impurity of terms towards categorization. The aim is to minimize the impurity to obtain the best subset of terms. The gini index of a term t over all categories is defined as

$$GI(t) = \sum_{i=1}^{m} P(t|c_i)^2 P(c_i|t)^2$$

In this formula, if t appears in every document of a particular category c_i and it does not occur in any other category, then the maximum value, $GI(t) = 1$, is obtained [114].

Several other investigations on term selection techniques for text categorization have been done such as Yang et al. [131] investigated five term selection methods and reported that good term selection methods improve the categorization accuracy with an aggressive term removal using DF, IG and CHI methods. Shoushan et al. [83] developed a new term selection method - weighted frequency and odds

for text categorization. Mladenic et al. [92] introduced some new term scoring measures based on odds ratio for large text data sets and web documents. Jana et al. [95] presented sequential forward selection methods based on an improved mutual information measure for text categorization. Basu et al. proposed an evaluation function *term relevance* for term selection in text categorization [10]. The function is designed heuristically, but has shown good performance on several well known text corpora. They have designed another evaluation function, *term significance* for term selection, which favors the common terms to construct the best subset of terms [9]. Zheng et al. [136] investigated a combination of term selection algorithms for text categorization on imbalanced data sets. The data set where the training samples are unevenly distributed among different categories is known as imbalanced data. Chen et. al [27] have developed two term evaluation metrics: Multi-class Odds Ratio and Class Discriminating Measure for the naive bayes classifier applied on multi-class text data sets. Uysal et al. proposed a novel filter based probabilistic term selection method, namely distinguishing feature selector for text classification [122]. A new term selection algorithm have been presented by Yang et al. [129], which comprehensively measures the significance of a term both in inter-category and intra-category. Feng et al. have designed a generative probabilistic model, describing categories by distributions and handling the term selection problem by introducing a binary exclusion or inclusion latent vector, which is updated via an efficient stochastic search search [46].

1.3.1.3 Unsupervised Term Selection Methods

The task of an unsupervised term selection method is to select important terms for the underlying clusters without using any prior knowledge of the data.

Document Frequency (DF) thresholding is the simplest technique for vocabulary reduction in text categorization [131]. Document frequency denotes the number of documents in which a term occurs. The document frequency of each term in the training corpus is computed and the terms with a document frequency less than a predefined threshold are discarded from the vocabulary. The DF thresholding method assumes that the terms with higher document frequency are more informative for categorization. But this assumption may sometimes lead to a poor categorization accuracy if a term occurs in most of the documents in each category (e.g., stopwords). The computational complexity of this method is approximately

linear to the number of documents in the training set. Hence it is scalable to any large corpus and usually considered as an ad hoc approach for term selection.

Entropy based ranking is proposed by Dash et al. [37]. In this method, a term is measured by the entropy reduction when it is removed. The entropy of a term is defined as follows:

$$E(t) = \sum_{i=1}^{N}\sum_{j=1}^{N}\Big(Sim_{d_i,d_j} \times \log Sim_{d_i,d_j} + (1 - Sim_{d_i,d_j}) \times \log(1 - Sim_{d_i,d_j})\Big)$$

where Sim_{d_i,d_j} is the similarity between documents d_i and d_j. *Sim* is defined as below.

$Sim_{d_i,d_j} = e^{\alpha \times \rho(d_i,d_j)}, \alpha = -\ln(0.5/\overline{\rho})$, where $\rho(d_i, d_j)$ is the distance between d_i and d_j after removing the term t. $\overline{\rho}$ is the average distance among the documents after removing t. The computational complexity of this method is $O(nN^2)$ for n number of terms, which is a serious problem of this method [86].

Liu et al. [86] developed two new unsupervised term selection methods. First one is *term contribution*, which ranks a term by its overall contribution to the document similarity in a data set. Another is *iterative feature selection*, which utilizes a successful term selection criterion function, such as IG or CHI, to iteratively select terms and perform document clustering at the same time. Dasgupta et al. [35] presented an unsupervised term selection algorithm and applied it to the regularized least square classification technique. The algorithm assigns a univariate score to every term and then randomly samples a small number of terms (independent of the total number of terms, but dependent on the number of documents and an error parameter), and solves the regularized least square classification problem induced on those terms. In addition the authors [35] have given a theoretical justification which provides worst case guarantees on the generalization power of the resultant classification function using the sample subset of terms with respect to that of the original classification function by using all the features. In a study by Tsivtsivadze et al., they have described the main ideas behind kernels for text analysis in particular, as well as provided an example of designing feature space for parse ranking problem with different kernel functions [121]. Pahikkala et al. have designed a framework that allows systematic incorporation of word positions and facilitates the efficient use of similarity information of words and their positions in the natural language text [98]. The framework is suitable for disambiguation tasks of natural

language texts, where the aim is to select a particular property of a word from a set of candidates based on the context of the word.

1.3.2 Clustering of Text Data

The task of clustering is to segregate data into groups of similar objects [17]. Intuitively, patterns within a valid cluster are more similar to each other than the patterns belonging to two different clusters [67]. It is also known as the unsupervised classification of data as the method of clustering need not require any prior knowledge about the nature of the data. Thus it has become useful for grouping the data objects when no information is available about the characteristics of the data. Hence clustering is a widely used technique in various pattern recognition problems e.g., document retrieval, image segmentation, clustering of text data, social network analysis etc. In this thesis the methods of clustering are discussed in the perspective of text data only, which is better known as document clustering.

Document clustering methods partition a set of documents into different clusters such that the documents in the same cluster are more similar to each other than documents in different clusters according to some similarity or dissimilarity measure. A pairwise document similarity measure plays the most significant role in any document clustering technique. Any document clustering algorithm first finds the document similarity and then groups similar documents into a cluster. There are two basic types of document clustering techniques available in the literature - *hierarchical* and *partitional* clustering techniques [73].

1.3.2.1 Hierarchical Clustering

Hierarchical clustering produces a hierarchical tree of clusters where each individual level can be viewed as a combination of clusters in the next lower level. This hierarchical structure of clusters is also known as dendrogram. The hierarchical clustering techniques can be divided into two parts - *agglomerative* and *divisive*. In an *agglomerative hierarchical clustering* (AHC) method [116], starting with each document as individual cluster, at each step, the most similar clusters are merged until a given termination condition is satisfied. In a divisive method, starting with the whole set of documents as a single cluster, the method splits a cluster into smaller clusters at each step until a given termination condition is satisfied. Several terminating

criteria for AHC algorithms have been proposed, but no universally acceptable terminating criterion is available for these algorithms. As a result some good clusters may be merged which will be eventually meaningless to the user. There are mainly three variations of AHC techniques - single-link, complete-link and group-average hierarchical method for document clustering [32].

Let $\rho(a, b)$ denote similarity between the two documents a, b. Let $A = \{d_{11}, d_{12}, ..., d_{1x}\}$ and $B = \{d_{21}, d_{22}, ..., d_{2y}\}$ denote two clusters of documents. Then, for *single-link* method the similarity between A and B is calculated as

$$\max\{\rho(d_{1i}, d_{2j}) : i = 1, ..., x,\ j = 1, ..., y\}$$

The *complete-link* method measures the similarity between A and B as

$$\min\{\rho(d_{1i}, d_{2j}) : i = 1, ..., x,\ j = 1, ..., y\}$$

The *group average* method merges two clusters if they have the larger average similarity than any other pair of clusters and average similarity between A and B is calculated as

$$\frac{1}{xy}\sum_{i=1}^{x}\sum_{j=1}^{y}\rho(d_{1i}, d_{2j})$$

In a *divisive hierarchical clustering* technique, initially, the method assumes the whole data set as a single cluster. Then at each step, the method chooses one of the existing clusters and splits it into two. The process continues till only singleton clusters remain or it reaches a given halting criterion. Generally the cluster with the least overall similarity is chosen for splitting [116].

1.3.2.2 Partitional Clustering

In contrast to hierarchical clustering techniques, partitional clustering techniques allocate data into a previously known fixed number of clusters. The commonly used partitional clustering technique is *k-means* [61] method where k is the desired number of clusters. Here initially k documents are chosen randomly from the data set, and they are called seed points. Each document is assigned to its nearest seed point, thereby creating k clusters. Then the centroids of the clusters are computed, and each document is assigned to its nearest centroid. The same process continues

until the clustering does not change, i.e., the centroids in two consecutive iterations remain the same. Generally, the number of iterations is fixed by the user. The procedure stops if it converges to a solution, i.e., the centroids are the same for two consecutive iterations, or the process terminates after a fixed number of iterations. The k-means algorithm has, generally, low computational complexity. In general it takes linear time (linear to the size of the corpus) to build the clusters. In some cases it suffers from high computational cost when the user does not fix the number of iterations (number of iterations can become really large) and the data set size is large or the dimensionality of the data set is very high [6]. The main disadvantage of this method is that the number of clusters is fixed and it is very difficult to select a valid k for an unknown text data set. Also there is no proper way of choosing the initial seed points. The method is sensitive to the initial seed points and may get stuck in the local optima [61]. An improper choice of seed points may lead to clusters of poor quality.

Bisecting k-means method [116] is a variation of basic k-means algorithm. This algorithm tries to improve the quality of clusters in comparison to the clusters produced by k-means algorithm. In each iteration, it selects the largest existing cluster (the whole data set in the first iteration) and divides it into two subsets using k-means (k=2) algorithm. This process is continued till k clusters are formed. It produces clusters of almost uniform size. Thus bisecting k-means algorithm can perform better than k-means algorithm when the actual groups of a data set are almost of similar size, i.e., the number of documents in the categories of a corpus are close to each other. On the contrary, the method produces poor clusters for the corpora, where the number of documents in the categories differ very much. This method also faces difficulties like k-means clustering technique, in choosing the initial centroids and a proper value of the parameter k.

Buckshot algorithm [32] is a combination of basic k-means and hierarchical clustering methods. It tries to improve the performance of k-means algorithm by choosing better initial seed points. Initially it randomly selects $\sqrt{kN}$ documents (N is the number of documents in the corpus) from the data set as sample documents and performs AHC on these sample documents. The centroids of the k clusters on the sample documents are the initial seeds for the whole collection. The basic k-means algorithm with these seed points is applied to partition the whole document set [1]. Repeated calls to this algorithm may produce different partitions. The main disadvantage of buckshot is the random selection of initial $\sqrt{kN}$ documents for hi-

erarchical clustering in the first stage, where N is the number of documents and k is the number of clusters [102]. The resulting clusters will be of poor quality, if the initial random sampling does not represent the whole data set properly. Note that appropriate value of k is necessary for this method too.

The *k-Nearest Neighbor* (kNN) technique is mostly known to be used for classification [33], it has also been used for clustering [24, 60]. It utilizes the property of k nearest neighbors, i.e., a document should be put in the same cluster to which most of its k nearest neighbors belong. Merge the documents d_1 and d_2 to form a cluster, if d_1 and d_2 share at least k nearest neighbors and d_1, d_2 are k-nearest neighbors of each other. The performance of the algorithm is highly dependent on the parameter k and choosing a proper value of the k is difficult for text data sets.

1.3.2.3 Spectral Clustering

Spectral clustering technique is a very popular method which works on the similarity matrix rather than the original data matrix using the idea of graph cut. It uses the top eigenvectors of the similarity matrix derived from the similarity between documents [94]. The basic idea is to construct a weighted graph from the initial data set where each node represents a pattern and each weighted edge represents the similarity between two patterns. In this methodology the clustering problem is formulated as a graph cut problem, which can be tackled by means of the spectral graph theory. The core of this theory is the eigenvalue decomposition of the Laplacian matrix of the weighted graph obtained from data [47]. Let $X = \{x_1, x_2, ..., x_N\}$ be the set of N documents to cluster. Let S be the $N \times N$ similarity matrix where S_{ij} represents the similarity between the documents x_i and x_j and $S_{ii} = 0$. Define D to be the diagonal matrix where $D_{ii} = \sum_{j=1}^{N} S_{ij}$. Then construct the Laplacian matrix $L = D - S$ and compute the eigenvectors of L. The corpus is partitioned using $D^{-1/2}e_2$ where e_2 is the eigenvector corresponding to the second largest eigenvalue of L. The same process is continued until k partitions are obtained. But experimentally it has been observed that using more eigenvectors and directly computing a k way partitioning is better than recursively partitioning the corpus into two subsets [4]. Another problem is to find a proper stopping criterion for a large and sparse text data set.

Ng. et al. [94] proposed a spectral clustering algorithm which simultaneously partitions the Laplacian matrix into k subsets using the k largest eigenvectors and they have used a Gaussian kernel on the similarity matrix. The steps of the algorithm

are described below.

i) Form the similarity matrix $S \in \mathbb{R}^{N \times N}$ by using a Gaussian kernel, defined by $S_{ij} = exp(-\frac{\rho(x_i,x_j)}{2\sigma^2})$, where $\rho(x_i, x_j)$ denotes the similarity between x_i and x_j and σ is the scaling parameter. Note that $S_{ii} = 0$.

ii) Compute the diagonal matrix D as described above.

iii) Construct the Laplacian matrix $L = D^{-1/2}SD^{-1/2}$.

iv) Find the k largest eigenvectors of L, say $z_1, z_2, ..., z_k$ and construct the matrix $Z = [z_1, z_2, ..., z_k] \in \mathbb{R}^{N \times k}$ with the eigenvectors as its column.

v) Form the matrix Y by re-normalizing the rows of Z to have unit length, i.e., $Y_{ij} = \frac{Z_{ij}}{\sqrt{\sum_j Z_{ij}^2}}$

vi) Partition Y into k clusters by treating each row of Y as a point in $\mathbb{R}^k$ using k-means algorithm.

vii) Assign $x_i, i = 1, 2, ..., N$ to cluster j, if and only if, the i^{th} row of Y is assigned to j.

The Gaussian kernel is used here to get rid of the curse of dimensionality. The main difficulty of using a Gaussian kernel is that, it is very sensitive to the parameter σ [87]. A wrong value of σ may highly degrade the quality of the clusters. It is extremely difficult to select a proper value of σ for a document collection, since the text data sets are generally sparse with high dimensionality. In the experiments the value of σ is set by search over values from 10 to 20 percent of the total range of the similarity values and the one that gives the tightest clusters is picked, as suggested by Ng. et al. [94]. It should be noted that the method also suffers from the disadvantages of the k-means method, discussed above.

1.3.2.4 Non-negative Matrix Factorization

Non-negative matrix factorization (NMF) has previously been shown to be a useful decomposition for multivariate data. It finds the positive factorization of a given positive matrix [79]. Xu et al. have demonstrated that NMF performs very well

for text clustering in compare to the other similar methods like singular value decomposition and latent sematic indexing [128]. The technique factorize the original term-document matrix D approximately as

$$D \approx UV^T$$

where U is a non-negative matrix of size $n \times m$, and V is an $m \times N$ non-negative matrix. The base vectors in U can be interpreted as a set of terms in the vocabulary of the corpus, while V describes the contribution of the documents to these terms. The matrices U and V are randomly initialized, and their contents iteratively estimated [5]. The non-negative matrix factorization method attempts to determine the U and V, which minimize the following objective function

$$J = \frac{1}{2} \left\| D - UV^T \right\| \tag{1.2}$$

where $\|.\|$ denotes the squared sum of all the elements in the matrix. This is an optimization problem with respect to the matrices $U = [u_{ik}], V = [v_{jk}]$, $\forall i = 1, 2, ..., n, \forall j = 1, 2, ..., N$ and $k = 1, 2, ..., m$ and as the matrices U and V are non-negative, we have $u_{ik} \geq 0, v_{jk} \geq 0$. This is a typical constrained non-linear optimization problem and can be solved using the Lagrange method [1]. The objective function continuously improves under the following update rules, and ultimately converges to an optimal solution.

$$u_{ik} \leftarrow u_{ik} \frac{(DV)_{ik}}{(UV^TV)_{ik}}, \qquad v_{jk} \leftarrow v_{jk} \frac{(D^TU)_{jk}}{(VU^TU)_{jk}}$$

It has been shown in the article by Xu et al. [128] that U and V can be normalized in the following way:

$$v_{jk} \leftarrow v_{jk} \sqrt{\sum_{i=1}^{n} u_{ik}}, \qquad u_{ik} \leftarrow \frac{u_{ik}}{\sqrt{\sum_{i=1}^{n} u_{ik}}}$$

The cluster label of each document can be obtained from the matrix V. Precisely, examine each row j of matrix V and assign document d_j to cluster cl, if $cl = \arg\max_k v_{jk}$

The interesting property of NMF technique is that it can also be used to find the

word-clusters instead of document clusters. As the columns of V provide a basis, which are used to determine document clusters, the columns of U can be used to discover a basis, which correspond to word clusters. The NMF has its disadvantages too. The optimization problem of equation 1.2 is convex in either U or V, but not in both U and V, which means that the algorithm can guarantee convergence to a local minimum only. In practice, NMF users often compare the local minima from several different starting points, using the results of the best local minimum found. On large sized problems this may be problematic [78]. Another problem with NMF is that it relies on random initialization and as a result, the same data might produce different results across runs [5].

1.3.2.5 Related Works

Document clustering has been traditionally investigated as a means of improving the performance of search engines by pre-clustering the entire corpus [108]. But it can also be seen as a post retrieval document browsing technique [32]. Several well known document clustering algorithms have been discussed. There are some more document clustering algorithms like the one proposed by Hammouda et al. [60]. They used a graph structure and a document index graph to represent documents and also proposed an incremental clustering algorithm by representing each cluster with a similarity histogram. Density based algorithm is a very famous algorithm for large scale spatial data sets [44]. DBSCAN is a famous and widely used density based clustering technique. The basic idea of DBSCAN is that it apply a local cluster criterion. Clusters are regarded as regions in data space in which the objects are dense and which are separated by regions of low object density [75]. The same can be applied for text data, but it may be difficult to find a local cluster criterion for a sparse and high dimensional text data. Suffix Tree Clustering algorithm were developed by Zamir et al. [133] and used in their meta-search engine. The algorithm is based on identifying phrases that are common to the groups of documents and it is a linear time clustering algorithm (linear to the size of the set of documents) [29]. Note that a phrase is an ordered sequence of one or more terms. Huang et. al presented a clustering method with active learning using Wikipedia [65]. They utilized Wikipedia to create a concept based representation of a text document with each concept associated to a Wikipedia article rather than terms. Banerjee et. all [7] investigated a method of improving the accuracy of clustering short texts by en-

riching their representation with additional features from Wikipedia. Cao et al. [23] proposed an extended vector space model with multiple vectors defined over spaces of entity names, types, name-type pairs, identifiers, and keywords for text searching and clustering. Oikonomakou et al. [96] have shown a comparative study of various document clustering approaches with their merits and demerits. Wang et al. [125] proposed a new method for improving text clustering accuracy based on enriching short text representation with keyword expansion. Jing et al. developed a new knowledge-based vector space model (VSM) for text clustering. In the new model, semantic relationships between terms (e.g., words or concepts) are included in representing text documents as a set of vectors [69]. Millar et al. have developed a document clustering and visualization method based on latent dirichlet allocation and self-organizing maps [91]. The method transforms the word histogram representations of documents into topic distributions, reducing dimensionality in the process. Dasgupta et al. [36] developed a simple active clustering algorithm which is capable of producing multiple clustering of the same data according to user interest. Carpineto et al. have done a very good survey on search results clustering. They have elaborately explained and discussed various issues related to web clustering engines [25]. Wang et al. [124] described an efficient soft-constraint algorithm by obtaining a satisfactory clustering result so that the constraints are respected as much as possible. Zhu et al. [137] presented a semi-supervised nonnegative matrix factorization based on the pairwise constraints - must-link and cannot-link. In this method must-link constraints are used to control the distance of the data in the compressed form, and cannot-link constraints are used to control the encoding factor to obtain a very good performance. Pahikkala et al. [97] proposed a novel unsupervised multiclass regularized least squares classification technique. The resulting kernel-based framework efficiently combines steepest descent strategies with powerful meta-heuristics for avoiding local minima.

1.3.3 Text Categorization Methods

Text categorization is the problem of assigning predefined categories to the new text documents by using the information from the existing documents in those predefined categories [130]. The method is also known as text classification or document classification. A growing number of statistical learning methods have been applied to this problem in recent years. Some useful text categorization techniques are discussed

here.

1.3.3.1 Naive Bayes Decision Rule

The *naive bayes* decision rule is a widely used simple probabilistic learning method for text categorization [89]. Let $T = \{t_1, t_2, ..., t_n\}$, be the set of n terms of the vocabulary and $C = \{c_1, c_2, ..., c_m\}$ be m pre-defined categories of the set of training documents. The objective is to find the best category of a new test document d_0. The multinomial naive bayes decision rule finds the best category of d_0 as the most likely or maximum a posteriori category C_0 as follows [89]:

$$C_0 = \arg\max_{c_j \in C} P(c_j|d) = \arg\max_{c_j \in C} P(c_j) \prod_{i=1}^{n} P(t_i|c_j) \tag{1.3}$$

where $P(c_j)$ is the prior probability of category c_j and $P(t_i|c_j)$ is the conditional probability of a term t_i belongs to category c_j. The prior probability $P(c_j)$ can be estimated as the simple relative frequency of category c_j from the training data, i.e., $P(c_j) = N_j/N,\ \forall c_j \in C$, where N_j is the number of documents in c_j and N is the number of documents in the training set. The conditional probability $P(t_i|c_j)$ can be estimated as

$$P(t_i|c_j) = \frac{f_{ij}}{\sum_{p=1}^{n} f_{pj}}$$

where f_{ij} is the number of occurrences of term t_i in category c_j for every i and j. The product of equation 1.3 for a particular category becomes zero if a term does not occur in that category. It is a serious problem as the term-document matrices are generally sparse. Hence, the integer 1 is added to both numerator and denominator of the conditional probability $P(t_i|c_j)$ to eliminate zeros [89] as stated below.

$$P(t_i|c_j) = \frac{f_{ij} + 1}{\sum_{p=1}^{n} (f_{pj} + 1)} = \frac{f_{ij} + 1}{\sum_{p=1}^{n} f_{pj} + n}$$

Here n is the number of terms in the vocabulary.

1.3.3.2 k-Nearest Neighbor Decision Rule

k-Nearest Neighbour (kNN) decision rule is one of the most fundamental, simple and effective method for classification in the areas of pattern recognition, machine learning, data mining and information retrieval [53]. In 1951, Fix and Hodges [48] introduced the nearest neighbor decision rule and in 1967, Cover and Hart [31] found out the classification error for $k = 1$. It was shown that for $k = 1$ and $N \to \infty$ the classification error is bounded by twice the Bayes error rate, where N is the size of the training set. Several research works on kNN may be found in the book by B. V. Dasarathy [33, 34]. Fukunaga et al. proposed a refinement on kNN rule with respect to Bayes error rate [54]. A study by Yang [130] shows that kNN rule is very useful for text categorization and it has comparable performance to various other classification methods.

Given a new test document d_0 and a set of training documents, the task of kNN rule is to assign d_0 to a particular category. It first finds the k nearest neighbors of d_0 from the training set by a distance function and assigns d_0 to a particular category, say C_0, if the majority of the documents among the k nearest neighbors belong to C_0. k is termed as neighborhood parameter. The cross validation technique is generally used to estimate the value of k [109,117], but effectively choosing an optimal k is still a difficult job. A discussion on the bias and the variance of the posterior probability estimates for different k values is available in the book of Duda and Hart [42] and an article of Friedman [52]. The majority voting of kNN groups d_0 to a category which has maximum representatives among the k nearest neighbors. It may be the case that a new test document is put into a category which has a win by one vote to the next competing category. Sometimes a test document is arbitrarily assigned to a category, if there is a tie between two competing categories. This assignment may not be intuitively satisfying if the test document belongs to the intersection region between categories, where one may not always necessarily be interested to categorize every such document. In practical cases, we may have a mixture of two or more data sets. In such cases the traditional voting process may not work. This issue is elaborately discussed in the next section and chapter 4.

1.3.3.3 Adaptive k-Nearest Neighbor Technique

In the *adaptive nearest neighbor* decision rule, the number of nearest neighbors (k) selected for different categories are adaptive to their sample size in the training

set. Baoli et al. [8] proposed an adaptive kNN rule for text categorization, which is adaptive to the sample size of different categories. The method uses different k values for different categories, rather than a fixed k value for each category. The value of k for each category $c_i,\ i = 1, 2, ..., m$ is determined as

$$adk(c_i) = \Bigg\{ \text{top } h_i \text{ nearest neighbors in traditional } k\text{NN rule:} \\ h_i = \min\left(\alpha + \left\lceil \frac{k \times |c_i|}{\max\limits_{i=1}^{m} |c_i|} \right\rceil, k, |c_i|\right) \Bigg\}$$

Here α is a non-negative integer. It is used to maintain a reasonable minimum value of $h_i,\ i = 1, 2, ..., m$. Without α, h_i may be too small or even equal to 1 for some smaller categories for a training set with a skewed category distribution. Basically to avoid unstable outcomes, a lower bound for h_i is set by selecting a reasonable integer α. When $\alpha = 0$ and the distribution of the categories in a training set is absolutely even, adaptive kNN rule behaves like the traditional kNN rule. The adaptive kNN decision rule determines the category of a new test document d_0 as follows:

$$Cat(d_0) = \arg\max_{i=1}^{m} \frac{\sum\limits_{j \in adk(c_i)} sim(d_0, d_j) \times y(c_i, d_j)}{\sum\limits_{j \in adk(c_i)} sim(d_0, d_j)}$$

where $sim(d_0, d_j)$ is the similarity between the test document d_0 and training document $d_j, j = 1, 2, ..., N$ and $y(c_i, d_j) \in \{0, 1\}$ indicates whether d_j belongs to the category c_i.

1.3.3.4 Distance Weighted k-Nearest Neighbor Technique

The *distance weighted kNN* decision rule gives different weights to different k nearest neighbors based on their distances with the test document d_0, where the closer neighbors get higher weights [43]. Let $d_1, d_2, ..., d_k$ be the k nearest neighbors of d_0 arranged in increasing order of $\rho(d_j, d_0),\ \forall j = 1, 2, ...k$, where ρ is the distance function. Here d_1, d_k are respectively the first and the k^{th} nearest neighbor of d_0.

The weight of the j^{th} nearest neighbor is defined as

$$W_j = \begin{cases} \dfrac{\rho(d_k, d_0) - \rho(d_j, d_0)}{\rho(d_k, d_0) - \rho(d_1, d_0)}, & \text{if } \rho(d_k, d_0) \neq \rho(d_1, d_0) \\ 1, & \text{if } \rho(d_k, d_0) = \rho(d_1, d_0) \end{cases}$$

The test document d_0 is assigned to a particular category for which the weights of the representatives of that category among k nearest neighbors sum to the greatest value.

1.3.3.5 k-Nearest Neighbor Model

The *kNN model* is another variant of kNN rule, proposed by Guo et. al [59]. The method constructs a model from the training documents rather than using the entire training documents, and categorizes new documents using the model. The model is a set of representatives of the training document set, and it is constructed by pruning some documents from the training set [58]. In model construction process, each document has its largest local neighborhood which covers the maximal number of documents of the same category. Based on these local neighborhoods, the largest local neighborhood (called largest global neighborhood) can be obtained in each cycle. This largest global neighborhood is used as a representative to represent all the documents covered by it. For documents not covered by any representative, the above operation is repeated until all the documents have been covered by a chosen representative. The number of documents covered by a representative is the optimal value of k for categorization in the next phase. This k is different for different representatives. Thus k is generated automatically in the model construction process. The new test document, say d_0, is then categorized using the kNN model as follows.

i) Calculate the similarity of d_0 to all representatives in the model M.

ii) If d_0 is covered only by one representative $< Cat(d_j), Sim(d_j), Num(d_j), Rep(d_j) >$, say, and the distance of d_0 to d_j is smaller than $Sim(d_j)$ then d_0 is categorized to the category of d_j. The distance of d_0 to the nearest boundary point of a representative d_j is equal to the difference of the distance of d_0 to d_j minus $Sim(d_j)$.

iii) If d_0 is covered by at least two representatives of different category, assign d_0 to the category of that representative with the largest $Num(d_j)$.

iv) If no representative in the model M covers d_0, assign d_0 as the category of that representative whose boundary points are closest to d_0.

In order to improve the accuracy of kNN model for text categorization, step 3 is modified in the model construction algorithm to allow each local neighborhood to cover r instances with different categories to the majority category in this neighborhood.

1.3.3.6 Support Vector Machines

Support Vector Machines (SVMs) were introduced to solve linearly separable binary classification problems using the Structural Risk Minimization principle [21]. In its simplest form, SVM is a hyperplane that separates a set of positive examples from a set of negative examples with maximum margin. Given a set of training documents in a vector space, SVM finds the best decision hyperplane that separates two categories. The quality of a decision hyperplane is determined by the distance (referred as margin) between two hyperplanes that are parallel to the decision hyperplane and touch the closest documents of each category. Therefore best decision hyperplane is the one with the maximum margin, and is used to categorize the new test documents. The best decision hyperplane is obtained by solving the following optimization problem:

$$\text{minimize } \frac{1}{2}\|\vec{w}^2\| \text{ subject to}$$
$$y_i(\vec{w} \cdot \vec{d_i} - b) \geq 1, \forall i$$

where $\vec{w}$ is the normal vector to the hyperplane, $\vec{d_i},\ i = 1, 2, ..., N$ is the i^{th} document of the training set, y_i is the actual category of d_i and b is a constant. Once the weights are learned, the test documents are categorized by computing

$$sgn(\vec{w'} \cdot \vec{d_0} + b'),$$

where $\vec{w'}$ is the vector of learned weights and b' is the constant of the optimal solution. Here $\vec{d_0}$ is the vector representing the test document and sgn is the sign function. The SVM problem can be solved using quadratic programming. SVM extends its applicability on the linearly non-separable data sets either by using soft margin hyperplanes, or by mapping the original data vectors into a higher dimensional space in which the documents are linearly separable. SVMs use kernels which non

linearly map into a higher dimensional feature space so that a separating hyperplane can be found [119]. An efficient implementation of SVM and its application in text categorization of Reuters-21578 corpus is reported in the study by T. Joachims [71].

The linear kernel is recommended for text categorization because the linear kernel performs nicely when there is a lot of features. The reason is that mapping the data to a higher dimensional space using a non linear kernel does not really improve the performance [63]. Note that the text data sets generally contain large number of documents and the data sets are high dimensional. Hence linear kernel is useful than the non linear kernels for text categorization [71].

1.3.3.7 Related Works

The Generalized Instance Set (GIS) Algorithm [76] is a text categorization algorithm that combines the advantages of kNN rule and linear classifiers. Linear classifiers [82] are a family of text categorization learning algorithms that learn a feature weight vector for every category. The weight learning techniques (such as Rocchio's method and Widrow-Hoff algorithm) are used to learn the feature weight vector from the training samples. Given a category, the vectors of documents belonging to this category are given a positive weight, and the vectors of remaining documents are given a negative weight. By summing up these positively and negatively weighted vectors, the prototype vector of this category is obtained [130]. A Bayesian citation kNN rule with distance weighting for multi-instance learning is developed by Jiang et al. [68] and it is applied to drug discovery data. In the multi-instance learning the training samples are represented by a bag of instances instead of a single instance and the task is to predict the class label of a new bag of instances rather than a single instance. Decision tree is a well known machine learning approach for automatic induction of classification trees based on training documents [106]. Decision tree algorithms are used to select informative terms based on an information gain criterion, and predict categories of each document according to the occurrence of term combination in the document [130]. Mubaid et al. [2] have designed a new text categorization method that combines the distributional clustering of words to generate an efficient representation of documents and applied a learning logic technique, called lsquare [45] for text categorization. Lsquare is a two class classification technique that is based on learning logic. It views the training data as logic formulas and the resulting classifiers are logic formulas as well. The method performed

better than SVM on Reuter-21578 data set. Dhillon et al. [40] have presented an information theoretic approach to word clustering for text classification. First, they derived a global objective function to capture the decrease in mutual information due to clustering and then presented a divisive algorithm that directly minimizes this objective function, converging to a local minimum. Huang et al. [66] have proposed a classification algorithm based on local cluster centers for data sets with a few labeled training data. The method can reduce the interference of labeled data whose labels are missing, including those provided by both domain experts and co-training paradigm algorithms. Chen et al. [28] have developed a simple but effective classifier for text categorization by using class dependent projection based method. Dhurandhar et al. [41] have designed a probabilistic characterization for the moments of generalization error of the kNN rule applied to categorical data. They have provided a principled way of studying the non-asymptotic behavior of kNN rule and in particular, they have derived the exact closed form expressions for the moments of the generalization error for kNN rule assuming that all the attributes present are categorical in nature. In a very recent study, Zhang et al. have presented a novel projected prototype based classifier for text categorization, in which a document category is represented by a set of prototypes, each assembling a representative for the documents in a subclass and its corresponding term subspace [134]. Blei et al. proposed a generative probabilistic model, Latent Dirichlet Allocation (LDA), for collections of large discrete data such as text corpora [18]. It is a three level hierarchical bayesian model, where each document of a text collection is modeled as a finite mixture over an underlying set of topics. Each topic is modeled as an infinite mixture over an underlying set of topic probabilities and the topic probabilities provide an explicit representation of a document [18]. Qiu et al. have developed a fast cluster computing framework, SPARK, for large scale data processing, using parallel collapsed Gibbs sampling method on LDA model [105]. Yu et al. have identified the unique challenges for product search on e-commerce sites (e.g., eBay) and proposed a LDA based diversified retrieval approach to address these challenges [132]. They have also proposed a Bernoulli LDA model, which is suitable for modeling short item titles without duplicated terms in the e-commerce data.

1.4 Evaluation Criteria

1.4.1 Criteria for Clustering

If the documents within a cluster are similar to each other and dissimilar to the documents in the other clusters then the clustering algorithm is considered to be performing well. The data sets under consideration have labeled documents. Hence quality measures based on labeled data are used here for comparison. These measures are *f-measure* and *normalized mutual information*.

F-measure and normalized mutual information are very popular and are used by a number of researchers [116, 123] to measure the quality of a cluster using the information of the actual categories of the corpus. Let us assume that R is the set of categories and S is the set of clusters. Consider there are I number of categories in R and J number of clusters in S. A total of N number of documents are there in the corpus, i.e., both R and S individually contains N documents. Let r_i is the number of documents belonging to category i, s_j is the number of documents belonging to cluster j and s_{ij} is the number of documents belonging to both category i and cluster j, for all $i = 1, 2, ..., I$ and $j = 1, 2, ..., J$.

- **F-measure** determines the recall and precision value of each cluster with a particular category [116]. Let, for a query the set of relevant documents be from category i and the set of retrieved documents be from cluster j. Then recall, precision and f-measure are given as :

$$recall_{ij} = \frac{s_{ij}}{r_i}, \quad \forall\, i, j \quad \text{and} \quad precision_{ij} = \frac{s_{ij}}{s_j}, \quad \forall\ i, j$$
$$F_{ij} = \frac{2 \times recall_{ij} \times precision_{ij}}{recall_{ij} + precision_{ij}}, \quad \forall\, i, j$$

  If there is no common instance between a category and a cluster (i.e., $s_{ij} = 0$) then $F_{ij} = 0$. The value of F_{ij} is maximum when $precision_{ij} = recall_{ij}$ and $s_{ij} \neq 0$ for a category i and cluster j. Thus the value of F_{ij} lies between 0 and 1. The best f-measure among all the clusters is selected as the f-measure for the query of a particular category is $F_i = \max_{j \in [0,J]} F_{ij}$, $\forall i$. The f-measure of all the clusters is weighted average of the sum of the f-measures of each category, $F = \sum_{i=1}^{I} \frac{r_i}{N} F_i$. We would like to maximize f-measure to achieve good quality clusters.

- **Normalized Mutual Information** (NMI) is the combination of mutual information and entropies of R and S. It was described by Strehl et al. [118]. Mutual information is a symmetric measure to quantify the statistical information shared between two distributions which provides a sound indication of the shared information between a set of categories and a set of clusters. Let $MI(R, S)$ denotes the mutual information between R and S and $E(R)$ and $E(S)$ be the entropy of R and S respectively. $MI(R, S)$ and $E(R)$ can be defined as

$$MI(R,S) = \sum_{i=1}^{I}\sum_{j=1}^{J} \frac{s_{ij}}{N} \log\left(\frac{Ns_{ij}}{r_i s_j}\right)$$

$$E(R) = -\sum_{i=1}^{I} \frac{r_i}{N} \log\left(\frac{r_i}{N}\right)$$

  There is no upper bound for $MI(R, S)$, so for easier interpretation and comparisons a normalized mutual information that ranges from 0 to 1 is desirable [118]. NMI is defined as follows:

$$NMI(R,S) = \frac{MI(R,S)}{\sqrt{E(R)E(S)}}$$

  Note that at least one document must be there in each category and in each cluster, i.e., $r_i > 0\ \forall, i \in I$ and $s_j > 0\ \forall, j \in J$. If there is no common documents between a category i and a cluster j (i.e., $s_{ij} = 0$) then the convention $0\log(0) = 0$ is used. Note that $NMI(S, S) = 1$, and thus normalized mutual information ranges from 0 to 1. It is desirable to maximize the NMI value.

1.4.2 Criteria for Categorization

- **Accuracy** of text categorization can be defined as the number of samples correctly chosen by the classifier from the test set divided by the number of total samples in the test set. The accuracy can be given in the following way.

$$\text{accuracy} = \frac{\text{number of correctly chosen documents}}{\text{total number of documents}}$$

- **Precision, recall and f-measure** determines the effectiveness of a classifier in category assignment, aggregated over all categories or over all the docu-

ments. The precision and recall for a set of m categories can be computed as follow :

$$\text{precision} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FP_i}$$

$$\text{recall} = \frac{1}{m}\sum_{i=1}^{m}\frac{TP_i}{TP_i + FN_i}$$

Here TP_i counts the number of documents correctly assigned to the i^{th} category, FP_i counts the number of documents incorrectly rejected from the i^{th} category, FN_i counts the number of documents incorrectly assigned to the i^{th} category, TN_i counts the number of documents correctly rejected from the i^{th} category.

The f-measure combines recall and precision with an equal weight in the following form:

$$\text{f-measure} = \frac{2 \times \text{recall} \times \text{precision}}{\text{recall} + \text{precision}}$$

The closer the values of precision and recall, the higher is the f-measure. F-measure becomes 1 when the values of precision and recall are 1 and it becomes 0 when precision is 0, or recall is 0, or both are 0. Thus f-measure lies between 0 and 1. A high f-measure value is desirable for good categorization.

1.5 Description of Text Data Sets

A number of text data sets are used in various experiments in the chapters of the thesis. A brief description of each of the data sets used is given below.

- **20-newsgroups** data[1] is a collection of news articles collected from 20 different sources. Each news source constitutes a different category. In this data set, articles with multiple topics are cross posted to multiple newsgroups i.e., there are overlaps between several categories. There are about 18,000 documents in the corpus. The data set is named as *20ns* in the thesis.

[1] http://www.cs.cmu.edu/∼TextLearning/datasets.html

Table 1.1: Overview of Text Data Sets

Data Set	No. of Documents	No. of Terms	No. of Categories
20ns	18000	35218	20
fbis	2463	2000	17
la1	3204	31472	6
la2	3075	31472	6
oh10	1050	3238	10
oh15	913	3100	10
rcv1	2017	12906	30
rcv2	2017	12912	30
rcv3	2017	12820	30
rcv4	2016	13181	30
tr31	927	10128	7
tr41	878	7454	10
tr45	690	8261	10
wap	1560	8460	20

- The data set **fbis** is developed[2] and used in the article by Karypis et al. [72]. The data set was originally collected from the Foreign Broadcast Information Service data of TREC-5 [120].

- **la1** and **la2** were created from the Los Angeles Times data of TREC-5 [120]. The categories of la1 and la2 were generated according to the name of the newspaper sections where these articles appeared, such as Entertainment, Financial, Foreign, Metro, National, and Sports. The documents that have single label were selected for these data sets, i.e., there are no overlaps between the documents of different categories. The data sets are developed in the Karypis lab[3] [72].

- **oh10** and **oh15** are the versions of *Ohsumed* data set. *Ohsumed* test collection[4] is a set of 348,566 references from MEDLINE, the on-line medical information database, consisting of titles and/or abstracts from 270 medical journals over a five-year period (1987-1991). The available fields are title, abstract, MeSH indexing terms, author, source, and publication type. Different subsets of the original categories of Ohsumed have been taken to construct oh10 and oh15.

[2]http://www-users.cs.umn.edu/∼han/data/tmdata.tar.gz
[3]http://glaros.dtc.umn.edu/gkhome/index.php
[4]http://disi.unitn.it/moschitti/corpora.htm

The information on these data sets are available in the article by Karypis et al. [72].

- **Reuters-21578** is a collection of documents that appeared on Reuters newswire in 1987. The documents were originally assembled and indexed with categories by Carnegie Group Inc., and Reuters, Ltd. The corpus contains 21578 documents in 135 categories [130]. The ModApte version used in [22] are considered here, in which there are 30 categories and 8067 documents. Basu et al. [11] have divided this corpus into four groups and with the name as *rcv1, rcv2, rcv3* and *rcv4*, which have shown very good performance on document clustering. The detailed description of the four subsets of the Reuters data are given in Table 1.1.

- **tr31, tr41** and **tr45** are derived from TREC-5, TREC-6, and TREC-7 data collections [120]. The categories of the tr31, tr41 and tr45 data sets were generated from the relevance judgment provided in these collections. The data sets are used in the articles by Karypis et. al [116].

- Data set **wap** is originated from the WebACE project [19]. These data sets are created in the karypis lab[5] and first used in the article by Han. et al. [72].

An overview of these data sets is given in Table 1.1. All the data sets are used here by removing stopwords using a standard English stopword list[6] and the terms are stemmed using Porter's suffix-stripping algorithm [104].

1.6 Thesis Contributions

The thesis aims to introduce new supervised and unsupervised methodologies for effective mining of text data. The three main contributions of the thesis are

a) A new similarity measure and a new hierarchical document clustering technique using this similarity measure.

b) A new nearest neighbor classification technique for text categorization.

c) A new term selection scheme for dimensionality reduction of text data.

[5]http://glaros.dtc.umn.edu/gkhome/index.php
[6]http://www.textfixer.com/resources/common-english-words.txt

Some extensions of these three works are presented in the other chapters. A brief description of these works is given below.

1.6.1 A Hierarchical Document Clustering Technique

One of the main challenges of any document clustering algorithm is the selection of a good similarity measure. In this work a new similarity measure between documents has been introduced to improve the accuracy of measuring the similarity between documents and the similarity measure is named as *extensive similarity*. A new hierarchical document clustering technique is designed by using the proposed extensive similarity. Intuitively, extensive similarity determines the similarity between two documents by finding their content similarity as well as their distances with every other document in the corpus. The significant characteristic of extensive similarity is that it can identify two dissimilar documents by using a similarity threshold.

The new clustering method is named as Clustering Using Extensive Similarity (CUES). CUES need not require the number of clusters prior to implementation, like the other document clustering algorithms. The number of clusters is automatically determined by CUES from the data. It is hierarchical in nature, but no stopping criteria is needed to terminate the algorithm. The algorithm could be stopped by varying the similarity threshold of extensive similarity. A data independent histogram thresholding based method is proposed for selecting the value of the similarity threshold. The experimental results show that CUES performs significantly better than several other document clustering techniques using the threshold values selected by the proposed histogram thresholding based technique. The material of this work can be found in article [11].

1.6.2 Document Clustering by A Hierarchical Approach to k-means Clustering

Another document clustering technique is introduced in this work, which is a combination of a hierarchical and k-means clustering technique. The hierarchical part creates some clusters by the extensive similarity between documents and by giving a bound on the distance between clusters. This bound is determined by a predefined threshold. A technique is provided to determine the threshold. The hierarchical clusters are named as *baseline clusters*. Rest of the documents which remain as sin-

gleton clusters are grouped to anyone of the existing clusters by k-means clustering technique. The centroids of the baseline clusters are used as the initial seeds for the k-means clustering technique. Thus the proposed method reduces the error due to random seed selection. On the other hand it is not as computationally expensive as the other hierarchical clustering algorithms like CUES or single-link clustering techniques. It will be observed from the experiments, later in chapter 3 that the proposed technique takes less time than several partitional clustering algorithms (e.g., buckshot, k-means). The contents of this work are taken from article [14].

1.6.3 Tweak on k-Nearest Neighbor Decision Rule for Text Categorization

A Tweak on the kNN decision rule (TkNN) is presented in this chapter, which restricts the majority voting of kNN decision rule by a predefined positive integer threshold to assign a document to a predefined category. In kNN decision rule, a new test document is put in a particular category that has maximum number of representatives among the k nearest neighbors. There are no particular bounds on the discrimination criterion of majority voting. Sometimes, a test document is assigned to an arbitrarily chosen competing category when a tie occurs between the competing categories. The decision about the category label of a document by kNN rule may not be accurate if the difference between the number of members (among the neighbors) of the competing categories is one or they have same number of members. Intuitively, if the difference between the number of members of the competing categories is more than one then it may enhance the confidence of the majority voting. Practically it is true for any majority voting process that the more the margin of win is, the more the confidence on the choice of decision.

The proposed decision rule does not categorize a document when a decision in not so certain. Thus the main objective of TkNN is to enhance the certainty of the decision. The experimental analysis shows that TkNN decision rule outperforms kNN and several other classifiers. The experiments also show that the processing time of the proposed method is less than the processing time of kNN decision rule and the other methods. The discussions on the proposed decision rule can be found in articles [16] and [15].

1.6.4 An Extensive Similarity based Supervised Decision Rule for Text Categorization

In this chapter, another decision rule for text categorization is proposed by combining the idea of TkNN and the extensive similarity between the test document and the training documents [11]. The extensive similarity measure is used in this chapter after a simple modification to use it for text categorization. The experimental results show that the extensive similarity has boost the performance of TkNN decision rule. The material of this study can be found in article [12].

1.6.5 A Supervised Term Selection Technique for Text Categorization

Text categorization is a challenging task due to the high dimensionality of the feature space and the sparsity of data. Each term in the vocabulary of a corpus is considered as a feature and thus the feature space becomes large. An effective term selection method is needed to improve the quality of text categorization by removing the redundant and unimportant terms. A supervised term selection technique is developed in this chapter which gives high priority to the terms that occur highly in each category irrespective of their occurrence in the entire corpus. The proposed evaluation function, Term ReLatedness (TRL) derives a similarity score between a term and a category and then finds the overall score among all the categories. All the terms are then ranked according to these scores and a selected number of top terms forms the ultimate feature subset for text categorization. It has been described that TRL may easily be applied to any high dimensional real life data sets as its computational complexity is reasonable. The description of this work can be found in article [13].

1.7 Organization

The rest of the thesis is organized in the following way.

- **Chapter 2** describes the proposed extensive similarity measure and its various properties. The proposed document clustering technique CUES is also described in this chapter. An experimental analysis is provided where CUES is compared with various other clustering algorithms on some well known text

data sets. The computational times of CUES and other algorithms are also reported.

- **Chapter 3** presents another document clustering technique, which is a combination of hierarchical and k-means clustering technique. The extensive similarity is used to design the hierarchical part. A discussion is provided on the impact of extensive similarity on the proposed document clustering technique. The document clustering technique is compared with CUES and various other clustering algorithms. A discussion is provided about the computational time of the proposed algorithm and the other methods.

- In **chapter 4**, a tweak on kNN decision rule for text categorization is described. An experimental evaluation is provided where the proposed decision rule for text categorization is compared with several other text classifiers using a set of text data sets discussed in section 1.1. A discussion has been provided about the processing time of the proposed decision rule and other methods.

- **Chapter 5** describes an extensive similarity based decision rule. The empirical study shows the performance of the proposed technique and other classifiers for text categorization. An analysis on the computational time of the proposed decision rule vis-a-vis other methods is also discussed.

- **Chapter 6** presents a novel evaluation function for term selection in text categorization. The properties of the proposed function are discussed. An analysis is given on the space and time complexity of the proposed scheme. The experimental evaluation shows the comparison of the performances of the proposed scheme and several other feature selection techniques. The exact processing times of the proposed technique and the other methods are given on various text data sets.

- **Chapter 7** consists of conclusions, discussion and scope for further work.

Chapter 2

CUES: A New Hierarchical Approach for Document Clustering

2.1 Introduction

In this chapter the problem of document clustering is addressed. A new similarity measure between documents is introduced and a new hierarchical document clustering technique is proposed here. It has been discussed in section 1.3.2 that the document clustering algorithms aim to reduce the within cluster distances such that they put the similar documents in the same group and segregate the dissimilar documents. Thus the performance of a clustering algorithm is mainly dependent on the similarity measure. Several similarity measures have been used to perform document clustering [64]. The document clustering algorithms determine the content similarity of a pair of documents for putting them into the same cluster. The standard way of finding the content similarity between two documents is to compute the cosine similarity between the document vectors [89]. An important property of cosine similarity is its independence of document length and so that it has become popular in finding similarity between documents [64]. Cosine similarity actually checks the number of common terms present in the documents. If two documents contain many common terms then the documents are very likely to be similar. The difficulty is that there is no clear explanation on how many common terms can identify two documents as similar. The text data sets are high dimensional data set and most

of the terms do not occur in each document, i.e., the term-document matrices are generally sparse. In this situation two documents with very low similarity value may be grouped in the same cluster. Consider a, b, c be the three centroids of three individual clusters and the task is to find the proper group of a document z. Assume that k-means clustering algorithm is performed and z has low content similarity with each of $a, b,$ and c. In principle z is to be assigned to one of a, b, c with which it has least dissimilarity, but in practice z should not be assigned to a cluster based on these low similarity values. The same scenario may be observed in all the other clustering techniques, i.e., two documents having very low content similarity may be grouped in the same cluster. Hence the issue is to find the content similarity in such a way so that it can restrict the low similarity values. The actual content similarity between two documents may not be found properly by checking the individual terms of the documents. Intuitively, if two documents are content-wise similar then they should have similar type of relations with most of the other documents, i.e., if two documents x and y have similar content and if x is similar to any other document z then y must be similar or somehow related to z. This important characteristic is not observed in cosine similarity measure. A similarity measure is proposed maintaining this property.

The proposed similarity measure is named as *extensive similarity* and the aim in this regard is to reduce the constraints of the cosine similarity for improving the performance of clustering. Extensive similarity initially restricts the low similarity values of the cosine similarity by a predefined threshold and then finds the similarity between documents by extensively checking all the documents in the corpus. The other significant property of extensive similarity is that it can identify two dissimilar documents by the threshold. Intuitively two documents with high extensive similarity between them should be in the same cluster. A new distance function is introduced to find the distance between two clusters by this property of extensive similarity. The distance between two clusters becomes negative if the extensive similarity between every two documents (taking one from each cluster) is negative. A hierarchical document clustering technique is proposed in this chapter utilizing the proposed cluster distance measure. The new document clustering technique is named as Clustering Using Extensive Similarity (CUES). CUES initially treats every document as a single cluster. The algorithm then merges two clusters which have minimum cluster distance among all the clusters and again finds two minimum distant clusters and merges them and so on. CUES terminates when the

distance between every two clusters is negative. It may be noted that two clusters with negative cluster distance is never merged by CUES. Thus CUES determines the number of clusters automatically, which is the most significant characteristic of the proposed technique. The performance of CUES is compared with several partitional and hierarchical clustering algorithms and two types of spectral clustering methods using various well known text data sets in the experimental evaluation. The experimental analysis shows that CUES performs significantly better than the other methods. The main material in this chapter is taken from the article [11].

The chapter is organized as follows - section 2.2 explains the proposed extensive similarity measure. The proposed document clustering technique is presented in section 2.3. Section 2.4 describes the experimental results on several text data sets. The conclusions of the proposed work are presented in chapter 2.5.

2.2 Extensive Similarity

A similarity measure is introduced to find the similarity between two documents, which is named as *extensive similarity*. The similarity measure extensively checks all the documents in the corpus to determine the similarity. The extensive similarity between two documents is determined depending on their distances with every other document in the corpus. Intuitively, two documents are exactly similar, if they have sufficient content similarity and they have almost same distances with every other document in the corpus (i.e., either both are similar or both are dissimilar to all the other documents). The content similarity is defined as a binary valued distance function. If two documents have sufficient terms in common then the content similarity is 0, i.e., the distance is minimum, otherwise the distance is 1, i.e., they have no content similarity. The content similarity between twodocuments d_i and d_j, $\forall i, j$ are determined by putting a threshold $\theta \in (0, 1)$ on their cosine similarity as follows:

$$dis(d_i, d_j) = \begin{cases} 1 & \text{if } cos(\vec{d_i}, \vec{d_j}) \leq \theta \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

where $\vec{d_i}$ and $\vec{d_j}$ are the corresponding vectors of documents d_i and d_j respectively. Here θ is a threshold value on the cosine similarity and it is used to restrict the low similarity values. A data dependent method for estimating the value of θ is discussed later. If the cosine similarity between two documents is 1 then it can be

strictly said that the documents are dissimilar. On the other hand, if the distance is 0, then they have sufficient content similarity and the documents are somehow related to each other. Let us assume that d_i and d_j have cosine similarity 0.52 and d_j and d_0 (another document) have cosine similarity 0.44 and $\theta = 0.1$. Hence both $dis(d_i, d_j) = 0$ and $dis(d_j, d_0) = 0$ and the task is to distinguish these two distances of same value. The extensive similarity is thus designed to find the grade of similarity of the pair of documents which are similar content-wise. If $dis(d_i, d_j) = 0$ then extensive similarity finds the individual content similarities of d_i and d_j with every other document, and assigns a score (l_{ij}) to denote the extensive similarity between the documents as below.

$$l_{ij} = \sum_{k=1}^{N} |dis(d_i, d_k) - dis(d_j, d_k)| \tag{2.2}$$

Thus the extensive similarity (ES) between documents d_i and d_j, $\forall i, j$ is defined as

$$ES(d_i, d_j) = \begin{cases} N - l_{ij} & \text{if } dis(d_i, d_j) = 0 \\ -1 & \text{otherwise} \end{cases} \tag{2.3}$$

Two documents d_i, d_j have maximum extensive similarity N, if the distance between them is zero, and distance between d_i and d_k is same as the distance between d_j and d_k for every k. In general, if the above said distances differ for l_{ij} times then the extensive similarity is $N - l_{ij}$. Unlike other similarity measures, ES takes into account the distances of the said two documents d_i, d_j with respect to all the other documents in the corpus when measuring the distance between them. l_{ij} indicates the number of documents with which the similarity of d_i is not the same as the similarity of d_j. As the l_{ij} value increases, the similarity between the documents d_i and d_j decreases. If $l_{ij} = 0$ then d_i and d_j are exactly similar. Actually l_{ij} denotes a grade of dissimilarity and it indicates that d_i and d_j have different distances with l_{ij} number of documents. The extensive similarity is used to define the distance between two clusters in the first stage of the proposed document clustering method.

2.2.1 Properties of Extensive Similarity

Extensive similarity has some interesting properties which are stated below.

- ES is symmetric. For every pair of documents d_i and d_j, we have $ES(d_i, d_j) = ES(d_j, d_i)$.

- If $d_i = d_j$ then $ES(d_i, d_j) = 0$. However $ES(d_i, d_j) = 0 \Rightarrow dis(d_i, d_j) = 0$ and $\sum_{k=1}^{N} |dis(d_i, d_k) - dis(d_j, d_k)| = 0$, but $dis(d_i, d_j) = 0 \not\Rightarrow d_i = d_j$. Hence ES is not a metric.

- Let d_i, d_j, d_0 be any three documents and $ES(d_i, d_j) \geq 0, \quad \forall\, i, j$. Then from equation 2.3 we have

$$\begin{aligned} ES(d_i, d_j) + ES(d_j, d_0) - ES(d_i, d_0) &= \sum_{k=1}^{N} \Big(|dis(d_i, d_k) - dis(d_j, d_k)| + \\ &\quad |dis(d_j, d_k) - dis(d_0, d_k)| - \\ &\quad |dis(d_i, d_k) - dis(d_0, d_k)| \Big) \\ &\geq 0 \end{aligned}$$

  since the operator *modulus* is a metric. Thus ES satisfies the triangular inequality, i.e.,

$$ES(d_i, d_j) + ES(d_j, d_0) \geq ES(d_i, d_0), \text{ if } ES(d_i, d_j) \geq 0, \quad \forall\ i, j$$

  Therefore the triangular inequality holds good for non negative ES values.

- Note that $ES(d_i, d_j) \ngeq 0$ for any two documents d_i and d_j. However the only negative value of ES is -1 and it has been used as a symbol to denote the complete dissimilarity between two documents.

2.2.2 Remarks

Some typical cases regarding the definition of the function *dis* (equation 2.1) are discussed below.

- If the cosine value between two documents is less than θ then it can be strictly said that the documents are dissimilar as they are not sharing a minimum number of common terms.

- If the cosine value between two documents d_i and d_j is very high (e.g., 0.85), then they share sufficiently many common terms and hence the documents are

very similar, and d_i, d_j are likely to have similar distances with most of the other documents in the corpus. As a result, the value of ES becomes very high, which indicates that d_i and d_j are similar.

- Now consider the situation where $dis(d_i, d_j) = 0$, but the number of common terms between d_i and d_j is neither very high nor very low, i.e., $\theta \leq cos(d_i, d_j) \leq \theta'$, where θ' is another threshold on cosine similarity. ES is very useful in this particular situation which occurs frequently. It checks the distances of d_i and d_j with the other documents in the data set and assigns a grade (l_{ij}) to the similarity between the documents. The l values of all pair of documents are used to group documents into clusters, which is introduced in the proposed document clustering algorithm.

2.3 Proposed Document Clustering Technique

All the document clustering algorithms discussed in section 1.3.2 are mainly based on similarity between two documents. Two documents belong to the same cluster, if they are dissimilar to the documents in other clusters and similar to the documents in that cluster. The distance between two documents is dependent on the number of common terms present in the documents. But there are no crisp bounds on the content similarity. Most of the traditional clustering methods identify two documents as similar if their similarity is more than the similarity of other pairs. This may lead, sometimes, to a case where the similarity between two documents is a low value (i.e., the content similarity is very low), but they are considered as similar to be grouped in the same cluster. In extensive similarity, the distance between two documents is determined after extensively checking their distances with every other document in the corpus.

2.3.1 Cluster Distance

A distance function has been introduced by using extensive similarity to find the distance between two clusters say, C_x and C_y. The cluster distance between two clusters C_x and C_y is the maximum of the set of non negative ES values between a pair of documents, one of which is from C_x and the other is from C_y. The cluster distance is -1, if there are no two documents which have non negative ES value, i.e.,

no similar documents are present in C_x and C_y. Let us assume,

$$SES(C_x, C_y) = \{ES(d_i, d_j) : ES(d_i, d_j) \geq 0,\ \forall d_i \in C_x \text{ and } \forall d_j \in C_y\}$$

be the *Set of Extensive Similarities*(SES) between the documents of C_x and C_y. SES becomes null, if there exists two elements in SES, one from C_x, and the other from C_y such that their ES value is -1. Now the distance between C_x and C_y is defined as

$$cluster\_dis(C_x, C_y) = \begin{cases} -1 & \text{if } SES(C_x, C_y) = \emptyset \\ \\ N - \max(SES(C_x, C_y)) & \text{otherwise} \end{cases} \tag{2.4}$$

Let us consider an example of two clusters C_x and C_y, where C_x contains four documents and C_y contains three documents . Thus totally 12 ES values are there between C_x and C_y. The cluster distance between C_x and C_y is -1, if at least one of these 12 values is negative. The intuition behind negative cluster distance is that the documents of C_x and C_y either share a very few number of terms, or no term is common between them, i.e., they have a very low content similarity. The essence of cluster distance lies in the fact that it would never merge two sets of dissimilar documents to the same cluster. This phenomenon of cluster distance makes the clustering task easier by segregating the dissimilar documents. It may be observed from the experiments in next section that some clusters remain singleton clusters when CUES terminates. Basically these singleton clusters (rather documents) have negative cluster distance with all other clusters.

2.3.2 Clustering Procedure

The algorithm initially assumes each document as a single cluster. A similarity matrix is created whose ij^{th} entry is the $ES(d_i, d_j)$ value where d_i and d_j are i^{th} and j^{th} documents respectively. It is a square matrix and has N rows and N columns for N number of clusters. Each row or column represents a cluster. Initially each document is taken as a cluster. CUES starts with N individual clusters. Note that at the end of the algorithm, the number of clusters remaining is not necessarily 1, since some clusters may not be merged with any other clusters. Sometimes, some of the singleton clusters may remain singleton clusters when the algorithm is terminated.

Algorithm 1 Clustering using Extensive Similarity (CUES)

Input: a) A set of N clusters, $C = \{C_1, C_2, ..., C_N\}$ and $noc = |C|$, number of clusters.
b) $C_i = \{d_i\},\ \forall i \in N$, where d_i is the i^{th} document of the data set.
c) A similarity matrix $Sim[i][j] = cluster\_dis(C_i, C_j),\ \forall i, j \in [1, N]$.

Steps of the Algorithm:

1: $X \leftarrow 0,\ \ Y \leftarrow 0$
2: **while** $noc > 1$ and $X \geq 0$ and $Y \geq 0$ **do**
3: $\quad min\_dist \leftarrow N$
4: $\quad X \leftarrow -1,\ \ Y \leftarrow -1$
5: $\quad$ **for** $i = 1$ to $noc - 1$ **do**
6: $\quad\quad$ **for** $j = i + 1$ to noc **do**
7: $\quad\quad\quad$ **if** $min\_dist \geq cluster\_dis(C_i, C_j)$ and $cluster\_dis(C_i, C_j) \geq 0$ **then**
8: $\quad\quad\quad\quad min\_dist \leftarrow cluster\_dis(C_i, C_j)$
9: $\quad\quad\quad\quad X \leftarrow i,\ \ Y \leftarrow j$
10: $\quad\quad\quad$ **end if**
11: $\quad\quad$ **end for**
12: $\quad$ **end for**
13: $\quad$ **if** $X \geq 0$ and $Y \geq 0$ **then**
14: $\quad\quad C_X \leftarrow C_X \cup C_Y$
15: $\quad\quad Sim \leftarrow \text{merge}(Sim, X, Y)$
16: $\quad\quad noc \leftarrow noc - 1$
17: $\quad$ **end if**
18: **end while**
19: return C

At the first step CUES merges those two clusters whose cluster distance is minimum and the similarity matrix is updated accordingly. Then, again those two clusters whose cluster distance is minimum are merged, and so on. This process is continued till no more merges take place, i.e., till there exist no two clusters with non negative cluster distance. In other terms, the algorithm is terminated when negative cluster distance is observed between every pair of clusters. Algorithm 1 describes the steps of the proposed document clustering method in detail. The merging procedure stated in step 15 of Algorithm 1 merges two rows say i and j and the corresponding columns of the similarity matrix by following a convention regarding numbering. It merges two rows into one, the resultant row is numbered as minimum of i, j, and the other row is removed. Similar numbering follows for columns too. Then the index structure of similarity matrix is updated accordingly.

It is to be noted that the algorithm never merges two clusters if the distance between them is -1. They remains separate till the end of the algorithm. All traditional document clustering algorithms except graph clustering algorithms [113] can not identify two dissimilar clusters. They always generate a grade of similarity between the clusters and eventually merge those clusters. But the proposed cluster distance can identify two dissimilar clusters and never merge them. By this property, CUES can automatically identify the natural clusters in the data set and does not require a prior information of number of clusters for implementation.

In the section on experimental results, it will be observed that CUES produces some singleton clusters. These singleton clusters have negative cluster distance with the other clusters and so that they remain single and could be treated as outliers of the data set. If the θ value is very high then the documents within a particular cluster must have high extensive similarity, on the other hand it may produce huge number of singleton clusters. In such cases the quality of the clusters including these singleton clusters would need to be evaluated. Consequently, the value of θ can be decreased to reduce the total number of singleton clusters. Cluster distance is inherited from extensive similarity between documents. The extensive similarity not only determines the similarity between documents but also describes the underlying structure of the corpus. Ideally within a cluster the ES values between each pair of documents are close to each other and the extensive similarity between every pair of clusters is high at the end of the clustering. Algorithm 1 normally continues if there remain some clusters with a non-negative cluster distance, otherwise it stops there. Note that no stopping criterion is needed for CUES. The flowchart of the proposed clustering technique is shown in figure 2.1.

2.3.3 Discussion

The structure of the proposed document clustering algorithm is quite similar to single-link hierarchical clustering (SLHC) technique. The main difference between SLHC and CUES is the negative cluster distance between two clusters. Two clusters with insignificantly low similarity may be merged at any hierarchy of SLHC, but CUES never merge them if they have negative cluster distance. Secondly, the SLHC technique needs to know the stopping criterion externally, but CUES is automatically stopped if there are no two clusters with non negative cluster distance.

The single-link algorithm, by contrast, suffers from chaining effect [115]. It has

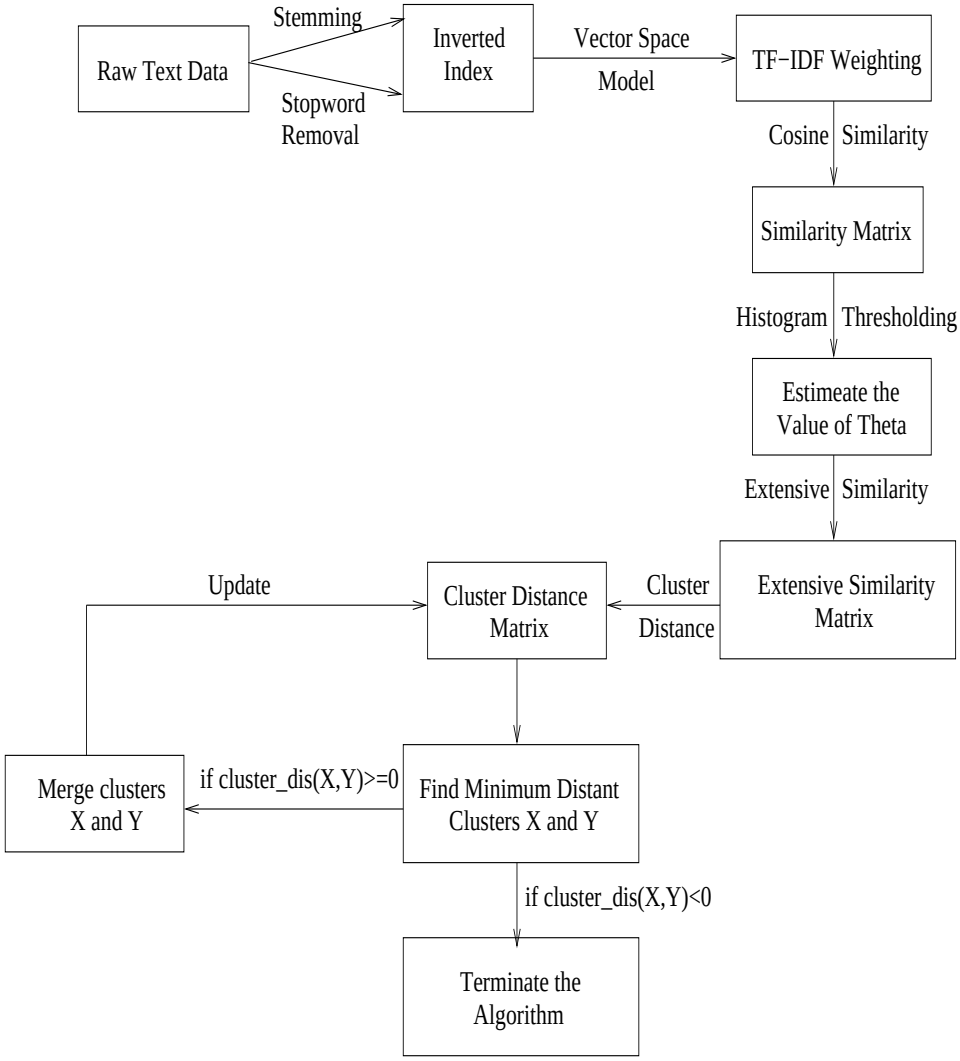

Figure 2.1: Clustering Using Extensive Similarity between Documents

a tendency to produce clusters that are straggly or elongated. The clusters that are separated by a bridge (thin line) of noisy patterns may be merged by single-link clustering. CUES is designed like single-link algorithm, but it never suffers from chaining effect. In Single-link algorithm, the similarity between two clusters is taken as the similarity between two most similar documents of the two clusters. This sometimes gives raise to merger of two clusters where two documents in two clusters possess very small similarity values. CUES merges two clusters where the similarities with respect to all the documents are taken into consideration, and consequently, the low similarities are also considered. This results in merging of two clusters when every similarity value exceeds a threshold. Thus, chaining effect is not present in the resultant clusters since similarities between every pair of documents are taken into consideration for calculating the cluster distance.

It may be observed that whenever two clusters are merged, the similarity between any two documents in the merged cluster will at least be equal to θ. This interesting property of CUES can be observed from the following theorem.

Theorem 1. *Let C_x and C_y be two resulting clusters of the proposed scheme then*
a) $C_x \cap C_y = \emptyset$, i.e., if $d_i \in C_x$ then $d_i \notin C_y$
b) $d_i, d_j \in C_x \Rightarrow dis(d_i, d_j) = 0$
c) $\exists\ d_i \in C_x$ and $d_j \in C_y$ such that $dis(d_i, d_j) = 1$

Proof. **1.a)** It is proved by the method of contradiction. Let us consider that $C_x \cap C_y \neq \emptyset$, i.e., $\exists\ d_i$ such that $d_i \in C_x$ and $d_i \in C_y$. Note that $dis(d_i, d_i) = 0$ and $ES(d_i, d_i) = N$, N is the total number of documents. As a result, after some iterations C_x, C_y would be merged, which is contradicting the assumption. Hence $C_x \cap C_y = \emptyset$.

1.b) Let us assume that $d_i, d_j \in C_x$ and $dis(d_i, d_j) = 1$. Initially we have two singleton clusters $\{d_i\}$ and $\{d_j\}$. After some iterations we would have clusters C_{11} and C_{12} in such a way that
i) $d_i \in C_{11}$ and $d_i \notin C_{12}$, ii) $d_j \in C_{12}$ and $d_j \notin C_{11}$ and iii) $C_{11}, C_{12} \subseteq C_x$
As a result C_{11}, C_{12} are merged according to the proposed criterion. Now $dis(d_i, d_j) = 1 \Rightarrow ES(d_i, d_j) = -1$ and so that $cluster\_dis(C_{11}, C_{12}) = -1$. Therefore C_{11} and C_{12} can not be merged, which is a contradiction and thus $dis(d_i, d_j) \neq 1$. Hence $dis(d_i, d_j) = 0$.

1.c) This is also proved here by the method of contradiction. Let us assume that

the statement 1.c) is not true. That means there exists no $d_i \in C_x$ and no $d_j \in C_y$ such that $dis(d_i, d_j) = 1$, i.e., $\forall\ d_i \in C_x$ and $\forall\ d_j \in C_y,\ dis(d_i, d_j) = 0$. Therefore $ES(d_i, d_j) \geq 0,\ \forall\ d_i \in C_x$ and $\forall\ d_j \in C_y$. As a result $cluster\_dis(d_i, d_j) \geq 0$, and C_x, C_y are merged, contradicting the assumption. Hence $\exists\ d_i \in C_x$ and $d_j \in C_y$ such that $dis(d_i, d_j) = 1$.

□

The quality of the resultant clusters produced by CUES may be observed from the above theorem.

2.3.4 A Method for Estimation of θ

There are several types of document collections available in real life. The similarities or dissimilarities between documents present in one data set may not be same as the similarities or dissimilarities of the other data sets, since the characteristics of the data sets are different. Additionally, one may view the clusters present in a data set (or in different data sets) under different scales, and different scales produce different partitions. Similarities corresponding to one scale in one data set may not be same as the similarities corresponding to the same scale in a different data set. This has been the reason to make the threshold on similarities data dependent. In fact, we feel that a fixed threshold on similarities can not give satisfactory results on several data sets.

There are several methods available in literature for finding a threshold for a two-class (one class corresponds to similar points, and the other corresponds to dissimilar points) classification problem. A popular method for such classification is histogram thresholding [57].

Let, for a given data set, the number of distinct similarity values be p, and let the similarity values be $s_0, s_1, ..., s_{p-1}$. Without loss of generality, let us assume that (a) $s_i < s_j$, if $i < j$ and (b) $(s_{i+1} - s_i) = (s_1 - s_0),\ \forall i = 1, 2, ..., (p-2)$. Let $g(s_i)$ denote the number of occurrences of $s_i,\ \forall i = 0, 1, ..., (p-1)$. Our aim is to find a threshold θ on the similarity values so that a similarity value $s < \theta$ implies the corresponding documents are practically dissimilar, otherwise they are similar. The aim is to make the choice of threshold to be data dependent. The basic steps of the histogram thresholding technique are as follow:

- Obtain the histogram corresponding to the given problem.

- Reduce the ambiguity in histogram. Usually this step is carried out using a window. One of the earliest such techniques is the moving average technique in time series analysis [20], which is used to reduce the local variations in a histogram. It is convolved with the histogram resulting in a less ambiguous histogram. The weighted moving averages have been used using window length 5 of the $g(s_i)$ values as,

$$f(s_i) = \frac{g(s_i)}{\sum_{j=0}^{p-1} g(s_j)} \times \frac{g(s_{i-2}) + g(s_{i-1}) + g(s_i) + g(s_{i+1}) + g(s_{i+2})}{5},$$

$$\forall i = 2, 3, ..., p-3$$

- Find the valley points in the modified histogram. A point s_i corresponding to the weight function $f(s_i)$ is said to be a valley point if $f(s_{i-1}) > f(s_i)$ and $f(s_i) < f(s_{i+1})$.

- The first valley point of the modified histogram is taken as the required threshold on the similarity values.

In the modified histogram corresponding to g, there can be three possibilities regarding the valley points, which are stated below.

i) There are no valley points in the histogram. If there is no valley point in the histogram then either it is a constant function, or it is an increasing or decreasing function of similarity values. These three types of histograms impose strong conditions on the similarity values which are unnatural for a document collection. Another possible histogram where there is no valley point is a *unimodal* histogram. There is a single mode in the histogram, and the number of occurrences of a similarity value increases as the similarity values increase to the mode, and decreases as the similarity values move away from mode. This is an unnatural setup since, there is no reason of having such a strong property to be satisfied by a histogram of similarity values.

ii) Another option is that there exists exactly one valley point in the histogram. The number of occurrences of the valley point is smaller than the number of occurrences of the other similarity values in a neighborhood of valley point. In practice this type of example is also rare.

iii) The third and most usual possibility is that the number of valley points is more than one, i.e., there exists several variations in the number of occurrences of similarity values. Here the task is to find a threshold from a particular valley. In the proposed technique the threshold is selected from the first valley point. The threshold may be selected from the second or third or a higher valley. But in this case some really similar documents may be treated as dissimilar, which lie in between the first valley point and the higher one. Practically the text data sets are sparse with a high dimensionality. Hence high similarities between documents are observed in very few cases. It is true that, for a high θ value the extensive similarity between every two documents in a cluster will be high, but the number of documents in each cluster will be too few due to the sparsity of the data. Hence θ is selected from the first valley point as the similarity values in the other valleys are higher than the similarity values in the first valley point.

Generally similarity values do not satisfy the property that $(s_{i+1} - s_i) = (s_1 - s_0),\ \forall i = 1, 2, ..., (p-2)$. In reality there are $(p+1)$ distinct class intervals of similarity values, where the i^{th} class interval is $[\chi_{i-1}, \chi_i)$, a semi-closed interval, for $i = 1, 2, ..., p$. The $(p+1)^{th}$ class interval corresponds to the set where each similarity value is greater than or equal to χ_p. $g(s_i)$ corresponds to the number of similarity values falling in the i^{th} class interval. The $\chi_i's$ are taken in such a way that $(\chi_{i-1} - \chi_i) = (\chi_1 - \chi_0),\ \forall i = 2, 3, ..., p$. Note that $\chi_0 = 0$ and the value of χ_p is decided on the basis of the observations. The last interval, i.e., the $(p+1)^{th}$ interval is not considered for the valley point selection, since we assume that if any similarity value is greater than or equal to χ_p then the corresponding documents are actually similar. Under this setup, the value of s_i is taken as $s_i = \frac{\chi_i + \chi_{i+1}}{2},\ \forall i = 0, 1, ..., (p-1)$. Note that the defined s_i's satisfy the properties a) $s_i < s_j,$ if $i < j$ and (b) $(s_{i+1} - s_i) = (s_1 - s_0),\ \forall i = 1, 2, ..., (p-2)$. The proposed method finds the valley point, and its corresponding class interval. The minimum value of that particular class interval is taken as the threshold.

Example: Let us consider an example of histogram thresholding for the selection of theta for a data set. The similarity values, the values of g and f are shown in Table 2.1. Initially the similarity values have been divided into a few class intervals of length 0.001. Let us assume that there are 80 such intervals of equal length and s_i represents the middle point of the i^{th} class interval for $i = 0, 1, ..., 79$. The

Table 2.1: An Example of θ Estimation by Histogram Thresholding Technique

Class Intervals ($\chi_i's$)	$s_i's$	No. of Elements of the Intervals	Moving Averages
$[0.000 - 0.001)$	0.0005	$g(s_0)$	–
$[0.001 - 0.002)$	0.0015	$g(s_1)$	–
$[0.002 - 0.003)$	0.0025	$g(s_2)$	$f(s_2)$
.	.	.	.
.	.	.	.
.	.	.	.
$[0.040 - 0.041)$	0.0405	$g(s_{40})$	$f(s_{40})$
.	.	.	.
.	.	.	.
.	.	.	.
$[0.077 - 0.078)$	0.0775	$g(s_{77})$	$f(s_{77})$
$[0.078 - 0.079)$	0.0785	$g(s_{78})$	–
$[0.079 - 0.080)$	0.0795	$g(s_{79})$	–
≥ 0.080	–	$g(s_{80})$	–

values of $g(s_i)$'s and the corresponding $f(s_i)$'s are then found. Note that the moving averages have been used to remove the ambiguities in the $g(s_i)$ values. Valleys in the similarity values corresponding to 76 $f(s_i)$'s are then found. Let s_{40}, which is equal to 0.0405, be the first valley point, i.e., $f(s_{39}) > f(s_{40})$ and $f(s_{40}) < f(s_{41})$. The minimum similarity value of the class interval $[0.040 - 0.041)$ is taken as the threshold θ.

2.3.5 Time and Space Complexity

The time and space complexity of the proposed clustering algorithm for N input documents is discussed here. In the initialization phase of Algorithm 1, a similarity matrix has been built up which takes $O(N^3)$ time in worst case. The merging procedure takes $O(N)$ time to merge two clusters and rest of the steps take $O(1)$ time to execute. The algorithm finds two clusters with minimum cluster distance in step 7 in at most $\frac{N\times(N-1)}{2}$ iterations and the merging procedure of step 15 takes at most $(N-1)$ iterations. Rest of the steps take $O(1)$ time to execute. Therefore to find two minimum distant clusters and then merge them to build a single cluster, it takes a total of at most $\left(\frac{N\times(N-1)}{2} + (N-1)\right)$, i.e., $O(N^2)$ time. Let there are m merges between two clusters, i.e., there are m iterations of the loop of step 2.

Thus the time complexity of CUES is $O(mN^2)$. In worst case there may be $(N-1)$ merges and thus the worst case time complexity of CUES is $O(N^3)$. Generally the text data sets are huge in size and the number of clusters are too less compared to the number of documents. Hence it is very unlikely in practical scenario for CUES to have the time complexity $O(N^3)$.

The similarity matrix requires $N \times N$ memory locations, and to store N clusters, initially, N memory locations are needed. Thus the space complexity of CUES is $O(N^2)$.

2.4 Experimental Evaluation

2.4.1 Experimental Setup

In order to evaluate the performance of CUES, eight basic clustering algorithms - bisecting k-means (BKM) [116], k-means (KM) [61], buckshot (BS) [32], single-link hierarchical clustering (SLHC) [126], average-link hierarchical clustering (ALHC) [116], k-nearest neighbor (kNN) [24] clustering and two types of spectral clustering methods (simple spectral clustering (SC) [47] and spectral clustering using a kernel (SCK) [94], as discussed in Chapter 1) are selected for comparison. k-means and bisecting k-means are executed 10 times to reduce the effect of random initialization of seed points and for each execution they have been iterated 100 times to reach a solution (if they are not converged automatically). Buckshot has also been executed 10 times to reduce the effect of random initialization of initial $\sqrt{kN}$ documents. The number of clusters of the other methods are same as the number of clusters produced by CUES. Note that the proposed method finds the total number of clusters automatically from the corpus. The neighborhood size of kNN clustering is chosen as $k = 10$ [60]. F-measure [116] and Normalized Mutual Information (NMI) [123] are used for evaluating each clustering technique. The f-measure and NMI values shown here for k-means and bisecting k-means are the average of 10 different results. A discussion is given on appendix B regarding the implementation of the competing algorithms used in this chapter.

The proposed histogram thresholding based technique for estimating a value of θ has been followed in the experiments. The class intervals of length 0.005 have been considered for the similarity values. It has been assumed that similarity (cosine similarity) value greater than 0.5 means that the corresponding documents are

similar. Thus, the issue here is to find a $\theta, 0 < \theta < 0.5$ such that a similarity value grater than θ denotes that the corresponding documents are similar. In the experiments the method of moving averages has been used with the window length of 5 for convolution. The text data sets are generally sparse and the number of high similarity values is practically very low and there are fluctuations in the heights of the histogram for two successive similarity values. Hence it is not desirable to take the window length of 3 as the method considers the heights of just the previous and the next value for calculating $f(s_i)$'s. The window length of 7 or 9 has been tried for experiment on some of the corpora, but the values of θ remain more or less same as they are selected by considering window length of 5. On the other hand window length of 7 or 9 need more calculations than window length of 5. These are the reasons for the choice of window of length 5. It has been found that several local peaks and local valleys are removed by this method. The number of valley regions after smoothing the histogram by the method of moving averages is always found to be greater than three.

Table 2.2 and Table 2.3 show the f-measure and NMI values respectively of all the data sets. Number of clusters (NC), number of singleton clusters (NSC) developed by CUES are also shown. Here number of clusters includes the singleton clusters also, i.e., the NSC values are included in the NC values. The f-measure and NMI are calculated using these NC values. All the data sets of Table 1.1 have been considered for experimentation in this chapter. Each document is represented by using the principles of vector space model [111], as described in section 1.2.

2.4.2 Analysis of Results

Each of the tables 2.2 and 2.3 has the results on 14 data sets and 8 methods (excluding the proposed one). Thus, from each table we have 112 comparisons with the results of the CUES. Out of the total of 224 comparisons, CUES performed better than the other methods in 206 cases, and the other methods performed better in the rest 18 cases. Few exceptions where the other methods have an edge over CUES in both of the tables are as follow: i) buckshot, k-means and spectral clustering (SCK) for *20ns* (0.435, 0.447 and 0.428 respectively for buckshot, k-means and SCK and the value of CUES is 0.394) when f-measure is used for performance evaluation, and ii) buckshot, k-means and SCK for *20ns* (0.437, 0.428 and 0.451 for buckshot, k-means and SCK and the value of CUES is 0.233) when NMI is used for performance

Table 2.2: Performance of Different Document Clustering Techniques on Various Text Data Sets using F-measure

Data Sets	θ	NC[1]	NSC[2]	F-measure								
				BKM[3]	KM	BS	SLHC	ALHC	kNN	SC	SCK	CUES (Proposed)
20ns	0.031	23	3	0.356	**0.447**	0.435	0.367	0.384	0.341	0.378	0.428	0.394
fbis	0.080	18	0	0.426	0.537	0.519	0.193	0.195	0.191	0.452	0.538	**0.545**
la1	0.005	8	2	0.503	0.528	0.504	0.327	0.321	0.326	0.448	0.537	**0.558**
la2	0.005	6	3	0.486	0.545	**0.553**	0.330	0.319	0.330	0.481	0.540	**0.553**
oh10	0.027	12	1	0.308	0.468	0.461	0.205	0.204	0.205	0.477	**0.527**	0.485
oh15	0.020	10	2	0.363	0.482	0.482	0.206	0.202	0.205	0.476	**0.516**	0.498
rcv1	0.077	34	3	0.188	0.212	0.307	0.408	0.362	0.418	0.301	0.318	**0.522**
rcv2	0.075	33	3	0.165	0.202	0.286	0.407	0.350	0.417	0.419	0.439	**0.551**
rcv3	0.085	32	2	0.218	0.239	0.355	0.409	0.372	0.413	0.211	0.331	**0.578**
rcv4	0.087	34	5	0.222	0.286	0.287	0.409	0.379	0.414	0.229	0.297	**0.590**
tr31	0.020	7	1	0.554	**0.652**	0.648	0.388	0.373	0.385	0.567	0.589	0.646
tr41	0.036	11	1	0.560	0.605	0.593	0.286	0.281	0.289	0.505	0.557	**0.617**
tr45	0.043	12	2	0.552	0.675	0.681	0.243	0.248	0.240	0.587	0.605	**0.695**
wap	0.037	20	0	0.283	0.412	0.417	0.177	0.180	0.178	0.174	0.381	**0.427**

[1] NC stands for number of clusters. [2] NSC stands for number of singleton clusters. [3] BKM, KM, BS, SLHC, ALHC, kNN, SC and SCK are - bisecting k-means, k-means, buckshot, single-link agglomerative hierarchical clustering, average-link hierarchical clustering, k nearest neighbor clustering , spectral clustering, and spectral clustering using kernel respectively and CUES is the proposed method.

evaluation.

A generalized version of paired *t-test* is suitable for testing the equality of means when the variances are unknown. This problem is the classical Behrens-Fisher problem in hypothesis testing and a suitable test statistic[4] is described and tabled in [80] and [107], respectively. It has been found that out of those 206 cases where CUES performed better than the other methods, the differences are found to statistically significant in 187 cases for the level of significance 0.05. On the other hand, 2 out of 18 cases the difference is found to be statistically significant when the other methods performed better for the same level of significance. Thus the performance of the proposed method is found to be *significantly better* than the other methods in 92.11 % cases.

It is to be noted that the results of t-test are significant in 20ns data for all the clustering algorithms and buckshot, k-means and SCK performed better than

[4]The test statistic is of the form $t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}}$, where $\bar{x}_1, \bar{x}_2$ are the means, s_1, s_2 are the standard deviations and n_1, n_2 are the number of observations

Table 2.3: Performance of Different Document Clustering Techniques on Various Text Data Sets using Normalized Mutual Information

Data Sets	θ	NC[4]	NSC	Normalized Mutual Information								
				BKM	KM	BS	SLHC	ALHC	kNN	SC	SCK	CUES (Proposed)
20ns	0.031	23	3	0.417	0.428	0.437	0.270	0.286	0.273	0.324	**0.451**	0.288
fbis	0.080	18	0	0.444	0.521	**0.523**	0.361	0.348	0.330	0.477	0.520	0.515
la1	0.005	8	2	0.269	0.293	0.295	0.221	0.216	0.216	0.240	0.285	**0.298**
la2	0.005	6	3	0.252	0.317	0.323	0.223	0.218	0.214	0.262	0.335	**0.366**
oh10	0.027	12	1	0.226	0.355	0.333	0.150	0.154	0.146	0.344	**0.417**	0.373
oh15	0.020	10	2	0.217	0.354	0.357	0.167	0.155	0.160	0.330	0.358	**0.370**
rcv1	0.077	34	3	0.300	0.434	0.444	0.083	0.108	0.160	0.190	0.381	**0.476**
rcv2	0.075	33	3	0.301	0.412	0.401	0.079	0.147	0.148	0.242	0.374	**0.466**
rcv3	0.085	32	2	0.312	0.394	0.401	0.078	0.162	0.133	0.194	0.390	**0.415**
rcv4	0.087	34	5	0.331	0.403	0.391	0.073	0.175	0.134	0.206	0.378	**0.416**
tr31	0.020	7	1	**0.478**	0.462	0.471	0.265	0.242	0.238	0.414	0.436	0.468
tr41	0.036	11	1	0.470	0.551	0.553	0.254	0.240	0.250	0.433	0.479	**0.577**
tr45	0.043	12	2	0.490	0.600	0.591	0.384	0.354	0.367	0.481	0.503	**0.609**
wap	0.037	20	0	0.268	0.412	0.423	0.075	0.041	0.072	0.126	0.426	**0.456**

[5] All the symbols in this Table are the same symbols used in Table 2.2.

CUES. In *20ns* data set there are overlaps between several categories. In general the centroid based algorithms, like k-means, buckshot have been found to produce better results than the hierarchical algorithms when there are overlaps between clusters [67]. Thus buckshot, k-means and SCK have produced better clusters in *20ns* data sets. Otherwise, the overall experimental evaluation shows the effectiveness of the extensive similarity based document clustering approach. The extensive similarity based method groups two documents not only based on their mutual similarity but also on their similarity with the other documents in the document collection. This fact is observed in the experimental results.

Table 2.4 shows the performance of the proposed clustering algorithm on different θ values, say $\theta = 0.1$, 0.2, 0.4. Since the sizes of the data sets are very large and average number of terms per document is very low compared to the number of terms of the vocabulary, the average similarity between the documents is very low. As a result total number of singleton clusters will be more on high values of θ, which is not desirable in practice. It can be seen from Table 2.4 that for $\theta = 0.1$, 0.2 and 0.4, the NC and NSC values are very high. Therefore $\theta = 0.1$, 0.2 and 0.4 can not be used for comparison with the other methods. In this situation, low values of θ have to be taken for experiment. The values of θ chosen for experiments are close to 0 in most of the data sets due to the sparsity of the data. Even then CUES outperforms

Table 2.4: Performance of CUES on Different Values of θ

Data Set	$\theta = 0.1$				$\theta = 0.2$				$\theta = 0.4$			
	NC	NSC	FM[6]	NMI[7]	NC[8]	NSC	FM	NMI	NC	NSC	FM	NMI
20ns	374	143	0.361	0.274	812	489	0.254	0.307	1427	1020	0.176	0.341
fbis	23	0	0.520	0.499	101	16	0.442	0.500	697	389	0.264	0.554
la1	223	31	0.271	0.324	1016	489	0.126	0.422	2261	1771	0.036	0.460
la2	227	29	0.256	0.357	962	462	0.124	0.430	2131	1665	0.048	0.462
oh10	81	6	0.365	0.412	331	113	0.166	0.489	852	725	0.075	0.557
oh15	86	7	0.440	0.457	320	130	0.188	0.527	732	615	0.084	0.565
rcv1	63	4	0.532	0.464	387	159	0.461	0.540	1139	890	0.214	0.536
rcv2	68	10	0.549	0.499	382	150	0.473	0.549	1170	953	0.281	0.542
rcv3	47	5	0.518	0.437	431	192	0.469	0.554	1187	956	0.164	0.536
rcv4	47	8	0.598	0.464	386	159	0.413	0.532	1200	989	0.197	0.535
tr31	50	9	0.435	0.509	161	58	0.324	0.508	453	288	0.164	0.489
tr41	63	13	0.577	0.646	176	69	0.424	0.609	483	334	0.204	0.568
tr45	54	20	0.635	0.608	172	85	0.431	0.617	413	310	0.242	0.593
wap	157	20	0.358	0.515	532	225	0.231	0.574	1044	766	0.149	0.599

[6] FM stands for f-measure. [7] NMI stands for Non Negative Matrix Factorization. [8] All the other symbols in this Table are the same symbols used in Table 2.2.

the other methods. Thus the proposed histogram thresholding approach has been found to yield a good estimate of θ.

2.4.3 Processing Time

The processing time of each method has been measured on a quad core Linux workstation. The times (in seconds) taken by different clustering methods to cluster different corpora are reported in Table 2.5. The time shown here for CUES is the sum of the times taken to estimate the value of θ, to build the similarity matrix, and to perform clustering. The times shown for bisecting k-means, buckshot and k-means are the average of the processing times for 10 repeated executions. The time for pre-processing of each data set, i.e., to build the tf-idf matrix from the raw text data, is not included in the execution time of each method. The time shown in Table 2.5 for a particular method is the execution time of that method on the tf-idf data matrix of a data set. It is to be mentioned that the codes for all the algorithms are written in C++ and the data structures for all the algorithms are developed by the authors. Hence the processing time can be reduced by incorporating some more efficient data structures for the proposed algorithm as well as the other methods. It may be observed from Table 2.5 that the time taken by CUES is less than the time taken by ALHC for each data set. The processing time of kNN is less than

Table 2.5: Processing Time (in seconds) of Different Document Clustering Methods

Data	BKM	KM	BS	SLHC	ALHC	kNN	SC	SCK	CUES
20ns	1582.25	1594.54	1578.12	1618.50	1664.31	1626.38	1577.31	1583.62	1670.21
fbis	94.28	91.33	91.96	112.15	129.65	95.44	92.75	100.90	99.36
la1	159.37	154.02	142.24	160.12	179.67	131.32	144.48	146.68	152.21
la2	149.15	144.66	139.47	163.47	182.47	133.28	143.70	142.33	151.55
oh10	18.37	18.87	17.26	26.31	33.49	21.54	19.69	24.82	25.31
oh15	18.12	20.02	18.26	24.15	31.48	20.14	19.58	20.94	24.72
rcv1	92.11	91.46	88.37	87.47	103.33	77.58	80.35	99.62	92.54
rcv2	104.23	114.88	112.10	104.17	120.41	100.23	111.17	115.21	109.39
rcv3	97.12	106.20	98.31	98.47	124.74	92.25	105.36	112.24	103.37
rcv4	100.12	95.54	96.45	109.32	126.24	95.31	108.38	120.87	115.28
tr31	29.41	30.57	30.43	33.15	40.11	28.38	34.68	37.98	38.68
tr41	27.29	28.31	27.49	33.96	39.58	27.17	25.64	26.85	33.77
tr45	25.45	25.08	26.11	31.17	38.24	25.54	25.23	26.12	32.15
wap	26.27	33.56	25.46	35.27	43.39	29.34	30.41	42.35	36.30

the time taken by CUES for each data set. The processing times of CUES are less than BKM, KM, BS, SLHC, SC and SCK for some of the data sets and for the other data sets the times taken to create the clusters by BKM, KM, BS, SLHC, SC and SCK are less than CUES. Hence one can not make a general comment that the computational times of CUES are better or worse than the other methods. It is true that the processing times of some of the methods e.g., BKM, BS and SC are better than CUES for most of the data sets, but CUES outperforms all the other methods in terms of cluster quality. Hence one can allow this much extra cost of expenditure of CUES due to time to get such a good performance for clustering of documents. The data sets used in the experiments have several dimensions starting from 2000 (fbis) to 35218 (20ns), and CUES is found to perform well in the presence of high dimensional data.

2.5 Conclusions and Discussion

A system has been developed, which composed of extensive similarity between documents to improve the accuracy of measuring the similarity between documents. The similarity measure has been used to perform document clustering. Intuitively, the similarity between two documents can not be obtained totally from their content, but from their inherent extensive similarity in a corpus. The extensive similarity has been introduced on the basis of similarity between two documents and their dis-

tances with every other document in the corpus. ES has been defined using cosine similarity, since it can find similarity between two documents of different lengths. ES can be defined using any other similarity or dissimilarity measure. Additionally, ES can be used not only for clustering but also for other tasks such as classification. It may be noted here that in Chapter 5 an algorithm for text categorization is proposed using ES.

CUES has been developed using ES. The documents with high ES values could be grouped in the same cluster. The salient features of CUES are given here. The algorithm can identify two dissimilar clusters and never merges them. The algorithm can be stopped if the distance between two clusters becomes very high, since at each step CUES checks the cluster distance to merge two clusters. There is no need to input the desired number of clusters prior to implement the algorithm. The range of similarity values between every two documents in a cluster is known to us and it is between θ and 1. The total number of clusters is determined automatically by the proposed method, but on requirement the total number of clusters can be bounded by varying the value of θ.

The performance of CUES is mainly dependent on θ, and the selection of θ is dependent upon two factors - nature of the data sets and the choice of the user. A histogram thresholding based method has been explained for selecting a value of θ from the actual similarity matrix of a data set. The results reveal the fact that using those θ values the proposed approach performs significantly better than other traditional approaches. The method presented here is aimed at document clustering, but it can be easily generalized to any data set as well.

Chapter 3

A Hierarchical Approach to k-Means Clustering for Effective Grouping of Documents

3.1 Introduction

In this chapter, another new document clustering algorithm is proposed. The document clustering technique CUES, proposed in the previous chapter is purely hierarchical in nature. It has been shown that CUES outperforms the other clustering techniques to cluster several document corpus, but it suffers from computational cost in some of the data sets of large size or dimensionality. The proposed one in this chapter is designed in such a way that it reduces the computational cost due to hierarchical design of CUES, but it maintains the quality of the clusters obtained by CUES. The proposed document clustering method is a combination of hierarchical clustering and k-means clustering techniques. It has been designed using *extensive similarity* between documents. A new distance function is defined here to find the distance between two hierarchical clusters by using extensive similarity. The algorithm is a two stage process and in the first stage, a few well segregated non singleton clusters are created. These non singleton clusters are named as *baseline clusters*. In the second stage, k-means algorithm is performed to cluster the rest of the documents, which are not grouped in the baseline clusters by using the centroids of the baseline clusters as the initial seeds.

The proposed clustering technique need not require the knowledge of desired

number of clusters, like CUES. It is automatically determined by varying the similarity threshold of the extensive similarity. The same histogram thresholding based technique proposed in the previous chapter is used here to estimate the value of the content similarity threshold of extensive similarity. The proposed method is compared with CUES and several other clustering techniques. The computational time of the proposed technique is reported and discussed. The material in this chapter is taken from the article [14].

The chapter is organized as follows - section 3.2 describes the proposed document clustering technique. Section 3.3 presents the experimental evaluation on a number of well known text data sets. The conclusions about the proposed work are presented in chapter 3.4.

3.2 Proposed Hierarchical Approach to k-Means Method for Effective Grouping of Documents

A combination of hierarchical clustering and k-means clustering methods is introduced to enhance the effectiveness of document clustering. The proposed method is a two stage process. In the first stage a hierarchical clustering algorithm is performed using *extensive similarity* between documents with the help of a new distance function between clusters, thereby, generating some non singleton clusters. Each of these non singleton clusters is named as *baseline cluster*. In the second stage k-means clustering algorithm is performed using the centroids of these baseline clusters as initial seed points. The entire method is described in detail in Algorithm 2.

3.2.1 Baseline Cluster

A distance function is proposed to create the baseline clusters. It finds the distance between two clusters say, C_x and C_y. Let T_{xy} be a multi set consisting of the extensive similarities between each pair of documents, one from C_x and the other from C_y and it is defined as,

$$T_{xy} = \{ES(d_i, d_j) : \quad ES(d_i, d_j) \geq 0, \quad \forall\, d_i \in C_x \quad \text{and}\, d_j \in C_y\}$$

Note that T_{xy} consisting of all the occurrences of the same extensive similarity values (if any) for different pairs of documents. The proposed distance between two clusters

C_x and C_y can be defined as

$$dist\_cluster(C_x, C_y) = \begin{cases} \infty & \text{if } T_{xy} = \emptyset \\ N - \text{avg}(T_{xy}) & \text{otherwise} \end{cases} \tag{3.1}$$

The function *dist_cluster* finds the distance between two clusters C_x and C_y as the average of the multi set of non-negative ES values. The distance between C_x and C_y is infinite, if there are no two documents that have a non-negative ES value i.e., no similar documents are present in C_x and C_y. Intuitively, infinite distance between clusters denotes that every pair of documents, one from C_x and the other from C_y either share a very few number of terms, or no term is common between them i.e., they have a very low content similarity. Later we shall observe that any two clusters with infinite distance between them remain segregated from each other. Thus, a significant characteristic of the function *dist_cluster* is that it would never merge two clusters with infinite distance between them.

The proposed document clustering algorithm initially assumes each document as a singleton cluster. Then it merges those clusters with minimum distance, and the distance is within a previously fixed limit α. The merging process continues until the distance between every two clusters is greater than α. The clusters which are not singletons are named as *Baseline Clusters* (BC). The selection of the value of α is discussed in section 3.3 of this chapter.

3.2.2 Properties of *dist_cluster*

The important properties of the function *dist_cluster* are described below.

- The minimum distance between any two clusters C_x and C_y is 0, when $\text{avg}(T_{xy}) = N$, i.e., the extensive similarity value between every pair of documents, one from C_x and the other from C_y is N. Although in practice this minimum value can be rarely observed between two different document clusters. The maximum value of *dist_cluster* is infinite.

- If $C_x = C_y$ then $dist\_cluster(C_x, C_y) = N - \text{avg}(T_{xx}) = 0$

- On the other hand, let $dist\_cluster(C_x, C_y) = 0$

$$\Rightarrow \text{avg}(T_{xy}) = N$$

$$\Rightarrow ES(d_i, d_j) = N, \ \forall \ d_i \in C_x \ \text{ and } d_j \in C_y$$

Now $ES(d_i, d_j) = N$ implies that two documents d_i and d_j are exactly similar. Note that $ES(d_i, d_j) = N \Rightarrow dis(d_i, d_j) = 0$ and $l_{ij} = 0$. Here $dis(d_i, d_j) = 0$ implies that d_i and d_j are similar in terms of content, but they are not necessarily same, i.e., we can not say $d_i = d_j$, if $dis(d_i, d_j) = 0$.

Thus $dist\_cluster(C_x, C_y) = 0 \not\Rightarrow C_x = C_y$ and hence $dist\_cluster$ is not a metric.

- It is symmetric. For every pair of clusters C_x and C_y, we have

  $$dist\_cluster(C_x, C_y) = dist\_cluster(C_y, C_x).$$

- $dist\_cluster(C_x, C_y) \geq 0$ for any pair of clusters C_x and C_y.

- For any three clusters C_x, C_y and C_0, we may have

  $$dist\_cluster(C_x, C_y) + dist\_cluster(C_y, C_0) - dist\_cluster(C_x, C_0) < 0$$

  when $0 \leq dist\_cluster(C_x, C_y) < N,\ 0 \leq dist\_cluster(C_y, C_0) < N$ and $dist\_cluster(C_x, C_0) = \infty$. Thus it does not satisfy the triangular inequality.

3.2.3 Procedure of the Proposed Document Clustering Technique

The proposed document clustering technique is described in Algorithm 2. Initially each document is taken as a cluster. Therefore Algorithm 2 starts with N individual clusters. In the first stage of Algorithm 2, a distance matrix is developed whose ij^{th} entry is the dist_cluster(C_i, C_j) value where C_i and C_j are i^{th} and j^{th} cluster respectively. It is a square matrix and has N rows and N columns for N number of documents in the corpus. Each row or column of the distance matrix is treated as a cluster. Then the Baseline Clusters (BC) are generated by merging the clusters whose distance is less than a fixed threshold α. The value of α is constant throughout Algorithm 2. The process of merging stated in step 3 of Algorithm 2 merges two rows say i and j and the corresponding columns of the distance matrix by following a convention regarding numbering. It merges two rows into one, the resultant row is numbered as minimum of i, j, and the other row is removed. Similar numbering follows for columns too. Then the index structure of the distance matrix is updated accordingly.

Algorithm 2 Iterative Document Clustering by Baseline Clusters

Input: a) A set of clusters $C = \{C_1, C_2, ..., C_N\}$, where N is the number of documents. $C_i = \{d_i\},\ i = 1, 2, ..., N$, where d_i is the i^{th} document of the corpus.
b) A distance matrix $DM[i][j] = \text{dist\_cluster}(C_i, C_j),\ \forall i, j \in N$.
c) α be the desired threshold on *dist_cluster* and *iter* be the number of iteration.

Steps of the Algorithm:

for each clusters $C_i, C_j \in C$ where $C_i \neq C_j$ and $N > 1$ **do**
  if $dist\_cluster(C_i, C_j) \leq \alpha$ **then**
    $DM \leftarrow \text{merge}(DM, i, j)$
    $C_i \leftarrow C_i \cup C_j$
    $N \leftarrow N - 1$
  end if
end for
$nbc \leftarrow 0,\ \ BC \leftarrow \emptyset$ //Baseline clusters are initialized to empty set
$nsc \leftarrow 0,\ \ SC \leftarrow \emptyset$ //Singleton clusters are initialized to empty set
for $i = 1$ to N **do**
  if $|C_i| > 1$ **then**
    $nbc \leftarrow nbc + 1$ // No. of baseline clusters
    $BC_{nbc} \leftarrow C_i$ // Baseline clusters
  else
    $nsc \leftarrow nsc + 1$ // No. of singleton clusters
    $SC_{nsc} \leftarrow C_i$ // Singleton clusters
  end if
end for
if $nsc = 0\ ||\ nbc = 0$ **then**
  return BC // If no singleton cluster at all exists or no baseline cluster is generated
else
  $EBC_k \leftarrow BC_k, \forall\ k = 1, 2, ..., nbc$ // Initialization of extended baseline clusters
  $ebct_k \leftarrow$ centroid of $BC_k, \forall\ k = 1, 2, ..., nbc$ // Extended base centroids
  $nct_k \leftarrow (0), \forall\ k = 1, 2, ..., nbc,\ \ it \leftarrow 0$
  while $ebct_k \neq nct_k, \forall\ k = 1, 2, ..., nbc$ and $it \leq iter$ **do**
    $ebct_k \leftarrow$ centroid of $EBC_k, \forall\ k = 1, 2, ..., nbc$
    $NCL_k \leftarrow BC_k, \forall\ k = 1, 2, ..., nbc$ // New set of clusters at each iteration
    for $j = 1$ to nsc **do**
      if $ebct_k$ is the nearest centroid of $SC_j,\ \forall k = 1, 2, ..., nbc$ **then**
        $NCL_k \leftarrow NCL_k \cup SC_j$ // Merger of singleton clusters to baseline clusters
      end if
    end for
    $nct_k \leftarrow$ centroid of $NCL_k,\ \forall k = 1, 2, ..., nbc$
    $EBC_k \leftarrow NCL_k,\ \forall k = 1, 2, ..., nbc$
    $it = it + 1$
  end while
  return EBC
end if

Output: A set of extended baseline clusters $EBC = \{EBC_1, EBC_2, ..., EBC_{nbc}\}$

After constructing the baseline clusters some clusters may remain as singleton clusters. Every such singleton cluster (i.e., a single document) is merged with one of the baseline clusters using k-means algorithm in the second stage. In the second stage the centroids of the baseline clusters (i.e., non singleton clusters) are calculated and they are named as *base centroids*. The value of k for k-means algorithm is taken as the number of baseline clusters. The rest of the documents which are not included in the baseline clusters are clustered by the iterative steps of the k-means algorithm using these base centroids as the initial seed points. Note that, those documents, which are not included in the baseline clusters, are only considered for clustering in this stage. But, for the calculation of a cluster centroid, every document in the cluster, including the documents in the baseline clusters, are considered. A document is put into that cluster for which the content similarity between the document and the base centroid is maximum. The newly formed clusters are named as *Extended Baseline Clusters* (EBC).

It may be noted that the processing in the second stage is not needed if no singleton cluster is produced in the first stage. We believe that such a possibility is remote in real life and none of our experiments yielded such an outcome. However, such a clustering is desirable as it produces compact clusters.

3.2.4 Impact of Extensive Similarity on the Document Clustering Technique

The extensive similarity plays significant role in constructing the baseline clusters. The documents in the baseline clusters are very similar to each other as their extensive similarity is very high (above a threshold θ). It may be observed that whenever two baseline clusters are merged in the first stage, the similarity between any two documents in the baseline clusters are at least be equal to θ. Note that the distance between two different baseline clusters is greater than or equal to α and the distance between a baseline cluster and a singleton cluster (or between two singleton clusters) may be infinite and they would never merge to construct a new baseline cluster. Infinite distance between two clusters indicates that the extensive similarity between at least one document of the baseline cluster and the document of the singleton cluster (or, between the documents of two different singleton clusters) is -1. Thus the baseline clusters intuitively determine the categories of the document collection by measuring the extensive similarity between documents.

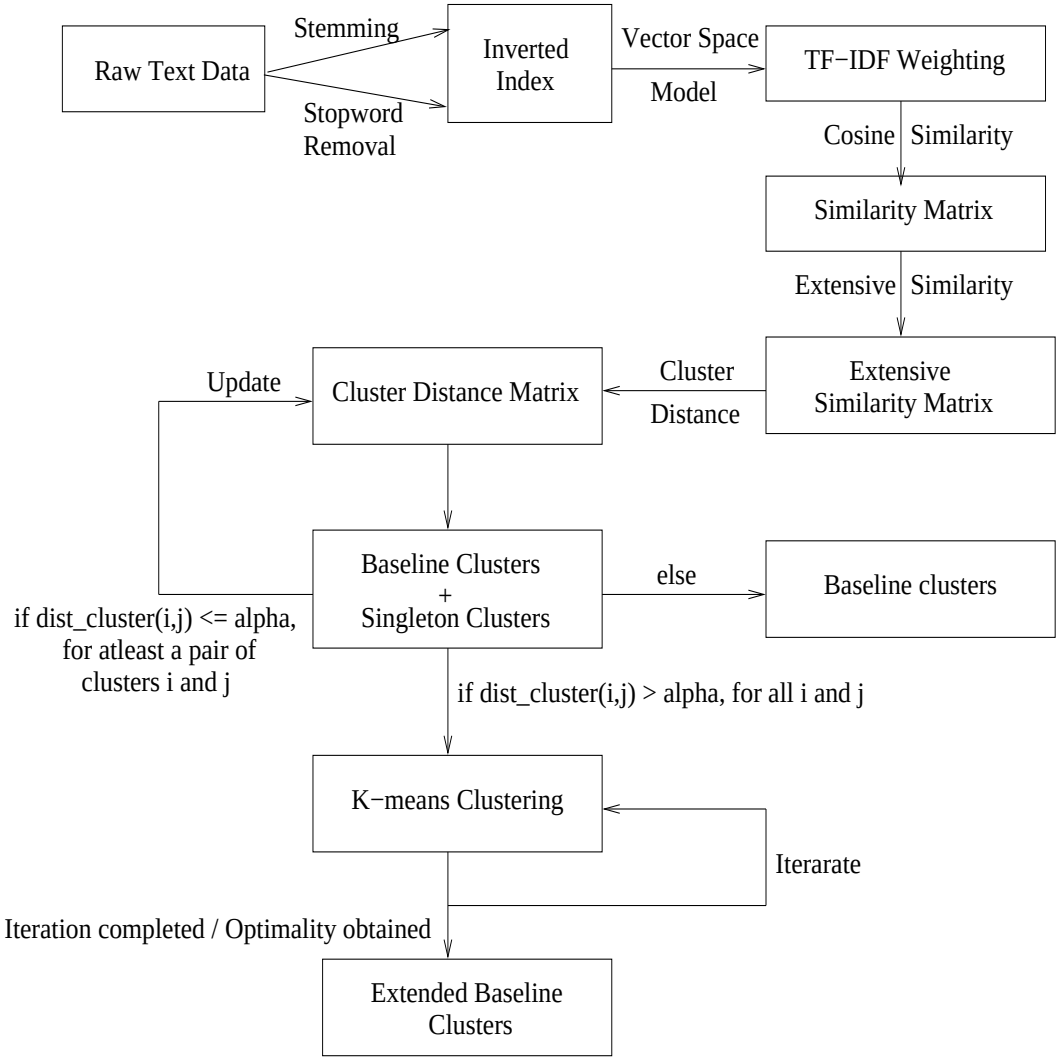

Figure 3.1: Document Clustering by the Proposed Baseline Clustering Technique

3.2.5 Discussion

The proposed clustering method is a combination of baseline clustering and k-means clustering methods. Initially it creates some baseline clusters. The documents which do not have much similarity with any one of the baseline clusters would remain as singleton clusters. Therefore k-means method is implemented to group these documents to the corresponding baseline clusters. k-means algorithm has been used due to its low computational complexity. It is also useful as it can be easily implemented. But the performance of k-means suffers from selection of initial seed points and there is no method for selecting a valid k. It is very difficult to select a proper k for a sparse text data set with high dimensionality. In various other clustering techniques, k-means has been used as an intermediary stage e.g., spectral clustering, buckshot etc. These algorithms also suffer from the said limitations of k-means method. Note that the proposed clustering method overcomes these two major limitations of k-means clustering algorithm and has utilized the effectiveness of k-means method by introducing the idea of baseline clusters. The effectiveness of the proposed technique in terms of clustering quality may be observed in the experimental results section later.

The proposed technique is designed like buckshot clustering algorithm. The main difference between buckshot and the proposed one lies in designing the hierarchical clusters in the first stage of both of the methods. Buckshot uses the traditional single-link clustering technique to develop the hierarchical clusters to create the initial centroids of the k-mean clustering in the second stage. Thus buckshot may suffer from the limitations of both single-link clustering technique (e.g., chaining effect) and k-means clustering technique. In practice the text data sets contain many categories of uneven sizes. In those data sets initial random selection of $\sqrt{kn}$ may not be proper, i.e, no documents may be selected from an original cluster if its size is small. Note that no random sampling is required for the proposed clustering technique. In the proposed one the hierarchical clusters are created using extensive similarity between documents, and these hierarchical baseline clusters would no more be used in the second stage. In the second stage, k-means algorithm is performed only to group those documents that have not been included in the baseline clusters and the initial centroids are generated from these baseline clusters. In Buckshot algorithm all the documents are taken into consideration for clustering by k-means algorithm. It uses the single-link clustering technique only to create the initial seed

points of the k-means algorithm. Later it can be seen from the experiments that the proposed one performs significantly better than buckshot clustering technique.

The process of creating baseline clusters in the first stage of the proposed technique is quite similar to the group-average hierarchical clustering technique [116]. Both the techniques find the average of similarities of the documents of two individual clusters for merging them into one. The proposed method finds the distance between two clusters using extensive similarity, whereas the group-average hierarchical clustering technique uses cosine similarity. The group-average hierarchical clustering technique cannot distinguish two dissimilar clusters explicitly, like the proposed method. This is the main difference between the two techniques.

3.3 Experimental Evaluation

The experiments have been done on all the data sets explained in Table 1.1. The number of documents, number of terms and number of categories of these data sets can be found in Table 1.1.

Single-link hierarchical clustering (SLHC) [116], average-link hierarchical clustering (ALHC) [116], k-means clustering [61], bisecting k-means clustering [116], buckshot clustering [32], spectral clustering by using kernel function (SCK) [94], clustering by matrix factorization (NMF) [128] and CUES [11] algorithms are selected for comparison with the proposed clustering technique. k-means and bisecting k-means are executed 10 times to reduce the effect of random initialization of seed points and for each execution they have been iterated 100 times to reach a solution (if they are not converged automatically). Buckshot algorithm has also been executed 10 times to reduce the effect of random initialization of initial $\sqrt{kN}$ documents. The f-measure and NMI values of$k$-means, bisecting k-means and buckshot clustering techniques shown here are the average of 10 different results. Note that the proposed algorithm finds the number of clusters automatically from the data sets. The proposed algorithm has been executed first and then all the other algorithms have been executed to produce the same number of clusters as the proposed one. The value of θ of CUES is chosen manually here to produce same number of clusters as the proposed method for each data set. Table 3.1 and Table 3.2 show the f-measure and NMI values respectively for all the data sets. Number of clusters (NCL) developed by the proposed method is also shown. The f-measure and NMI are calculated using these NCL values. The value of α of the proposed technique is

Table 3.1: Performance of Different Document Clustering Methods on Various Text Data Sets using F-measure

Data Sets	NCT[1]	NCL[2]	F-measure								
			BKM[3]	KM	BS	SLHC	ALHC	SCK	NMF	CUES	Proposed
20ns	20	23	0.358	0.449	0.436	0.367	0.384	0.426	0.445	0.394	**0.474**
fbis	17	19	0.423	0.534	0.516	0.192	0.192	0.535	0.435	0.542	**0.584**
la1	6	8	0.506	0.531	0.504	0.327	0.325	0.536	0.544	0.558	**0.570**
la2	6	6	0.484	0.550	0.553	0.330	0.328	0.541	0.542	0.553	**0.563**
oh10	10	12	0.304	0.465	0.461	0.205	0.206	**0.527**	0.481	0.485	0.500
oh15	10	10	0.363	0.485	0.482	0.206	0.202	0.516	0.478	0.498	**0.532**
rcv1	30	31	0.201	0.229	0.325	0.416	0.371	0.327	0.516	0.528	**0.551**
rcv2	30	30	0.193	0.217	0.314	0.415	0.360	0.442	0.489	**0.555**	0.519
rcv3	30	32	0.216	0.241	0.351	0.409	0.373	0.342	0.411	**0.578**	0.301
rcv4	30	32	0.236	0.308	0.291	0.405	0.381	0.314	0.409	**0.593**	0.295
tr31	7	7	0.558	0.665	0.646	0.388	0.387	0.589	0.545	0.656	**0.678**
tr41	10	10	0.564	0.607	0.593	0.286	0.280	0.557	0.537	0.617	**0.698**
tr45	10	11	0.556	0.673	0.681	0.243	0.248	0.605	0.596	0.695	**0.750**
wap	20	22	0.279	0.408	0.420	0.175	0.180	0.388	**0.442**	0.424	0.385

[1]NCT stands for number of categories. [2]NCL stands for number of clusters.
[3]BKM, KM, BS, SLHC, ALHC, SCK, NMF and CUES stand for bisecting k-means, k-means, buckshot, single-link hierarchical clustering, average-link hierarchical clustering, spectral clustering, non-negative matrix factorization respectively and clustering using extensive similarity.

chosen as, $\alpha = \sqrt{\sqrt{N}}$ for N number of documents in the corpus. The NMF based clustering technique has been executed 10 times to reduce the effect of random initialization and for each time it has been iterated 100 times to reach a solution. The value of σ of the spectral clustering technique is set by search over values from 10 to 20 percent of the total range of the similarity values and the one that gives the tightest clusters is picked, as suggested by Ng. et al. [94]. The histogram thresholding based technique stated in chapter 2 has been used in the experiments here for estimating θ. The technique has been implemented exactly in the same way as it has been mentioned in chapter 2. A discussion is given on appendix B regarding the implementation of the competing algorithms used in this chapter.

Table 3.1 and Table 3.2 show the comparison of proposed document clustering method with the other methods using f-measure and NMI respectively, for all data sets. There are 112 comparisons for the proposed method using f-measure in Table 3.1. The proposed one performs better than the other methods in 93 cases and for the rest 19 cases other methods (e.g., buckshot, spectral clustering) have an edge over the proposed method. Few of the exceptions, where the other methods perform

Table 3.2: Performance of Different Document Clustering Methods on Various Text Data Sets using Normalized Mutual Information

Data Sets	NCT	NCL	Normalized Mutual Information								
			BKM[4]	KM	BS	SLHC	ALHC	SCK	NMF	CUES	Proposed
20ns	20	23	0.417	0.428	0.437	0.270	0.286	**0.451**	0.432	0.288	0.433
fbis	17	19	0.443	0.525	0.524	0.051	0.362	0.520	0.446	0.515	**0.544**
la1	6	8	0.266	0.299	0.295	0.021	0.218	0.285	0.296	0.298	**0.308**
la2	6	6	0.249	0.312	0.323	0.021	0.215	0.335	0.360	0.366	**0.386**
oh10	10	12	0.226	0.352	0.333	0.050	0.157	**0.417**	0.410	0.373	0.406
oh15	10	10	0.213	0.352	0.357	0.067	0.155	0.358	0.357	0.370	**0.380**
rcv1	30	31	0.309	0.429	0.451	0.087	0.108	0.418	0.434	0.476	**0.498**
rcv2	30	30	0.312	0.421	0.410	0.083	0.150	0.405	0.420	0.468	**0.470**
rcv3	30	32	0.315	0.411	0.406	0.079	0.166	0.394	**0.476**	0.415	0.446
rcv4	30	32	0.337	0.414	0.416	0.078	0.165	0.394	**0.507**	0.418	0.454
tr31	7	7	0.478	0.463	0.471	0.065	0.212	0.436	0.197	0.468	**0.509**
tr41	10	10	0.470	0.550	0.553	0.054	0.237	0.479	0.506	0.577	**0.619**
tr45	10	11	0.492	0.599	0.591	0.084	0.354	0.503	0.488	0.609	**0.694**
wap	20	22	0.271	0.405	0.416	0.075	0.043	0.418	**0.507**	0.452	0.423

[4]All the symbols in this Table are the same symbols used in Table 3.1

better than the proposed one are e.g., SCK and CUES for rcv4 (Here the f-measure of SCK and CUES are respectively 0.314, 0.593 and the f-measure of the proposed method is 0.295). Similarly, Table 3.2 shows that the proposed method performs better than the other methods using NMI in 104 out of 112 cases.

Statistical test for judging the significant difference is performed for every comparison here. This test has already been described in section 2.4.2. It has been found that out of the 93 cases where the proposed algorithm performed better than the other algorithms, in Table 3.1, the differences are statistically significant in 85 cases for the level of significance 0.05. On the other hand, 2 out of the 19 differences are significant where the other methods are found to perform better than the proposed method in Table 3.1, for the same level of significance. Hence the performance of the proposed method is found to be significantly better than the other methods in 83.65% (87/104) cases using f-measure. Similarly, in Table 3.2 the results are significant in 90 out of 104 cases when proposed method performed better than the other methods and the results are significant in 2 out of the other 6 cases when the other methods performed better than the proposed one. Thus in 95.74% (90/94) cases the proposed method performs significantly better than the other methods using NMI. Clearly, these results show the effectiveness of the proposed document clustering technique.

3.3.1 Remark:

A point to be mentioned that the number of clusters produced by the proposed method is close to the number of actual categories of each corpus. It may be observed from Table 3.1 and Table 3.2 that the number of clusters is equal to the actual number of categories for la2, oh15, rcv2, tr31 and tr41 corpora. The difference between the number of clusters and actual number of categories is at most 3 for rest of the corpora. The proposed method has two parameters, α and θ. Since the text data sets provided here are very sparse with a high dimensionality, it may be implied that the choice of α and the method proposed for estimating the value of θ is able to detect the actual grouping of the corpora.

3.3.2 Choice of α :

The value of α is chosen heuristically, and it is taken as $\alpha = \sqrt{\sqrt{N}}$. The distance between two different clusters must be greater than α. It is very difficult to fix a lower bound on the distance between two clusters in practice as the corpora are sparse in nature and have high dimensionality. It has been observed that, when $\alpha > \sqrt{\sqrt{N}}$, say $\alpha = N^{\frac{1}{3}}$ or $\alpha = N^{\frac{1}{2}}$ then some really different clusters may be merged into one, which is surely not desirable. On the other hand, if $\alpha < \sqrt{\sqrt{N}}$, say, $\alpha = N^{\frac{1}{5}}$ then many small sized clusters would be created for any corpus of standard size. It has been observed from the experiments that the number of clusters produced by the proposed technique is very close to the actual number of categories for each corpus and the proposed method outperforms the other methods. Hence the selection of $\alpha = \sqrt{\sqrt{N}}$ is appropriate, though it has been selected heuristically.

3.3.3 Time and Space Complexity

The similarity matrix requires $N \times N$ memory locations, and to store N clusters, initially, N memory locations are needed for the proposed method. Thus the space complexity of the proposed document clustering algorithm is $O(N^2)$. $O(N^3)$ time is required to build the extensive similarity matrix and to construct (say) $m(<< N)$ baseline clusters, the proposed method takes $O(mN^2)$ time. In the final stage of the proposed technique, k-means algorithm takes $O((N-a)mt)$ time to merge (say) a singleton clusters to the baseline clusters, where t is number of iterations of k-means algorithm. Thus the time complexity of the proposed algorithm is $O(mN^2)+O((N-$

Table 3.3: Processing Time (in seconds) of Different Document Clustering Techniques

Methods	BKM[5]	KM	BS	SLHC	ALHC	SCK	NMF	CUES	Proposed
20ns	1582.25	1594.54	**1578.12**	1618.50	1664.31	1583.62	1587.36	1670.21	1595.23
fbis	94.17	91.52	92.46	112.15	129.58	100.90	93.19	99.36	**90.05**
la1	159.23	153.22	142.36	160.12	179.62	146.68	153.50	152.21	**140.31**
la2	149.41	144.12	**139.34**	163.47	182.50	142.33	144.46	151.55	140.29
oh10	18.57	18.32	**17.32**	26.31	33.51	24.82	22.48	25.31	18.06
oh15	18.12	20.02	18.26	24.15	31.46	20.94	17.61	24.72	**16.22**
rcv1	91.61	90.28	86.37	87.23	103.22	97.38	87.80	92.26	**86.18**
rcv2	103.32	112.08	111.31	103.76	120.25	97.81	93.51	109.18	**92.53**
rcv3	97.11	106.29	98.35	98.47	124.72	108.24	94.96	103.37	**93.54**
rcv4	99.42	95.08	96.01	109.14	126.05	113.81	98.70	115.04	**93.32**
tr31	29.41	30.23	30.34	33.15	40.13	37.98	**29.33**	38.68	29.38
tr41	27.29	28.16	27.46	33.96	39.58	**25.85**	27.54	33.77	26.49
tr45	25.45	25.01	26.06	31.17	38.23	29.72	26.51	32.15	**24.65**
wap	27.21	34.13	**26.09**	36.07	44.12	43.04	40.38	37.11	27.14

[5]All the symbols in this Table are the same symbols used in Table 3.1

$a)mt$), where m is a very small number compared to N.

The processing time taken by each algorithm has been measured on a quad core Linux workstation. The processing times taken by different clustering algorithms to cluster different data sets are reported in Table 3.3. The time shown here for the proposed algorithm is the sum of the times taken to estimate the value of θ, to build the baseline clusters, and to perform the k-means clustering algorithm to merge the remaining singleton clusters to the baseline clusters. The time shown for CUES is the sum of the times taken to estimate the value of θ, to build the similarity matrix, and to perform clustering. The time shown for bisecting k-means, buckshot, k-means and NMF are the average of the processing times of 10 iterations. It is to be mentioned that the codes for all the algorithms are written in C++ and the data structures for all the algorithms are developed by the authors. Hence the processing time can be reduced by incorporating some more efficient data structures for the proposed algorithm as well as the other methods. Note that the processing time of the proposed algorithm is less than KM, SLHC, ALHC and CUES for each data set. The execution time of BKM is less than the proposed one for 20ns, the execution time SCK is less than the proposed one for tr41 and the execution time of NMF is less than the proposed algorithm for tr31. The execution time of the proposed algorithm is less than BKM, SCK and NMF for each of the other data sets. The

processing time of the proposed algorithm is comparable with buckshot (though in most of the data sets the processing time of the proposed algorithm is less than buckshot). The dimensionality of the data sets (used in the experiments) varies from 2000 (fbis) to 35218 (20ns). Hence the proposed clustering algorithm may be useful in terms of processing time for any real life corpus with high dimensionality.

3.4 Conclusions

A hybrid document clustering algorithm, which is a combination of a hierarchical and k-means clustering technique, is developed in this chapter. It uses the concept of extensive similarity stated in chapter 2. The baseline clusters produced by the hierarchical technique are the clusters where the documents possess high similarity among them. The extensive similarity between documents ensures this quality of the baseline clusters. It is developed on the basis of similarity between two documents and their distances with every other documents in the document collection. Thus the documents with high extensive similarity are grouped in the same cluster. Most of the singleton clusters are nothing but the documents which have low content similarity with every other document. In practice the number of such singleton clusters is sufficient and can not be ignored as outliers. Therefore k-means algorithm is performed iteratively to assign these singleton clusters to one of the baseline clusters. Thus the proposed method reduces the error of k-means algorithm due to random seed selection. Moreover the method is not as expensive as the hierarchical clustering algorithm which can be observed from Table 3.3. The significant characteristic of the proposed clustering technique is that the algorithm automatically decides the number of clusters in the data. The automatic detection of number of clusters for a sparse and high dimensional text data is very important. The value for α is chosen heuristically and it is taken as $\alpha = \sqrt{\sqrt{N}}$ in the experiments, although it is found to provide good results. Extensive experimental results demonstrate the effectiveness of the proposed method.

Chapter 4

A Tweak on k-Nearest Neighbor Decision Rule for Text Categorization

4.1 Introduction

The task of kNN decision rule is to assign a test document x to a particular category using a training sample set. It first finds the k-nearest neighbors from the training sample set by a distance function and assigns x to a particular category by taking a majority vote among the k-nearest neighbors. The performance of the k-nearest neighbor decision rule depends heavily upon the value of the neighborhood parameter k. Different values of k can change the result of text categorization and hence choice of k is crucial for proper categorization. The cross validation technique is generally used to estimate an optimal value of k [42], but choosing an optimal k which provides satisfactory results for all test documents is still a difficult job. The cross-validation method uses the training data to select a single value of k, and then that selected value is used for categorization of all the test documents.

In kNN decision rule one may put a test document into a category which has a win by one vote to the next competing category. A document may also be arbitrarily assigned to a category if there is a tie between two competing categories, i.e., if the number of members of the competing categories among the nearest neighbors are equal. Text categorization is a very challenging task due to the large number of terms present in every document. In text categorization, a test document is assigned to a

particular category, if it has some content similarity with some of the documents belonging to that category. The content similarity is generally measured by the cosine similarity using the vector space model [111]. Since each document contains many non informative features, the measure used for finding content similarity between two documents may not reflect the actual similarity of the documents. Thus text categorization using kNN rule may not be accurate when the difference between the number of members (among the neighbors) of the competing categories is one or zero. Consider the example of an email data set consisting of two categories, spam and non spam email. Now the task is to find the category of a new email. Using kNN rule, if there are same number of members of the two categories among the nearest neighbors then the new document is assigned either to the spam category or to the non spam category. It may cause a serious harm if the new email is actually a spam email and it is assigned to the non spam category. On the other hand, if the non spam email is categorized to the spam category in this way, then one may loose some important content (as we generally remove the spam emails). Though this is an extreme example, but the decision using kNN rule may sometimes degrade the quality of text categorization. If the difference between the number of members of the competing categories is considered to be more than one then it may enhance the confidence of the majority voting by kNN rule. For any pattern classification problem too, the more the number of points in the majority class, the more the confidence on the choice of decision.

In this chapter a tweak on the kNN decision rule is introduced for text categorization, which is a two stage process. In the first stage it may reduce the actual search space of the kNN rule. The neighbors of a test document is ordered in such a way that they have at least a minimum amount of content similarity with the test document. The content similarity is determined by a predefined threshold θ. A set of neighbors is developed by incorporating the documents, whose content similarity is greater than θ. In the second stage a discrimination criterion on the voting process of kNN restricts the majority voting of kNN by a predefined positive integer threshold, say β, to assign a document to a category. The proposed method need not require a fixed k prior to categorizing a document. The method starts with an initial value of k as β. If the difference between the number of members of the best and the second best competing categories is β, then the document is categorized to the best competing category. Otherwise the value of the neighborhood parameter k is increased by one. The process continues till a decision is made or it reaches the

last document of the set of neighbors. If the test document is not categorized till the process traverses all the documents of the set of neighbors, then the test document remains unclassified. Consider again the example of email data set in this context. If the new email is not able to satisfy the said criterion of the proposed method then it remains unclassified. From a practical view point, when we are not sure about a decision, it is better not to take the decision, rather than deciding arbitrarily. The main objective of the proposed decision rule is to enhance the confidence on the decision making. The material presented in this chapter is taken from [15, 16].

The rest of this chapter is organized as follows, section 4.2 describes the proposed tweak on kNN decision rule for text categorization. The experimental evaluation on several text data sets is given in section 4.3. Conclusions and discussion are presented in section 4.4.

4.2 Proposed Tweak on kNN Decision Rule for Text Categorization

A *Tweak on kNN* (TkNN) decision rule has been proposed for text categorization. The proposed methodology is a two stage process. In the first stage the method ensures that each neighbor has at least some content similarity with the test document, since all the documents in the training sample set may not be required in determining the category of a particular document. For this purpose, it considers only those documents whose similarity with the test document is at least equal to a predefined threshold $\theta > 0$. The obtained set of documents is named as Set of Neighbors (SN). Note that kNN rule considers all the documents in the training set as neighbors for a test document. In the proposed method, every document in the training set is not considered to be a neighbor of the test document. Let d_0 be the test document and $D = \{d_1, d_2, ..., d_N\}$ be the set of N training documents. Then SN is developed in the following way.

$$SN = \{d \in D : \quad cos(\vec{d_0}, \vec{d}) > \theta\} \tag{4.1}$$

Here $\vec{d_0}$ and $\vec{d}$ are the corresponding vectors of documents d_0 and d respectively. After finding SN, the documents in it are sorted in decreasing order of similarity.

The categorization procedure is stated in the second stage. Algorithm 3 describes

in detail the proposed TkNN decision rule for categorizing a particular test document using reduced search space.

In the second stage, the first L documents from SN are considered to obtain the decision. Let the two categories in which the maximum and the second maximum number of neighbors among the L documents lie be denoted by (C_{x1}) and (C_{x2}). Let the number of neighbors in these two categories among L documents be denoted by (Z_{x_1} and Z_{x_2}), where $Z_{x_1} \geq Z_{x_2}$. Let β be a chosen positive integer. The value of L ranges from β to $|SN|$. If $Z_{x_1} - Z_{x_2} = \beta$ then the test document is labeled with that particular category C_{x1}. Otherwise L is increased by one and the same procedure is continued until it has reached the last document of SN. The algorithm never categorizes a document to a particular category, if $(Z_{x_1} - Z_{x_2}) < \beta$ and it reaches the last document of SN.

Consider the following example for $\beta = 2$. Let there be two categories C_1 and C_2, and there are 11 documents in SN and the categories of those documents are ordered as

$$C_1, C_2, C_1, C_2, C_2, C_1, C_1, C_2, C_1, C_2, C_1$$

Initially $Z_{x_1} = 1$ and $Z_{x_2} = 1$ and $(Z_{x_1} - Z_{x_2} < 2)$. Therefore L has to be increased by one to execute the same operation and so on. Thus, at the end $Z_{x_1} = 6$ and $Z_{x_2} = 5$, which concludes that the test document remains unclassified. But using kNN, for $k = 3$, the document is categorized to C_1, for $k = 4$ there is a tie and for $k = 5$, the document is categorized to C_2 and so on. It reveals the fact that simple majority voting may not be appropriate. Basically, when there is more or less same representation from the competing categories among the neighbors, we believe that the test document should not be put into one of those categories. Intuitively, the likelihood of correct categorization is more for the proposed scheme than kNN. If TkNN is able to categorize a document then it has sufficiently many members among the neighbors in support of the corresponding category, otherwise the document remains unclassified. The flowchart of the proposed decision rule for text categorization is given in figure 4.1.

Remarks:

- In the first stage, by reducing the search space, we are interested in ensuring sufficient content similarity between the test document and its neighbors. On

Algorithm 3 Tweak on kNN Decision Rule for Text Categorization

Input: a) $D = \{d_1, d_2, ..., d_N\}$ be the set of N training documents.
b) A set of m predefined categories, $C = \{C_1, C_2, ..., C_m\}$
c) Let d_0 is a particular test document, β is the threshold for majority voting and θ be the document similarity threshold.

Steps of the Algorithm:

for j $\leftarrow$ 1 to N **do**
  $dist \leftarrow cos(\vec{d_0}, \vec{d_j})$
  if $dist > \theta$ **then**
    $S[j] \leftarrow dist$
  end if
end for
$L \leftarrow \beta$
$SN \leftarrow$ Sort S in decreasing order and arrange D accordingly
$SN_0 \leftarrow$ First L documents from SN
for $i \leftarrow 1$ to m **do**
  $Z_i \leftarrow$ Number of documents in $SN_0 \in C_i$
end for
$Cat(d_0) \leftarrow \emptyset$
while $L \neq |SN|$ **do**
  $Z_x \leftarrow \max(Z_1, Z_2, ..., Z_m)$
  $Z_y \leftarrow \max\big(\{Z_1, Z_2, ..., Z_m\} - \{Z_x\}\big)$
  if $(Z_x - Z_y) = \beta$ **then**
    $Cat(d_0) \leftarrow C_x$ // i.e., d_0 is categorized to C_x
    return $Cat(d_0)$
  else
    $L \leftarrow L + 1$
    if $SN[L] \in C_i, \quad i = 1, 2, ..., m$ **then**
      $Z_i \leftarrow Z_i + 1$
    end if
  end if
end while
return $Cat(d_0)$ // i.e., the method does not categorize d_0

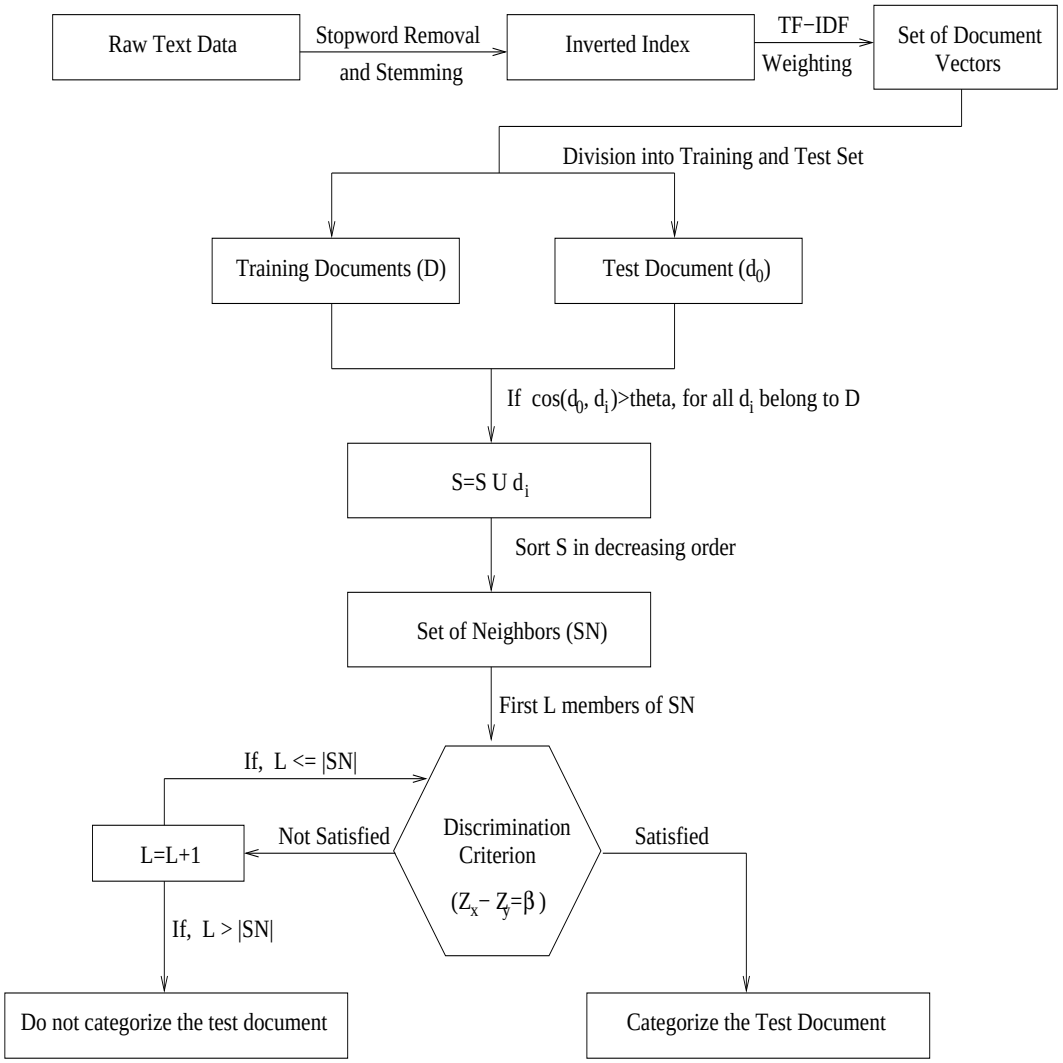

Figure 4.1: A Tweak on k-Nearest Neighbor Decision Rule for Text Categorization

the other hand a significant difference between the number of documents of majority category and its competing categories is enhancing the confidence on the decision process.

- It may be noted that the whole procedure has two parameters, namely θ and β. For high values of θ, SN may become an empty set. Similarly, for high values of β, many documents may remain unclassified. The parameters need to be set appropriately.

4.3 Experimental Evaluation

The experimental results on various text data sets are presented in this section to compare the performance of the proposed TkNN decision rule with traditional kNN, three other variants of kNN, naive bayes (NB) and SMV classifiers.

4.3.1 Parameter Settings

The text categorization using kNN [33], weighted kNN [43] and adaptive kNN [8] desicion rule categorizes a document on the basis of k nearest neighbors. Note that the value of k can vary from one to the total number of documents in the training set, but k must be fixed for all test documents. If k is very high then the computational complexity of these decision rules become high, which is not desirable. Hence k should not be very high. The proposed decision rule reduces to the one nearest neighbor (1-NN) method, or simply nearest neighbor decision rule, when $\beta = 1$. Hence in the experimental evaluation 1-NN is not used for comparison. Here 10 fold cross validation is performed on the training set by varying k from 2 to 15. The k value for which each of kNN, weighted kNN and adaptive kNN decision rule have best performance among these 14 values is used in the experiments of the test documents. The value of the parameter α has been taken as $\alpha = 5$ for the adaptive kNN technique. In the experiments, the r values of kNN model [59] are varied from 1 to 10 and cosine similarity is used as the similarity measure. The traditional kNN decision rule is implemented using the default package available in Matlab for kNN classifier. The codes to implement adaptive kNN, weighted kNN decision rules and kNN model for text categorization have been developed by the author as no such codes are available in public domain on any reliable source. The code for the proposed TkNN decision rule for text categorization has been developed by the author.

TkNN is dependent on the parameter β. A high β value can produce a large number of unclassified documents when the size of the training set is not very high. Hence the values of β have been restricted to 2, 3, 4 for comparison with the other methods [15]. Here also 10 fold cross validation is performed on the training sets using $\beta = 2, 3, 4$. The β values for which TkNN has best performance are used in the experiments. The value of θ is taken as 0.05 for all the data sets. The code for the proposed ESDR has been developed by the author.

SVM is performed using the libsvm[1] tool developed by C. C. Chang and C. J. Lin [26]. In the experiments the linear kernel is used [135] and the other parameters of SVM remain the same as provided by the default option of libsvm tool. Naive bayes is implemented using the mallet[2] tool developed by A. McCallum [90].

For all the methods including naive bayes and support vector machine, 10 fold cross validation is performed on the entire data set to split the data into training and test sets. All the decision rules for text categorization are executed for 10 times to reduce the effect of random selection of the documents by cross validation. The mean of the accuracies along with their standard deviations and f-measure values of 10 executions are reported in Table 4.1 and Table 4.2 respectively for each data set. The characteristics of various document collections used in our experiments are summarized in Table 1.1.

4.3.2 Analysis of Results

The proposed TkNN algorithm is compared with traditional kNN [33], weighted kNN [43], adaptive kNN [8], kNN model [58], support vector machine [71] and naive bayes [89] methods using all the data sets described in section 1.5. Table 4.1 and Table 4.2 show the performances of these methods using accuracy and f-measure respectively on all the text corpora. The accuracy (ac) of TkNN is determined as, $ac = \frac{cc}{tp - up}$, where cc is the number of correctly categorized documents, tp is the number of documents of the test set, and up is the number of documents remaining unclassified. The average of L values for all the test documents is shown in a separate column for each data set. Note that L_{avg} indicates the L^{th} member of SN at which the test document has been assigned to a category by TkNN. The value of k (neighborhood size), which provides best accuracy among $k = 2, ..., 15$ for kNN, weighted kNN and adaptive kNN is reported in separate column for each of these three methods and for each data set. The r values of kNN model are also reported in a separate column. The text data sets are represented using vector space model [111].

Table 4.1 shows the categorization accuracies along with standard deviations of seven methods on fourteen data sets. For each data set TkNN is compared with five other decision rules. Thus, in total, 84 comparisons are made for TkNN. Out of

[1]http://www.csie.ntu.edu.tw/∼cjlin/libsvm/
[2]http://mallet.cs.umass.edu/

Table 4.1: Performance of Different Classifiers on Various Text Data Sets using Accuracy (in %)

Data Set	TkNN (Proposed)			kNN		Weighted kNN		Adaptive kNN		kNN Model		Naive Bayes	SVM
	β	L_{avg}	AC[3]	β	AC	k	AC	k	AC	k	AC	AC	AC
20ns	2	3	77.61 (0.09)	2	**79.53** (0.15)	14	73.13 (0.01)	4	75.58 (0.09)	7	73.68 (0.28)	78.31 (0.13)	77.86 (0.22)
fbis	3	7	79.19 (0.24)	7	79.08 (0.20)	15	**81.38** (0.19)	13	79.06 (0.19)	6	69.56 (0.27)	78.34 (0.30)	77.84 (0.25)
la1	4	24	**92.36** (0.26)	15	82.86 (0.23)	15	84.18 (0.24)	15	82.23 (0.23)	9	85.91 (0.44)	88.03 (0.17)	90.84 (0.14)
la2	2	3	82.08 (0.10)	3	81.98 (0.27)	9	83.64 (0.04)	3	80.82 (0.29)	9	81.29 (0.31)	89.14 (0.21)	**89.27** (0.23)
oh10	4	6	**77.32** (0.81)	11	72.63 (0.57)	15	73.37 (0.43)	15	72.60 (0.46)	6	74.72 (0.41)	76.71 (0.32)	75.43 (0.26)
oh15	4	11	**82.12** (0.43)	15	79.81 (0.48)	15	76.48 (0.64)	15	78.97 (0.29)	7	80.65 (0.32)	81.68 (0.48)	81.58 (0.36)
rcv1	4	7	**88.90** (0.40)	3	86.84 (0.31)	9	87.18 (0.29)	8	86.58 (0.32)	7	85.31 (0.47)	88.42 (0.26)	87.79 (0.1)
rcv2	3	7	**89.15** (0.17)	3	86.96 (0.19)	15	88.59 (0.22)	11	86.80 (0.36)	8	85.59 (0.16)	88.28 (0.19)	88.87 (0.16)
rcv3	3	6	**89.12** (0.26)	5	87.10 (0.19)	11	88.50 (0.21)	7	86.32 (0.23)	8	87.12 (0.47)	88.00 (0.20)	88.67 (0.09)
rcv4	4	8	**90.11** (0.15)	3	87.28 (0.18)	5	88.76 (0.24)	5	86.82 (0.26)	8	86.82 (0.28)	88.49 (0.21)	89.31 (0.07)
tr31	2	3	92.59 (0.31)	3	93.05 (0.28)	5	**94.77** (0.30)	3	92.14 (0.30)	5	91.56 (0.37)	92.56 (0.39)	92.16 (0.36)
tr41	4	5	**95.51** (0.35)	5	92.32 (0.34)	8	93.24 (0.24)	5	91.82 (0.40)	3	91.68 (0.32)	93.47 (0.23)	93.85 (0.29)
tr45	2	3	88.79 (0.71)	3	88.98 (0.32)	7	**89.46** (0.42)	3	87.71 (0.091)	5	87.56 (0.40)	85.13 (0.31)	88.76 (0.50)
wap	4	5	**84.02** (0.38)	14	74.37 (0.37)	15	73.59 (0.52)	15	73.74 (0.44)	9	75.56 (0.46)	79.22 (0.29)	78.30 (0.21)

[3]AC stands for accuracy. The mean of the accuracy are shown in % for 10 iterations along with the standard deviation (within ()) for each method.

Table 4.2: Performance of Different Classifiers on Various Text Data Sets using F-measure

Data Set	TkNN (Proposed)			kNN		Weighted kNN		Adaptive kNN		kNN Model		Naive Bayes	SVM
Set	β	L_{avg}	FM[4]	β	FM	k	FM	k	FM	k	FM	FM	FM
20ns	3	8	0.770	2	**0.789**	14	0.726	4	0.752	7	0.729	0.778	0.774
fbis	4	6	0.777	3	0.773	15	**0.802**	12	0.782	6	0.690	0.780	0.776
la1	4	9	**0.916**	3	0.793	11	0.822	3	0.792	9	0.801	0.886	0.892
la2	3	9	0.818	4	0.814	9	0.834	3	0.801	9	0.806	**0.887**	**0.887**
oh10	4	6	**0.765**	11	0.721	15	0.729	15	0.721	6	0.742	0.760	0.748
oh15	4	6	**0.818**	14	0.792	15	0.760	15	0.783	6	0.802	0.807	0.807
rcv1	4	6	**0.884**	3	0.858	9	0.867	8	0.860	7	0.848	0.879	0.872
rcv2	4	8	**0.887**	3	0.860	14	0.880	11	0.860	8	0.849	0.877	0.884
rcv3	4	7	**0.886**	6	0.860	11	0.878	7	0.857	6	0.865	0.875	0.881
rcv4	4	7	**0.896**	3	0.864	5	0.881	6	0.861	8	0.863	0.877	0.886
tr31	4	7	0.920	3	0.925	6	**0.942**	3	0.917	5	0.908	0.920	0.916
tr41	3	5	**0.948**	5	0.918	9	0.928	5	0.916	3	0.912	0.930	0.932
tr45	2	4	0.881	3	0.884	7	**0.888**	3	0.872	5	0.871	0.849	0.882
wap	3	7	**0.835**	14	0.742	15	0.730	15	0.737	9	0.748	0.789	0.784

[4] FM stands for f-measure.

these 84 comparisons, TkNN is found to work better in 73 cases. The other methods are found to work better in the rest 11 cases only, out of 84. This indicates the value and validity of TkNN. The methods which perform better than TkNN in some data sets are e.g., naive bayes for la2 (the accuracy is 82.08% for TkNN and 89.14% for naive bayes), weighted kNN for tr45 (the accuracy is 88.79% for TkNN and 89.46% for weighted kNN).

The statistical significance test as described in chapter 2 has been performed to check whether these differences are significant e.g., whether 89.46 (TkNN) is significantly different from 88.79 (weighted kNN) for tr45. When TkNN performs better than the other decision rules, it has been checked whether the differences are significant. It has been found that the results are significant in 65 out of 73 cases, where TkNN performs better than the other methods for the level of significance 0.05. The test results are significant in 9 out of 11 cases when other methods have an edge over TkNN. Thus in 87.83% cases the performance of TkNN is significantly better than the other methods.

It can be seen from Table 4.2 that the proposed decision rule is compared with six other classifiers using f-measure for fourteen data sets. A total of 84 comparisons have been made for TkNN in this table. The f-measure values of TkNN is better

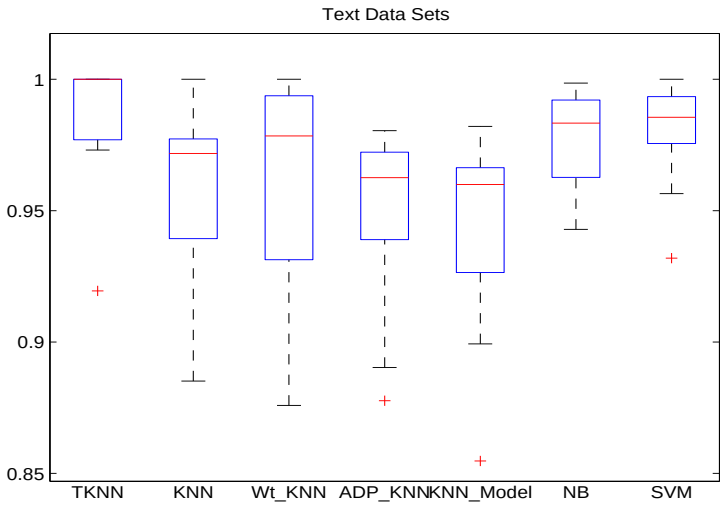

Figure 4.2: Robustness of Different Decision Rules

than the other methods in 80 out of these 84 comparisons. The other classifiers are found to work better than TkNN in the rest 14 cases. It has been found using t-test that the results are significant in 58 out of 70 cases where TkNN performs better than the other methods for the level of significance 0.05. Out of the rest 14 cases when other methods performed better than TkNN 9 results are found to be significant for the same level of significance. Thus the performance of TkNN is significantly better than the other methods in 86.56% cases.

The robustness of different nearest neighbor decision rules on different data sets can be determined by using the idea of Friedman [51,56]. Robustness of a particular decision rule in a particular example is defined by the misclassification probability ratio, $mc_i = ac_i/ac_0$, where ac_i is the accuracy of the i^{th} decision rule and $ac_0 = \max_i ac_i$. In any particular example the best decision rule will have $mc_i = 1$, while the other decision rules will have $mc_i \leq 1$. Lower values of mc_i indicate the lack of robustness of the i^{th} decision rule. For all the text data sets this ratio is computed for all the decision rules, and they are graphically shown by boxplots in Figure 4.2. The figure shows the superiority of TkNN for text categorization.

4.3.3 Time and Space Complexity of TkNN

$O(N)$ memory locations are needed to store N documents of the training set. Therefore the space complexity of TkNN is $O(N)$. $O(N)$ time is required to find cosine

Table 4.3: Processing Time (in seconds) of Different Classifiers

Data Sets	TkNN Proposed)	kNN	Weighted kNN	Adaptive kNN	kNN Model	Naive Bayes	SVM
20ns	**58.67**	117.02	287.72	124.63	1036.36	94.96	102.35
fbis	**5.57**	18.52	68.50	15.59	118.12	6.92	7.56
la1	**1.89**	8.82	9.72	7.14	89.32	4.69	6.24
la2	**0.67**	1.80	1.95	1.49	76.21	4.05	6.55
oh10	**0.84**	8.27	12.17	4.01	61.59	2.30	3.12
oh15	**0.54**	7.09	10.89	3.53	51.66	2.43	2.54
rcv1	**1.30**	3.25	8.38	3.36	63.21	5.63	6.25
rcv2	**1.36**	4.36	12.63	5.92	70.56	6.20	7.49
rcv3	**1.68**	4.28	12.45	5.84	69.25	6.36	7.32
rcv4	**1.75**	4.45	12.42	5.68	72.75	6.35	7.54
tr31	**0.88**	3.40	4.13	2.55	131.20	3.35	3.47
tr41	**0.63**	6.68	10.33	3.26	106.97	3.18	3.55
tr45	**0.49**	5.37	8.43	2.57	85.55	3.64	4.25
wap	**1.18**	52.79	158.66	10.65	279.08	5.01	5.53

similarity between the test document and N documents of the training set. Let us assume y number of documents have been discarded from the training set to develop SN, which takes $O(N-y)\log((N-y)$ time. The rest of the steps take at most $O(N-y)$ time to categorize a test document. Thus the time complexity of TkNN is $O(N)+O(N-y)\log((N-y)$. The worst case time complexity of TkNN is $O(N)\log((N)$ when y is very close to N, that is when the size of SN is close to N.

Table 4.3 summarizes the time taken by different decision rules to categorize the text data sets. For the proposed method SN is created in the first stage and then TkNN rule is performed on the SN in the second stage for each test document. The time shown in Table 4.3 for each data set is the sum of the times taken by the two stages of TkNN rule on all the test documents. Similarly for the other methods also the time shown for each data set is the sum of the processing times of all the test documents. It can be seen from Table 4.3 that the processing time of the proposed TkNN rule is the least among all the methods for all the data sets.

4.4 Conclusions and Discussion

The k-nearest neighbor decision rule is a simple and robust decision rule for categorization in statistical pattern recognition and it is widely used for text categorization.

One major issue with the kNN rule is that the majority voting may not be convincing, if the margin of vote between two competing categories is very small. The other limitation is that no good criterion is available to select an optimal value for neighborhood parameter k. These two issues have been addressed here, and an improved solution has been provided.

The proposed method chooses those documents from the training set that have high content similarity (greater than a predefined threshold θ) with the test document and constructs a Set of Neighbors (SN) with the rest of the documents. The test document is then assigned to the category whose number of members (among SN) is greater than a threshold (β) than the next competing category. This condition is tested until a decision is made or it reaches the last neighbor. Therefore no prior knowledge about neighborhood parameter k is required for the TkNN rule. A document is not categorized by TkNN rule, if it fails to fulfill the said criterion after searching the entire SN and thus enhances the confidence of the majority voting.

TkNN rule has two parameters. The parameter θ builds SN and the parameter β is used in majority voting. The experimental results suggest that one of the values 2, 3, 4 of β would provide better results for text categorization. θ is dependent on the nature of the corpus. Practically, the vocabulary size becomes very large in comparison with the number of training documents, and the data matrix is generally sparse. Therefore θ should not be very high. Most of the similarity values of all pair of documents of each corpus used in the experiments lie between 0.05 and 0.1. Hence $\theta = 0.05$ is selected in the experiments. Note that for large θ the size of SN may become very small, and this is not desirable to train a classifier. But for a dense term-document matrix (with a low vocabulary size), where most of the terms in the vocabulary carry some information for every document, a high value of θ may be chosen. It has been observed from the experiments that $|SN| << N$ for each corpus. Note that the value of L is very low (at most one fifth of the size of SN) for every data set. That is, the number of neighbors traversed by TkNN rule for a decision is very small compared to the size of SN for each data set. Hence, the computational time of TkNN decision rule is the least among all the other classifiers.

Chapter 5

An Extensive Similarity based Decision Rule for Text Categorization

5.1 Introduction

In the previous chapter, a tweak on kNN has proposed, and it has been found to perform well on several data sets. It is stated that kNN or the proposed tweak on kNN (TkNN) perform text categorization by observing the documents of the training set which are most similar to it. Similarity measure plays a significant role in deciding the category of a document for kNN or TkNN. In this chapter a new similarity based decision rule has been proposed for text categorization by combining the idea of TkNN [16] decision rule and the extensive similarity measure proposed in chapter 2.

The text data sets are generally represented by vector space model [89], where cosine similarity is used to determine the content similarity between the documents. It may be noted that extensive similarity has been proposed to overcome some limitations of cosine similarity (section 2.1) for effective clustering of the documents. The extensive similarity measure is used in this chapter for supervised learning. The extensive similarity used here for text categorization considers the documents of the training set alone for computing similarity, whereas the one proposed in chapter 2 considers all the documents of the corpus. The extensive similarity between a test document and a training document is determined after extensively checking the

distances of the test document with all the documents of the training set.

The proposed decision rule initially finds the extensive similarity of a test document with all the documents of the training set. An ordered set of neighbors is developed from the training set by including those documents, which have non-negative extensive similarity with the test document. Thus the proposed method reduces the search space of majority voting, which is essential for large scale text data sets. Then all the categories are ordered based on the number of similar documents with the test document among the set of neighbors. The test document is assigned to the best one among the competing categories, if the difference of the number of similar documents between the best category and the second best category is greater than a predefined positive value. The method terminates when it reaches the last document of the set of neighbors, or it satisfies the said condition. The proposed decision rule for text categorization is compared with adaptive kNN, weighted kNN, naive bayes and support vector machine classifiers using several well known data sets. It has been observed from the empirical studies that the proposed method performs significantly better than the other classifiers for text categorization. The material of this chapter can be found in article [12].

The chapter is organized as follows - section 5.2 describes the proposed extensive similarity measure for text categorization. Section 5.3 presents the proposed extensive similarity based decision rule for text categorization. The experimental evaluation on several text data sets is discussed in section 5.4. The conclusions about the proposed work are presented in chapter 5.5.

5.2 Extensive Similarity for Text Categorization

A new similarity measure, *extensive similarity* has been proposed in chapter 2 to find the content similarity between two documents for effective document clustering. Extensive similarity checks all the documents of the corpus in determining the similarity of any two documents. If two documents share a certain number of common terms and they have almost same distances with every other document in the corpus (i.e., either both are similar or both are dissimilar to all the other documents) then the documents are said to be exactly similar by the extensive similarity. In this chapter extensive similarity has been used for text categorization. It checks the documents of the training set instead of the entire corpus. The content similarity between a test document d_0 and a training document $d_i,\ \forall i = 1, 2, ..., N$ is determined

by a threshold $\theta \in (0, 1)$ using the following distance function

$$dis(d_0, d_i) = \begin{cases} 1 & \text{if } cos(\vec{d_0}, \vec{d_i}) \leq \theta \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Here $\vec{d_0}$ and $\vec{d_i}$ are the vectors corresponding to the documents d_0 and d_i. Two documents are dissimilar if distance is 1, i.e., they share few number of common terms. On the other hand, distance 0 indicates that there exists some content similarity between documents d_0 and d_i, i.e., they have a minimum number of terms in common. θ is a fixed threshold on cosine similarity which restricts the low similarity values. The value of θ is determined by the same histogram thresholding method discussed in chapter 2.

The extensive similarity between the test document d_0 and the training document d_i is defined in this chapter based on their distances with every other document in the training set. Let $l_{0i} = \sum_{k=1}^{N} |dis(d_0, d_k) - dis(d_i, d_k)|$, where N is the number of documents in the training set. Here l_{0i} denotes the number of documents among the documents of the training set, where similarity with d_0 is not the same as the similarity with d_i. The similarity between the documents d_0 and d_i decreases as the l_{0i} value increases. The documents are said to be totally similar when $l_{0i} = 0$. Basically l_{0i} is a grade of dissimilarity and it indicates that any two documents d_0 and d_i have different distances with l_{0i} number of documents in the training set. The extensive similarity (ES) between d_0 and d_i is defined as

$$ES(d_0, d_i) = \begin{cases} N - l_{0i} & \text{if } dis(d_0, d_i) = 0 \\ -1 & \text{otherwise} \end{cases} \tag{5.2}$$

In this extensive similarity, two documents are more similar than any other pair of documents, if they have a very high ES value than the other pairs. The main significance of extensive similarity is the negative similarity value. If two documents have negative extensive similarity then the documents are no more similar. In the next section we shall observe that this negative similarity value will help to reduce the number of participation in majority voting to determine the category of a document. The extensive similarity measure for text categorization is described in Algorithm 4. It explains how the function extensive similarity finds the similarity between a test document and a set of training documents, where dis_trn is the set of distances

Algorithm 4 extensive_similarity(dis_trn, dis_tst, N)

```
for j ← 1 to N do
  if dis_tst[j] = 1 then
    ES[d_0, d_j] ← −1
  else
    l ← 0
    for i ← 1 to N do
      l ← l + |dis_tst[j] − dis_trn[i][j]|
    end for
    ES[d_0, d_j] ← N − l
  end if
end for
return ES
```

between every pair of documents of the training set following equation 5.1 and dis_tst is the set of distances between the test document and every document of the training set following the same equation.

5.3 Framework of the Extensive Similarity Based Decision Rule for Text Categorization

Generally the similarity based classifier e.g., kNN decision rule categorize a document to a particular category based on the majority voting among the competing categories using the nearest neighbors. Basu et al. [16] proposed a tweak on the kNN classifier which enhances the confidence of decision making. A test data point has been grouped to a predefined category by TkNN, if the difference between the number of members of two competing categories is equal to a predefined positive integer threshold $\beta > 1$, otherwise the method continues to check the entire neighborhood until the condition is satisfied. They have shown that for $\beta = 2, 3, 4$ the method has performed well for several benchmark and artificial data sets. The proposed decision rule for text categorization has been developed using the idea of the discrimination criteria for majority voting of the TkNN decision rule and extensive similarity between the test document and the training documents. The proposed decision rule is named as *E*xtensive *S*imilarity based *D*ecision *R*ule (ESDR) and the steps of the method is described in Algorithm 5.

The proposed ESDR for text categorization finds the extensive similarity of a test

Algorithm 5 Procedure of ESDR to Categorize a Test Document

Input: a) D = $\{d_1, d_2, ..., d_N\}$ be the set of N training documents.
b) $C = \{C_1, C_2, ..., C_m\}$, set of m predefined categories
c) d_0 be the test document, β is a threshold on majority voting and θ is the document similarity threshold.

Steps of the Algorithm:

$Sim\_trn[i][j] \leftarrow \cos(\vec{d_i}, \vec{d_j}), \forall\ i, j \leftarrow 1, ..., N$ and $i \neq j$
Build $dis\_trn[i][j]$ from Equation 5.1, $\forall\ i, j \leftarrow 1, ..., N$ and $i \neq j$
$Sim\_tst[j] \leftarrow \cos(\vec{d_0}, \vec{d_j}), \forall\ j \leftarrow 1, ..., N$
Build $dis\_tst[j]$ from Equation 5.1, $\forall\ j \leftarrow 1, ..., N$
$ES \leftarrow$ extensive_similarity(dis_trn,dis_tst,N) // by using Algorithm 1
$Cat(d_0) \leftarrow \emptyset$; $loc \leftarrow 0$
for $j \leftarrow 1$ to N **do**
  if $ES[d_0, d_j] \geq 0$ **then**
    $SN[loc] \leftarrow ES[d_0, d_j]$
    $\widehat{SN}[loc] \leftarrow j$
    $loc \leftarrow loc + 1$
  end if
end for
Sort SN in decreasing order maintaining the index ($\widehat{SN}$)
$L \leftarrow \beta$
$SN_0 \leftarrow$ First L documents from SN
for $i \leftarrow 1$ to m **do**
  $Z_i \leftarrow$ Number of documents in $SN_0 \in C_i$
end for
while $L \neq |SN|$ **do**
  $Z \leftarrow \{Z_1, Z_2, ..., Z_m\}$ // Z is a multi set that may contain multiple equal value items
  $Z_x \leftarrow \max(Z)$; $Z_y \leftarrow \max\left(Z - \{Z_x\}\right)$
  if $(Z_x - Z_y) = \beta$ **then**
    $Cat(d_0) \leftarrow C_x$ // i.e., d_0 is categorized to C_x
    return $Cat(d_0)$
  else
    $L \leftarrow L + 1$
    if $SN[L] \in C_i,\ \ i = 1, 2, ..., m$ **then**
      $Z_i \leftarrow Z_i + 1$
    end if
  end if
end while
return $Cat(d_0)$ // i.e., the method does not categorize d_0

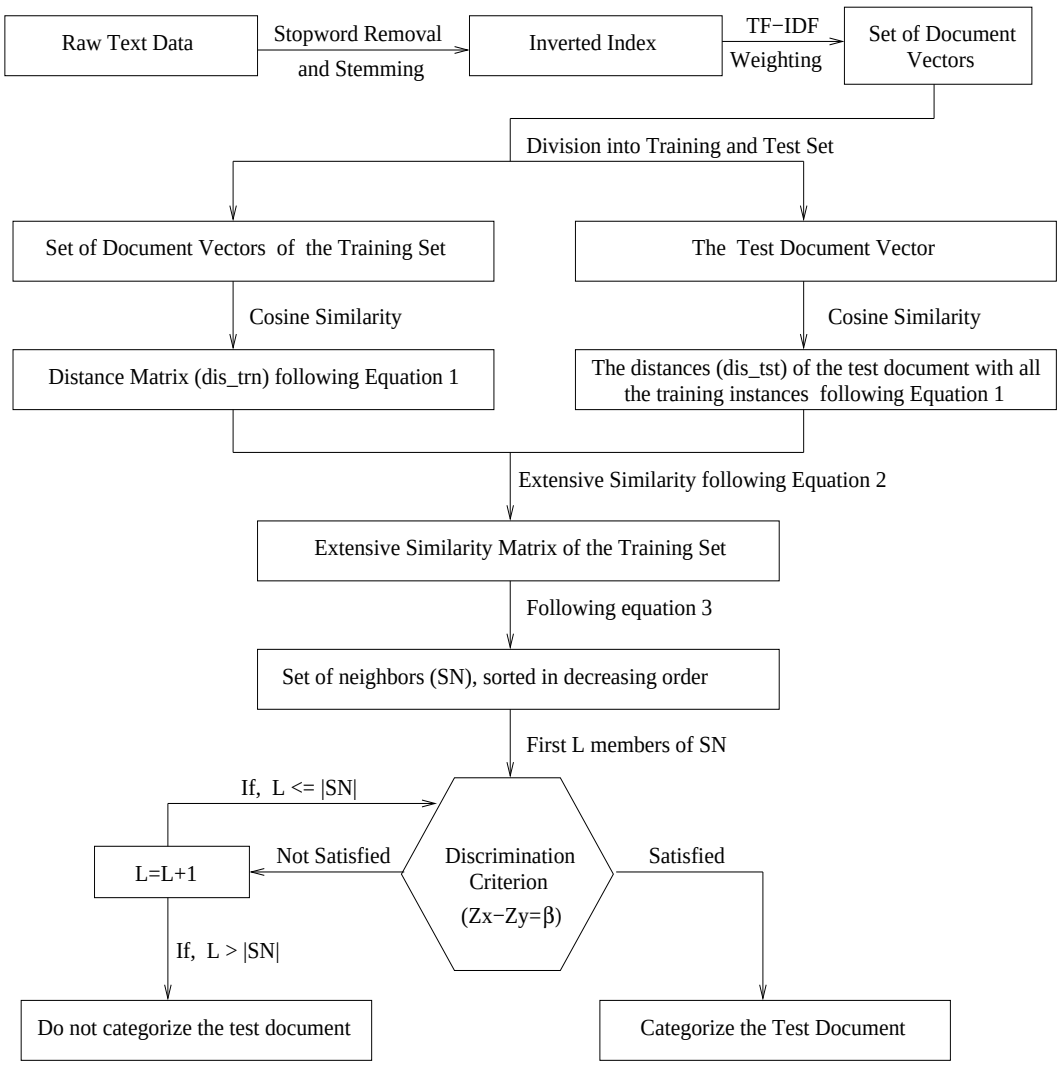

Figure 5.1: Text Categorization by Extensive Similarity based Decision Rule

document with all the documents in the training set. Let d_0 be the test document and $D = \{d_1, d_2, ..., d_N\}$ be the set of N training documents. A Set of Neighbors (SN) is developed by including the documents, which have non-negative extensive similarity with the test document as follows.

$$SN = \{d \in D : \quad ES(\vec{d_0}, \vec{d}) \geq 0\} \tag{5.3}$$

Thus the number of members for majority voting in the next step is reduced by using the extensive similarity measure. The documents in SN are sorted in decreasing order of their extensive similarity values. Then first β documents from

SN are taken to perform majority voting with a criterion that the difference between the number of documents of the competing two categories (Z_x and Z_y) is equal to β. If this condition is satisfied then the document is assigned to the best category C_x, having maximum number of documents in SN. Otherwise it checks the next document in SN and performs the same operation. The same procedure is continued until it has traversed the last document of SN. Algorithm 5 does not categorize a document, if $(Z_x - Z_y) < \beta$ and it reaches the last member of SN. In this case the proposed decision rule does not arbitrarily categorize this document to any category. It has become clear from this fact that the proposed decision rule never takes a decision when the supporting information is not so strong. The flowchart of the proposed technique is shown in figure 5.1.

5.4 Experimental Evaluation

Text categorization using kNN [33], weighted kNN [43] and adaptive kNN [8] rule classifies a document on the basis of k nearest neighbors of a test document. The value of k can vary from one to the number of documents in the training set, but k must be fixed for all test documents. Here 10 fold cross validation is performed on the training set by varying k from 2 to 15. The k value which provides best accuracy and f-measure among these 14 values is used in the experiments of the test documents for kNN, weighted kNN and adaptive kNN. The value of the parameter α has been taken as $\alpha = 5$ for the adaptive kNN technique. The traditional kNN decision rule is implemented using the default package available in Matlab for kNN classifier. The codes to implement adaptive kNN, weighted kNN and TkNN decision rules for text categorization has been developed by the author as no such codes are publicly on any reliable source. The code for the proposed ESDR has been developed by the author.

The proposed ESDR is dependent on the β values. The proposed decision rule will behave like the one nearest neighbor (1-NN) decision rule, or simply nearest neighbor classifier, when $\beta = 1$. Hence in the experimental evaluation 1-NN is not used for comparison. A high β value can produce a large number of unclassified documents if the training set size is not very high. Hence the values of β have been restricted to 2, 3, 4 for comparison with the other methods [15]. Here also 10 fold cross validation is performed on the training sets using $\beta = 2, 3, 4$, like TkNN decision rule. The β value for which the proposed method gives best accuracy is used

in the experiment for each data set. The TkNN decision rule uses the same procedure for selecting β in the experiments as stated in section 4.3.1. It may be noted that cosine similarity is used as the similarity measure for TkNN, kNN, weighted kNN and adaptive kNN.

SVM is performed using the libsvm[1] tool developed by C. C. Chang and C. J. Lin [26]. In the experiments the linear kernel has been used [135] and the other parameters of SVM remain the same as provided by the default option of libsvm tool. Naive bayes is implemented using the mallet toolkit[2] [90].

For all the methods including SVM and naive bayes, 10 fold cross validation is performed on the entire data set to split the data into training and test sets. The classifiers are executed for 10 times to reduce the effect of random selection of the documents by cross validation. The mean of the accuracies along with their standard deviations and f-measure values of 10 executions are reported in Table 5.1 and Table 5.2 respectively for each data set. The histogram thresholding based technique stated in chapter 2 has been used in the experiments here for estimating the parameter θ of extensive similarity measure of ESDR. The technique for estimating the value of θ has been implemented exactly in the same way as it has been mentioned in section 2.3.4.

5.4.1 Analysis of Results

The proposed ESDR for text categorization is compared with traditional kNN [33], TkNN [16], weighted kNN [43], adaptive kNN [8], SVM [71] and naive bayes [89] classifiers using all the data sets described in section 1.5. Table 5.1 and Table 5.2 show the performances of these methods using accuracy along with standard deviations and f-measure respectively on all the text corpora. The accuracy(ac) of the proposed ESDR is determined as, $ac = cc/(ntd - ud)$, where cc is the number of correctly classified documents, ntd is the number of documents in the test set, and ud is the number of documents for which no category is assigned. The accuracy of TkNN is determined as the same way as described in section 4.3.2 and reported in Table 5.1. The average of L (L is the L^{th} member of SN where a decision is made by ESDR for a test document) for all the test documents is shown in a separate column for each data set as L_{avg} in Table 5.1 and Table 5.2. The value of k (neighborhood

[1] http://www.csie.ntu.edu.tw/∼cjlin/libsvm/
[2] http://mallet.cs.umass.edu/

Table 5.1: Performance of Different Classifiers on Various Text Data Sets using Accuracy (in %)

Data Set	ESDR (Proposed)			TkNN		kNN		Weighted kNN		Adaptive kNN		Naive Bayes	SVM
Set	β	L_{avg}	AC[3]	β	AC	k	AC	k	AC	k	AC	AC	AC
20ns	3	8	77.74 (0.12)	2	77.63 (0.09)	2	**79.52** (0.15)	14	73.15 (0.03)	4	75.58 (0.09)	78.30 (0.12)	77.85 (0.22)
fbis	4	6	81.23 (0.18)	7	79.17 (0.23)	3	79.06 (0.20)	15	**81.38** (0.19)	13	79.07 (0.20)	78.35 (0.30)	77.84 (0.24)
la1	4	9	**93.15** (0.19)	4	92.15 (0.26)	3	79.75 (0.20)	12	82.70 (0.17)	3	79.65 (0.26)	89.09 (0.30)	90.84 (0.14)
la2	3	9	**90.90** (0.24)	2	82.18 (0.10)	3	81.95 (0.18)	9	83.76 (0.16)	3	80.75 (0.21)	89.14 (0.20)	89.27 (0.23)
oh10	4	6	76.26 (0.21)	4	**77.16** (0.82)	11	72.64 (0.57)	15	73.38 (0.43)	15	72.62 (0.46)	76.70 (0.33)	75.45 (0.26)
oh15	4	6	81.69 (0.18)	4	**82.10** (0.45)	15	79.80 (0.48)	15	76.47 (0.63)	15	78.97 (0.29)	81.65 (0.48)	81.64 (0.37)
rcv1	4	6	**89.11** (0.21)	4	88.90 (0.41)	3	86.86 (0.32)	9	87.16 (0.29)	8	86.56 (0.31)	88.40 (0.25)	87.77 (0.08)
rcv2	4	8	**90.63** (0.28)	3	89.12 (0.17)	3	86.99 (0.19)	15	88.56 (0.24)	11	86.76 (0.36)	88.29 (0.19)	88.87 (0.16)
rcv3	4	7	**89.40** (0.15)	3	89.10 (0.26)	5	87.00 (0.19)	11	88.52 (0.22)	7	86.30 (0.23)	88.04 (0.20)	88.70 (0.07)
rcv4	4	7	**91.06** (0.29)	4	90.10 (0.14)	3	87.29 (0.19)	5	88.76 (0.24)	5	86.79 (0.28)	88.49 (0.21)	89.34 (0.07)
tr31	4	7	93.48 (0.24)	3	92.59 (0.30)	3	93.09 (0.28)	5	**94.75** (0.30)	3	92.17 (0.30)	92.56 (0.38)	92.11 (0.34)
tr41	3	5	93.55 (0.30)	4	**95.49** (0.36)	5	92.38 (0.34)	8	93.24 (0.24)	5	92.02 (0.42)	93.66 (0.23)	93.86 (0.27)
tr45	2	4	87.82 (0.31)	2	88.75 (0.70)	3	88.98 (0.32)	7	**89.42** (0.42)	3	87.72 (0.08)	85.45 (0.30)	88.71 (0.50)
wap	3	7	**84.57** (0.31)	4	84.14 (0.36)	14	74.73 (0.37)	15	73.59 (0.51)	15	74.15 (0.41)	79.56 (0.26)	79.15 (0.23)

[3]AC stands for accuracy. The mean of the accuracy are shown in % for 10 iterations along with the standard deviation (within ()) for each method.

Table 5.2: Performance of Different Classifiers on Various Text Data Sets using F-measure

Data Set	ESDR (Proposed)			TkNN		kNN		Weighted kNN		Adaptive kNN		Naive Bayes	SVM
Set	β	L_{avg}	FM[4]	β	FM	k	FM	k	FM	k	FM	FM	FM
20ns	3	8	0.773	2	0.770	2	**0.789**	14	0.726	4	0.752	0.778	0.774
fbis	4	6	0.788	7	0.777	3	0.773	15	**0.802**	12	0.782	0.780	0.776
la1	4	9	**0.925**	4	0.916	3	0.793	11	0.822	3	0.792	0.886	0.892
la2	3	9	**0.897**	2	0.818	4	0.814	9	0.834	3	0.801	0.887	0.887
oh10	4	6	0.757	4	**0.765**	11	0.721	15	0.729	15	0.721	0.760	0.748
oh15	4	6	0.810	4	**0.818**	14	0.792	15	0.760	15	0.783	0.807	0.807
rcv1	4	6	**0.886**	3	0.884	3	0.858	9	0.867	8	0.860	0.879	0.872
rcv2	4	8	**0.901**	3	0.887	3	0.860	14	0.880	11	0.860	0.877	0.884
rcv3	4	7	**0.886**	3	**0.886**	6	0.860	11	0.878	7	0.857	0.875	0.881
rcv4	4	7	**0.905**	4	0.896	3	0.864	5	0.881	6	0.861	0.877	0.886
tr31	4	7	0.927	3	0.920	3	0.925	6	**0.942**	3	0.917	0.920	0.916
tr41	3	5	0.930	4	**0.948**	5	0.918	9	0.928	5	0.916	0.930	0.932
tr45	2	4	0.872	2	0.881	3	0.884	7	**0.888**	3	0.872	0.849	0.882
wap	3	7	**0.840**	3	0.835	14	0.742	15	0.730	15	0.737	0.789	0.784

[4] FM stands for f-measure.

size) among $k = 2, ..., 15$ of kNN, weighted kNN and adaptive kNN for which the accuracy and f-measure is best is noted in separate column for each of these three methods.

The proposed ESDR is compared with six other classifiers for each data set in Table 5.1. Thus 84 comparisons have been made for ESDR in total. The proposed one is found to work better in 69 out of these 84 comparisons. The other classifiers are found to work better than ESDR in the rest 15 cases. These results indicate the effectiveness of ESDR. It can be observed from Table 5.1 that ESDR performs better than all other classifiers in most of the cases. There are some cases where other methods perform better than ESDR e.g., weighted kNN performs better than ESDR for fbis, and tr45 and SVM performs better than ESDR for 20ns and tr41.

A statistical significance test that has been performed in chapter 2 is used here to check the significance of the differences of accuracies between ESDR and any other competing algorithm. It has been found that the results are significant in 60 out of 69 cases where ESDR performs better than the other methods for the level of significance 0.05. Out of the rest 15 cases when other methods performed better than ESDR 11 results are found to be significant for the same level of significance. Thus the performance of ESDR is significantly better than the other methods in 84.50% cases.

Table 5.2 shows the comparison of the proposed technique with six other classifiers using f-measure for each data set. A total of 84 comparisons have been made for ESDR in this table. The f-measure values of ESDR is better than the other methods in 70 out of these 84 comparisons. The other classifiers are found to work better than ESDR in the rest 14 cases. It has been found using t-test that the results are significant in 58 out of 70 cases where ESDR performs better than the other methods for the level of significance 0.05. Out of the rest 14 cases when other methods performed better than ESDR 11 results have been found as significant for the same level of significance. Thus the performance of ESDR is significantly better than the other methods in 84.05% cases. These results show the value and validity of ESDR.

Note that a few number of test documents (maximum 0.5% of the total documents in the test set) remain unclassified by ESDR for la1, rcv2, rcv4 and wap. This phenomenon of ESDR may be allowed to achieve such a good categorization accuracy that a very small amount of documents (e.g., 3 documents out of 200 for rcv4) remain unclassified when a decision is not so strong.

5.4.2 Time and Space Complexity of ESDR

The proposed algorithm takes $O(N^2)$ time to build Sim_trn and then it takes $O(N^2)$ time to create the distance matrix (dis_trn) of the set of training documents in Algorithm 5. Successively $O(N)$ time is required to create Sim_tst and then it takes $O(N)$ time to find the distances (dis_tst) of the test document with all the training documents. Let us assume g number of documents have been discarded from the training set to develop SN, which takes $O(N-g)log((N-g)$ time. The rest of the steps, i.e., the categorization using SN can be done in $O(N-g)$ time in worst case. Thus the time complexity of ESDR is $\big(O(N^2) + O(N-g)\log(N-g) + O(N-g)\big)$ i.e., $O(N^2)$.

The similarity matrix of the set of N training documents (Sim_trn) requires $N \times N$ memory locations, and similarity values of the test document with all the training documents (Sim_tst) require N memory locations. Thus, at most N space is needed to store the set of neighbors (SN) for the test document. Therefore the space complexity of ESDR is $O(N^2)$.

Table 5.3 summarizes the processing time of each method on a quad core Linux workstation. The proposed one initially develops the extensive similarity matrix by

Table 5.3: Execution Time (in seconds) of Different Classifiers

Data Sets	ESDR	TkNN	kNN	Weighted kNN	Adaptive kNN	Naive Bayes	SVM
20ns	312.17	58.67	117.02	287.71	124.62	94.96	102.35
fbis	47.16	5.55	18.52	68.50	15.61	6.90	7.58
la1	105.59	1.90	8.81	9.71	7.16	4.67	6.25
la2	107.06	0.70	1.82	1.93	1.49	4.07	6.72
oh10	45.06	0.85	8.27	12.18	4.04	2.30	3.15
oh15	46.70	0.56	7.10	10.90	3.51	2.40	2.87
rcv1	76.21	1.32	3.26	8.38	3.36	5.63	6.26
rcv2	84.28	1.38	4.36	12.62	5.90	6.22	7.76
rcv3	84.06	1.68	4.28	12.45	5.84	6.36	7.72
rcv4	83.58	1.75	4.44	12.42	5.68	6.35	7.53
tr31	56.07	0.89	3.40	4.12	2.55	3.36	3.67
tr41	53.25	0.65	6.68	10.33	3.28	3.18	3.84
tr45	54.46	0.50	5.36	8.42	2.57	3.64	4.25
wap	62.18	1.18	52.77	158.66	10.65	5.04	5.63

the documents of training set, which takes quadratic time in respect to the number of documents in the training set. Then SN is created and ESDR is performed on SN to categorize a test document. The execution time of ESDR is the total time to perform these two stages, i.e., building extensive similarity matrix (including the time of estimating θ) and performing text categorization using extensive similarity between documents. The time of ESDR shown in Table 5.3 for each data set is the sum of the times taken to categorize all the test documents. Similarly for the other methods also the time shown for each data set is the sum of the processing times to categorize all the test documents. It can be observed from Table 5.3 that in most of the cases the computational time of ESDR is more than the other competing methods, because of the development of extensive similarity matrix before categorization. The effectiveness of ESDR by extensive similarity has become clear from the experiments. Hence this much computational cost may be allowed to achieve such a good performance for text categorization. The data sets used in the experiments are of different sizes with different dimensionalities. Note that the range of dimensionality of the data sets varies from 2000 (fbis) to 35218 (20ns) and the size of the data sets ranges between 690 (tr45) to 18000 (20ns). Hence the proposed one can be applied to any high dimensional real life data set.

5.5 Conclusions and Discussion

A similarity based decision rule for text categorization is introduced in this chapter. The proposed decision rule finds the category of a new document by applying a discrimination criterion on majority voting for category selection and extensive similarity measure is used to order the documents of the training set for majority voting. The similarity measure finds the similarity between any two documents by determining their individual distances with every other document in the training set. ESDR is compared with TkNN, which uses the cosine similarity in finding the similarity between documents and the same discrimination criterion as used in ESDR. The experimental results show that the proposed one performs better than TkNN. Thus it is revealed from the results that extensive similarity measure plays significant role in the performance of ESDR for text categorization.

Initially ESDR discards those documents which are dissimilar with the test document (practically the extensive similarity values are negative for these documents) and develops an ordered Set of Neighbors (SN) with rest of the documents of the training set. The method starts with first β number of documents of SN and continues until a decision is made or it reaches the last member of SN. The test document is not (arbitrarily) assigned to a category, if the proposed decision rule fails to satisfy the said criterion after checking all the documents of SN. This is the most significant property of ESDR.

The proposed decision rule has two parameters, the first one is θ for extensive similarity between documents and the second one is β for majority voting. It has become clear from the experimental results that one of the values 2, 3, 4 of β provides better classification results. The experimental results also confirm that the technique used for estimation of θ is fruitful. Hence this estimation technique may be applied for any text data set.

Chapter 6

A Supervised Term Selection Technique for Text Categorization

6.1 Introduction

Text categorization is the process of automatic grouping of documents which helps to effectively retrieve documents from large text corpora. The performance text categorization techniques suffer from high dimensionality and sparsity of the text data. It is very important to find the discriminating terms for each category and to reduce the size of vocabulary by removing the irrelevant terms for effective text categorization. In text data each unique term is considered as a feature. Hence feature selection may be the solution for dimensionality reduction without compromising on the quality of text categorization. The feature subset selection methods for text categorization task use an evaluation function that is applied to a single term [92]. All the terms are independently evaluated and a score is assigned to each of them. Then the terms are sorted according to those weights and a predefined number of best terms form the resultant subset of terms. Various such term selection methods for text categorization are available in the literature.

Document Frequency thresholding (DF), Information Gain (IG), Mutual Information (MI), χ^2 statistic (CHI), Gain Ratio (GR), Odds Ratio (OR) etc. are commonly used term selection techniques in text categorization [38] [92] [131]. The evaluation functions for term selection of these methods stress on either common terms (e.g., CHI, GR) or the rare terms (e.g., MI) of the vocabulary. They do not provide special attention on those terms which occur in high ratio in each category.

It is observed from the literature survey that the methods which select the rare terms can not identify the important terms for categorization [92]. Thus the methods which favor the common terms of the vocabulary have become popular [131], but these methods also suffer in the presence of uneven sizes of the categories in the corpus [136]. Consider an example of a term that is occurring highly in a category, but the size of the category is small in comparison to the other categories of the corpus. Practically the term is a rare term in the global feature space and the method which gives importance to common terms give low priority to this term. Hence these terms which uniquely represent a category containing few documents may not be selected in the resultant subset of terms by the methods which favor common terms. Thus the performance of the text categorization may be degraded.

A supervised term selection technique is proposed in this article which gives high priority to the terms which occur highly in each category irrespective of their occurrence in the entire corpus. A new evaluation function named as, *term relatedness* is proposed which finds similarity between a term and a category and then every term of the vocabulary is assigned a score depending on its similarity with all the categories. Subsequently all the terms are ranked according to their individual score and a predefined number of terms having high rank (low score) are selected as discriminating terms. The proposed term relatedness assigns low score for high priority.

The proposed method has been applied on several well corpora. kNN and SVM classifiers are used to judge the effectiveness of the proposed technique. The experimental results show that the proposed method performs significantly better than the other term selection techniques for text categorization even after the removal of 90% terms from the vocabulary.

The chapter is organized as follows - section 6.2 explains the proposed term selection technique. The experimental results on several text data sets are given in section 6.3. Conclusions about the proposed method is presented in section 6.4.

6.2 Proposed Term Selection Framework

The evaluation functions for term selection that are used frequently (e.g., CHI, OR IG, MI etc.) would ultimately find either common terms or rare terms [131]. The methods which give importance to rare terms (e.g., MI) may be able to find the distinct terms of a category (rather document). Note that only those rare terms that

occur in few documents in a particular category are selected for high dimensional text data, if the threshold is not selected properly. As a result the documents belong to the same category may become dissimilar to each other and this may degrade the performance of the classifier. In practice for high dimensional corpora the methods like MI (which favor rare terms) can not be taken as a good term selection technique [131].

On the other hand the methods which give importance to common terms (e.g., CHI, OR etc.) have shown good performances in text categorization in some previous studies [92, 131]. Some of the issues regarding the existing methods are stated here. Let us consider an example where the same term is present in multiple categories, and its number of occurrences in every such category is not so high. Hence this term should get less importance than the term which occurs in high ratio (The ratio of the number of occurrences of the given term in a particular category with the number of occurrences of the same term in all the categories is said to be the ratio of the term.) in a particular category. The methods like CHI, and OR do not give special attention to this phenomenon. Consider another example of a corpus which is not evenly distributed among different categories i.e., the number of documents in one category (say C_1) is significantly more than the other category (say C_2). The terms which exist only in C_2 may not get high score by these evaluation functions which favor the common terms. Basically the terms of C_1 may dominate the terms of C_2 in the best subset of terms in most of the above term selection techniques, which is not desirable in practice. A supervised term selection method for text categorization is proposed here by considering the above issues. The aim of the proposed term selection method is to give importance to the representative terms of each category (even if the data is unevenly distributed). The proposed framework maintains the following order of preference for the terms to develop the ultimate subset of terms for text categorization.

1) The terms occurring in a particular category with high ratio and they do not exist in the other categories.

2) The terms occurring in multiple categories and they occur with a higher ratio in a particular category than the other categories.

3) The terms occur in several categories, but they do not occur with as high a ratio as in case 2 in any particular category.

4) The terms that occur in a few documents, i.e., the term is a rare term. This type of term may occur in multiple categories, but they occur in a few documents of those categories.

An evaluation function is proposed here to maintain this order of preference for term selection. Let there are m categories in the corpus. The proposed evaluation function for a term t and a category $C_i,\ i = 1, 2, ..., m$ is named as *Term ReLatedness (TRL)* and it is defined as follows:

$$TRL(t, C_i) = \begin{cases} 1, & \text{if } P(t, C_i) = 0 \\ 0, & \text{if } P(t, C_i) = P(t) = P(C_i) \\ 1 - TCR(t, C_i), & \text{if } P(t, C_i) = P(t) \neq P(C_i) \\ 1 - TRF(t, C_i), & \text{if } P(t, C_i) = P(C_i) \neq P(t) \\ 1 - TF(t, C_i), & \text{otherwise} \end{cases} \tag{6.1}$$

Here $P(t)$ is the probability that a document contains term t and $P(C_i)$ is the probability that a document belongs to category C_i. $P(t, C_i)$ denotes the probability that a document belonging to category C_i and containing the term t.

The *Term Factor (TF)* of term t and category C_i is represented as

$$TF(t, C_i) = \frac{\min(P(t), P(C_i)) - P(t, C_i)}{\max(P(t), P(C_i)) - P(t, C_i)} \times \frac{P(t) - P(t, \overline{C_i})}{P(t)} \times E(C_i)$$

$P(t, \overline{C_i})$ is the probability that a document containing the term t, but does not belong to category C_i. $E(C_i)$ is the entropy of category C_i and $E(C_i) = -P(C_i) \log P(C_i)$. TF determines the existence of a term among all the categories. TF becomes high when $P(t, C_i)$ is close to $P(t)$ and $P(C_i)$ and $P(t, \overline{C_i})$ is very low, i.e., when a term occur highly in a category and does not exist (or exists with a low ratio) in other categories. The estimation of $P(t), P(C_i), P(t, C_i)$ and $P(t, \overline{C_i})$ are described in section 1.3.1.2.

The *Term Category Ratio (TCR)* of term t and category C_i is represented as

$$TCR(t, C_i) = \frac{1 + P(t, C_i)}{1 + P(C_i)} \times E(C_i)$$

TCR determines how frequently a term occurs in a category, but it does not check whether the term exists only in that category. TCR becomes high when $P(t, C_i)$ is close to $P(C_i)$.

The *Term Relative Frequency (TRF)* of term t in category C_i is represented as

$$TRF(t, C_i) = \frac{1 + P(t, C_i)}{1 + P(t)} \times E(C_i)$$

TRF determines how frequently a term occurs in a category compared to the other categories. TRF becomes high when $P(t, C_i)$ is close to $P(t)$. The values of *TF*, *TCR* and *TRF* belong to $(0, 1)$ whenever they are applied to find TRL between a term and a category. The objective is to maximize the *TCR*, *TRF* and *TF* scores, i.e., to minimize TRL.

The TRL values of a term for all the categories are calculated and then the minimum value is taken as the ultimate TRL value of the term. Thus the over all *TRL* score of a term, $TRL_{all}(t)$ can be obtained in the following way.

$$TRL_{all}(t) = \min\{TRL(t, C_i) : i = 1, 2, ..., m\} \tag{6.2}$$

Eventually all the terms are ranked in increasing order according to their $TRL_{all}(t)$ values and a predefined number of top terms are selected for categorization. Algorithm 6 describes the proposed term selection method for text categorization.

6.2.1 Properties of Term Relatedness

- Note that a term t and a category $C_i,\ i = 1, 2, ..., m$ under consideration should exist, which necessarily implies that $P(t) > 0$ and $P(C_i) > 0$.

- The value of TRL ranges between 0 and 1. The value is 0 when a term occurs in all the documents of a category and it does not occur in any other category. The value of TRL is 1 when a term does not occur in any document of that category.

- $TRL(t, C_i) \geq 0,\ \forall t$ and $\forall C_i,\ i = 1, 2, ..., m$.

- $TRL(t, C_i) = 0$ for a term t and a category $C_i,\ i = 1, 2, ..., m$, if $P(t) = P(C_i) = P(t, C_i)$, i.e., t occurs only in C_i and it gets the lowest score among all the terms.

Algorithm 6 Term Selection by Term Relatedness

Input: a) $D = \{d_1, d_2, ..., d_N\}$ be a set of N training documents.
b) A set of m categories, $C = \{C_1, C_2, ..., C_m\}$.
c) $T = \{t_1, t_2, ..., t_n\}$ is the set of n terms in the vocabulary.
d) γ is the number of terms to be selected.
Output: A reduced set of terms RST.

Steps:

1: **for** each $t_i \in T$ **do**
2: **for** each $C_j \in C$ **do**
3: Calculate $TRL(t_i, C_j)$ following equation 6.1
4: **end for**
5: Calculate $TRL_{all}(t_i)$ by equation 6.2
6: **end for**
7: $ST \leftarrow$ Sort $TRL_{all}(t_i),\ \forall t_i \in T$ in increasing order
8: $RST \leftarrow \{ST(i) : 1 \leq i \leq \gamma\}$
9: return RST

6.2.2 Discussion

It is to be noted that a term gets highest preference, when it occurs with high ratio in one category and it does not occur in the other categories. If a term occurs with high ratio in most of the categories then the term gets very low preference. The value of TF is low when both the fractions $\frac{\min(P(t), P(C_i)) - P(t, C_i)}{\max(P(t), P(C_i)) - P(t, C_i)}$ and $(P(t) - P(t, \overline{C_i})/P(t)$ of TF are low (say 0.2). As a result TRL becomes high. Thus the proposed method restricts the stopwords by giving low preference to them. The entropy is used to maintain the homogeneity between categories. Otherwise the terms of a category with few documents may be suppressed by the terms of a category with large number of documents. Let us discuss about how the proposed TRL maintains the order of preference between a term and a category that have been assumed initially to design it. All possible TRL values between a term and a category are described in figure 6.1.

Case 1: If $P(t, C_i) = P(C_i) = P(t)$ for any C_i and t then the value of TRL is 0, which is the minimum value of TRL. As TRL values are ranked in increasing order, so this term gets the highest priority in the ultimate subset of terms. If t occurs in all the documents of C_i, and it does not occur in any other category, then $P(t, C_i) = P(C_i) = P(t)$, and hence it gets the highest priority. Although this type

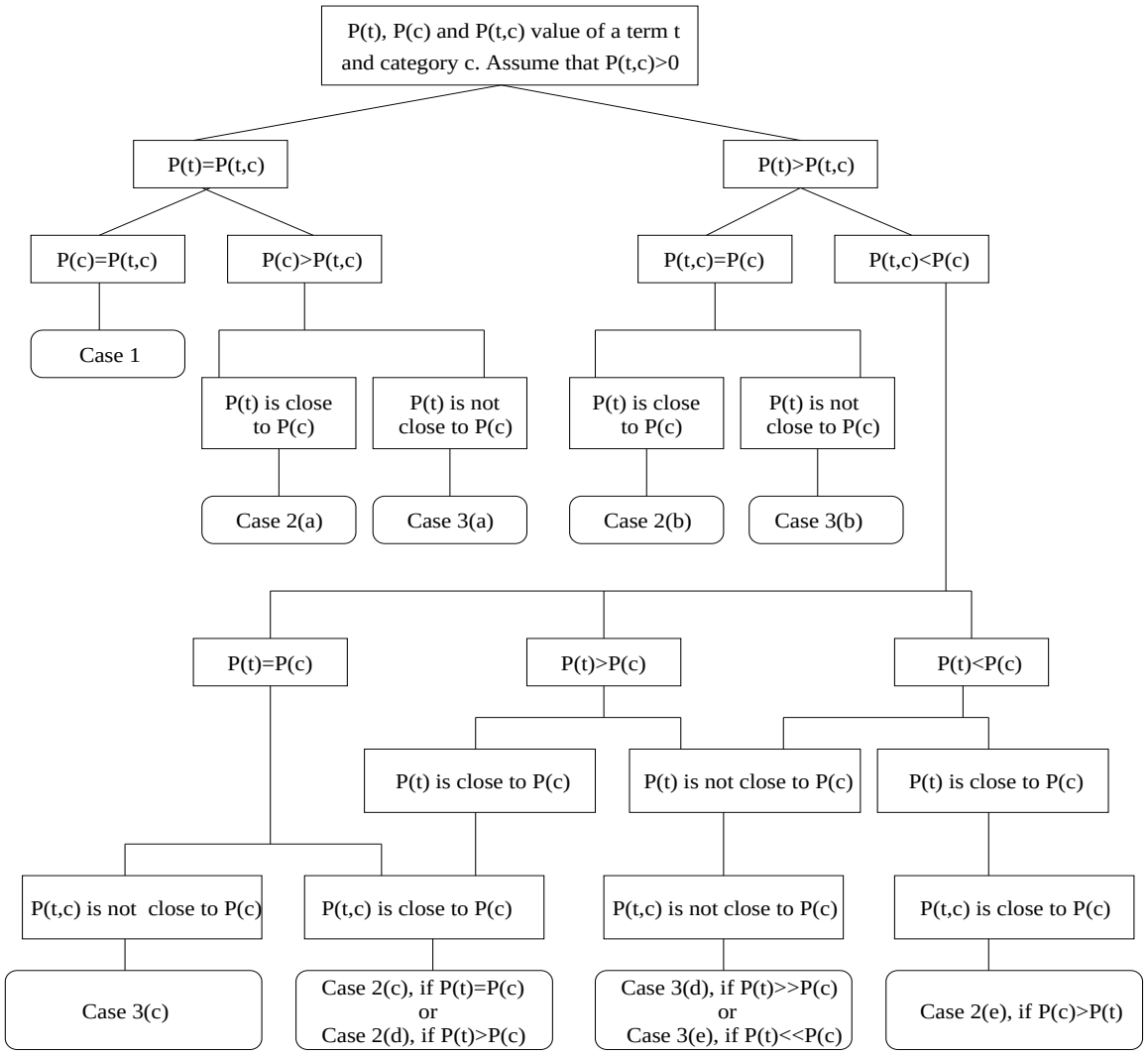

Figure 6.1: All Possible TRL Values of a Term t and a Category c

---

Rectangular box with rounded corner indicates the final state.

of terms are very important for a category, this sort of terms hardly occur in a corpus. The following claim has been made regarding case 1 of TRL. The justification of the claims is given in appendix C.

Claim 1): The terms of case 1 get highest preference in a particular category.

Case 2: The value of $TRL(t, C_i)$ is close to 0 for any category C_i and term t in the following situations:

(a) $P(C_i) > P(t), P(t) = P(t, C_i)$ and $P(t)$ is close to $P(C_i)$.

(b) $P(t) > P(C_i), P(C_i) = P(t, C_i)$ and $P(t)$ is close to $P(C_i)$.

(c) $P(t) = P(C_i), P(t) \neq P(t, C_i)$, but $P(t)$ is close to $P(t, C_i)$.

(d) $P(t) > P(C_i), P(t) \neq P(t, C_i)$, and both $P(t), P(t, C_i)$ are close to $P(C_i)$.

(e) $P(C_i) > P(t), P(t) \neq P(t, C_i)$ and both $P(t), P(t, C_i)$ are close to $P(C_i)$.

The terms of all the last five cases generally represent a particular category, since they exist in most of the documents of that category. Hence these terms are very important for categorization. The terms of case 2(b) get higher preference than the terms of case 2(c) and case 2(d) for the same $P(t)$ and $P(C_i)$ values. The terms of case 2(c) get higher preference than the terms of case 2(d) for the same $P(t, C_i)$ and $P(C_i)$ values. For the same $P(t)$ and $P(t, C_i)$ values, the terms of cases 2(d) and 2(e), the terms of cases 2(b) and 2(e) get comparable priority depending on their individual TRL value. All the terms of cases 2(b), 2(c), 2(d) and 2(e) appear after the terms of case 2(a) for the same $P(t)$ and $P(t, C_i)$ values in the best subset of terms. The following claims have been made whose justification is given in appendix C regarding all the sub-cases of case 2.

Claim 2.a): The terms of case 2(a) get higher preference than the terms of case 2(b) for the same $P(C_i)$ and $P(t, C_i)$ values.

Claim 2.b): For the same $P(C_i)$ and $P(t)$ values the terms of case 2(b) get higher preference than the terms of case 2(c).

Claim 2.c): For the same $P(C_i)$ and $P(t)$ values the terms of case 2(b) get higher preference than the terms of case 2(d).

Claim 2.d): The terms of case 2(a) get higher preference than the terms of case 2(e) for the same $P(C_i)$ and $P(t)$ values.

Claim 2.e): For the same $P(C_i)$ and $P(t, C_i)$ values the terms of case 2(c) get higher preference than the terms of case 2(d).

Case 3: The value of $TRL(t, C_i)$ is close to 1 for any category C_i and term t in the following situations:

(a) $P(C_i) >> P(t)$, i.e., $P(t)$ is not close to $P(C_i)$ and $P(t) = P(t, C_i)$.

(b) $P(t) >> P(C_i)$, i.e., $P(t)$ is not close to $P(C_i)$ and $P(C_i) = P(t, C_i)$.

(c) $P(t) = P(C_i)$ and $P(C_i) >> P(t, C_i)$.

(d) $P(t) >> P(C_i)$ and $P(t) \neq P(t, C_i)$.

(e) $P(C_i) >> P(t)$ and $P(t) \neq P(t, C_i)$.

The terms of case 3(a) occur poorly in C_i and they do not occur in any other category, i.e., these terms are rare terms occurring in a corpus. The terms of cases 3(b), 3(c) and 3(d) also occur poorly in C_i, but they may occur highly in any other category and may be important for that category. Note that the terms belong to all the sub-cases of case 3 are not so important for C_i. Although the terms of case 3(a) get higher preference than the terms of other sub-cases of case 3 for the same $P(t)$ and $P(t, C_i)$ values, since they occur only in C_i. Let us discuss how far the proposed TRL maintains all the sub-cases of case 3 by the following claims. The justification of these claims are given in appendix C.

Claim 3.a): The terms of all the sub-cases of case 3 get lower preference than the terms of all the sub-cases of case 2 in C_i.

Claim 3.b): The terms of case 3(a) get higher preference than the terms of case 3(b) for the same $P(C_i)$ and $P(t, C_i)$ values.

Claim 3.c): For the same $P(C_i)$ and $P(t)$ values the terms of case 3(b) get higher preference than the terms of case 3(c).

Claim 3.d): For the same $P(C_i)$ and $P(t)$ values the terms of case 3(b) get higher

preference than the terms of case 3(d).

Claim 3.e): The terms of case 3(a) get higher preference than the terms of case 3(e) for the same $P(C_i)$ and $P(t)$ values.

Claim 3.f): For the same $P(C_i)$ and $P(t, C_i)$ values the terms of case 3(c) get higher preference than the terms of case 3(d).

It is to be noted here that the above cases state the characteristics of TRL that we want it to possess. It is also to be stated that the characteristics are stated intuitively, but not mathematically. The meaning of *two quantities being close* is not mathematically clear. Proving mathematically the characteristics is an impossible task since all the properties are not stated mathematically. Additionally, there are many gray areas in the last few cases (for example, there is no clear distinction between the statements $P(t) >> P(C_i)$ and $P(t) > P(C_i)$). The proposed measure is an attempt to find significant terms of each category of any corpus.

6.3 Experimental Evaluation

This section describes the performance of various term selection techniques in text categorization using fbis, la1, la2, rcv1, rcv2, rcv3, rcv4, tr41, tr45 and wap data sets. The overview of the data sets is given in Table 1.5. The effectiveness of different term selection algorithms is evaluated using the performances of kNN and SVM classifiers. The kNN classifier is chosen since it showed very good performance for text categorization in some previous studies by Yang et al. [130, 131]. The value of neighborhood parameter k has been fixed by applying 10-fold cross validation method on the training set. The range of k has been set from 1 to 15. The best value among those 15 performances for each data set is reported in Table 6.1 and Table 6.2. Similarly SVM has been chosen for performance evaluation, since it has been shown as one of the most powerful learning algorithms for text categorization [71, 103].

The data sets used in the experiments have no separate test and training sets. Hence 10 fold cross validation is performed on the entire data set. kNN and SVM have been executed for 10 times to reduce the effect of random selection of the documents by cross validation. The average results of this 10 executions are reported in Table 6.1, Table 6.2, Table 6.3, Table 6.3 and Table 6.4. The codes to implement

BNS, CE, CHI, DF, GI, GR, IG, MI and ODR for term selection have been developed by the author as no such codes are publicly available on any reliable source. The codes to implement the proposed TRL to create reduced set of terms has been developed by the author in Matlab. The kNN classifier is implemented using the default package available in Matlab and the SVM classifier is implemented using the libsvm tool [26]. In the experiments the linear kernel has been used [135] and the other parameters of SVM remain the same as provided by the default option of libsvm tool.

The proposed term selection technique has been compared with nine other term selection methods for text categorization namely, BNS [49], CE [92], CHI [55], DF [131], GI [114], GR [38], IG [131], MI and ODR [92] (all the methods are described in section 1.3.1.2) using accuracy and f-measure. The maximum score among all the categories has been selected for performance evaluation for BNS, CE, CHI, GR, MI and ODR in the experimental analysis. Table 6.1 shows the f-measure values of kNN and Table 6.3 shows the f-measure values of SVM of all the term selection methods for text categorization. Similarly Table 6.2 shows the accuracy values of kNN and Table 6.4 shows the accuracy values of SVM respectively of all the term selection methods for text categorization. The performances of the kNN and SVM classifiers on all data sets are reported after removing 50%, 70%, 80%, and 90% least important terms according to the measure under consideration. The vocabulary size (VS) in each of these Tables indicates that the experiments are performed when there are 10%, 20%, 30%, 50% and 100% most important terms in the vocabulary. The best f-measure and accuracy values among all the term selection techniques are highlighted for each data set in all these tables of experimental results.

6.3.1 Analysis of Results using kNN classifier

It can be seen from Table 6.1 that the performances of kNN are better on the reduced set of terms than the complete set of terms for all the data sets (except fbis) using the proposed term selection framework (even after removal of 90% terms). The performance of kNN would not be so decent in the rest of the cases of Table 6.1, if some significant terms are removed from the complete set of terms. Thus it has become clear from the experiments that the proposed method is able to retrieve significant information from the data. The performances of kNN classifier on the reduced set of terms created by BNS, CE, DF, MI, ODR are not better than the

Table 6.1: Performance of Various Term Selection Methods using F-measure of kNN Classifier

Data Set	VS in (%)	F-measure									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (proposed)
fbis	10%	0.351	0.352	0.761	0.737	**0.759**	0.751	0.753	0.415	0.613	0.744
	20%	0.430	0.434	**0.766**	0.754	0.761	0.766	0.766	0.594	0.664	0.763
	30%	0.471	0.469	0.768	0.756	**0.770**	0.768	**0.770**	0.745	0.672	0.766
	50%	0.552	0.553	0.768	0.759	0.768	**0.773**	0.772	0.768	0.722	0.770
	100%	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773	0.773
la1	10%	0.490	0.539	**0.818**	0.649	0.811	0.812	0.814	0.297	0.659	0.796
	20%	0.561	0.625	0.812	0.686	0.819	0.817	0.821	0.582	0.746	**0.831**
	30%	0.587	0.670	0.816	0.740	0.817	0.820	0.822	0.682	0.721	**0.834**
	50%	0.638	0.726	0.821	0.778	0.817	0.821	0.820	0.778	0.632	**0.826**
	100%	0.794	0.794	0.794	0.794	0.794	0.794	0.794	0.794	0.794	0.794
la2	10%	0.522	0.537	**0.846**	0.659	0.834	0.835	0.836	0.487	0.775	0.823
	20%	0.591	0.625	0.835	0.713	0.835	0.834	0.836	0.571	0.781	**0.854**
	30%	0.653	0.690	0.830	0.750	0.839	0.837	0.839	0.756	0.758	**0.860**
	50%	0.686	0.735	0.833	0.825	0.831	0.835	0.833	0.810	0.687	**0.842**
	100%	0.814	0.814	0.814	0.814	0.814	0.814	0.814	0.814	0.814	0.814
rcv1	10%	0.500	0.500	0.868	0.841	**0.873**	0.865	0.866	0.501	0.546	0.858
	20%	0.565	0.564	0.869	0.853	0.866	**0.870**	0.869	0.582	0.622	0.862
	30%	0.580	0.578	0.873	0.852	0.867	0.871	0.870	0.650	0.658	**0.874**
	50%	0.651	0.653	0.860	0.852	0.870	0.865	0.866	0.695	0.711	**0.876**
	100%	0.858	0.858	0.858	0.858	0.858	0.858	0.858	0.858	0.858	0.858
rcv2	10%	0.488	0.487	0.863	0.846	**0.876**	0.865	0.867	0.491	0.525	0.856
	20%	0.549	0.547	0.868	0.858	**0.873**	0.866	0.869	0.576	0.580	0.858
	30%	0.584	0.585	0.871	0.858	0.873	0.869	0.870	0.644	0.650	**0.876**
	50%	0.638	0.637	0.865	0.855	0.870	0.867	0.865	0.728	0.705	**0.875**
	100%	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860
rcv3	10%	0.490	0.490	0.872	0.836	**0.874**	0.868	0.866	0.486	0.536	0.862
	20%	0.562	0.562	**0.874**	0.865	0.874	0.867	0.869	0.592	0.615	0.866
	30%	0.592	0.591	0.869	0.850	0.865	0.866	0.865	0.643	0.659	**0.873**
	50%	0.654	0.656	0.861	0.856	0.866	0.863	0.864	0.713	0.715	**0.870**
	100%	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860	0.860
rcv4	10%	0.467	0.469	0.868	0.854	0.871	0.872	**0.873**	0.498	0.531	0.865
	20%	0.533	0.533	0.871	0.874	0.876	**0.880**	0.878	0.565	0.575	0.868
	30%	0.566	0.564	0.869	0.867	0.874	0.872	0.869	0.620	0.634	**0.878**
	50%	0.652	0.650	0.871	0.861	0.874	0.871	0.872	0.694	0.711	**0.876**
	100%	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864	0.864
tr41	10%	0.672	0.688	0.918	0.827	0.918	0.920	0.921	0.650	0.798	**0.925**
	20%	0.772	0.768	0.925	0.873	0.919	0.921	0.920	0.720	0.811	**0.928**
	30%	0.844	0.855	0.920	0.922	0.920	0.924	0.920	0.788	0.835	**0.930**
	50%	0.889	0.894	0.921	0.905	0.920	0.924	0.922	0.921	0.875	**0.929**
	100%	0.918	0.918	0.918	0.918	0.918	0.918	0.918	0.918	0.918	0.918

cont.

Data Set	VS in (%)	F-measure									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (proposed)
tr45	10%	0.728	0.740	**0.899**	0.836	0.897	0.890	0.892	0.705	0.807	0.881
	20%	0.768	0.767	**0.900**	0.856	0.897	0.896	0.898	0.720	0.810	0.884
	30%	0.840	0.841	0.896	0.831	0.890	0.890	0.889	0.823	0.887	**0.910**
	50%	0.895	0.886	0.896	0.850	0.888	0.888	0.885	**0.929**	0.832	0.918
	100%	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884
wap	10%	0.513	0.527	0.764	0.559	0.751	**0.776**	0.768	0.326	0.428	0.758
	20%	0.532	0.608	0.771	0.610	0.759	**0.781**	0.777	0.452	0.475	0.765
	30%	0.592	0.649	0.778	0.642	0.763	**0.784**	0.781	0.558	0.518	0.770
	50%	0.644	0.690	**0.783**	0.732	0.722	0.781	0.780	0.640	0.587	0.774
	100%	0.742	0.742	0.742	0.742	0.742	0.742	0.742	0.742	0.742	0.742

case of using complete set of terms in most of the cases of Table 6.1.

The performances of kNN classifier using the reduced sets of terms developed by CHI, GI, IG and GR are better than the case of using the complete set of terms for all the data sets except fbis. For fbis CHI, GI, IG and GR have failed to retrieve significant information. It may be noted that the proposed method performs better than the other competing methods in most of the cases in Table 6.1.

It can be seen that the proposed method has 360 comparisons with the other methods in Table 6.1 for 50% to 90% removal of terms from the actual vocabulary. TRL performed better than the other methods in 285 cases and in the remaining 75 cases other methods (e.g., CHI, GR etc.) beat the proposed method. An example is noted here where the other methods have an edge over TRL, for the data set la1, when there are 10% terms in vocabulary. CHI, GI, GR and IG (the f-measure values are 0.818, 0.811, 0.812 and 0.814 respectively) are seen to be better than TRL (f-measure is 0.796) in Table 6.1 for kNN classifier.

The same statistical significance test discussed in chapter 2 and used in the other chapters is performed here to check whether the significance of the differences of two competing methods, e.g., whether 0.818, 0.811 and 0.814 are significantly different from 0.810. The t-test has been done for all the comparisons between the proposed method and other term selection methods in this chapter. It has been found using t-test that out of the 285 cases where TRL performed better than the other methods in Table 6.1, the differences are statistically significant in 250 cases for the level of significance 0.05. Out of the rest 75 cases where other methods have an edge over TRL, 60 differences are found to be statistically significant for the same level of significance. Thus in 80.64% cases TRL performs significantly better than the other

Table 6.2: Performance of Various Term Selection Methods using Accuracy (in %) of kNN Classifier

Data Set	VS in (%)	Accuracy									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (Proposed)
fbis	10%	39.84	39.41	79.34	76.29	**79.41**	79.23	79.37	44.74	64.87	77.38
	20%	48.02	48.24	**80.44**	77.58	80.10	80.25	80.36	61.46	68.30	79.53
	30%	51.51	51.55	80.27	78.15	80.33	**80.93**	80.76	77.50	69.80	79.67
	50%	60.30	60.18	80.66	78.63	80.77	**81.11**	81.09	80.44	75.77	79.61
	100%	79.04	79.04	79.04	79.04	79.04	79.04	79.04	79.04	79.04	79.04
la1	10%	50.64	54.98	**81.56**	66.50	81.16	81.35	81.47	34.23	65.79	80.75
	20%	56.13	63.91	81.25	69.49	82.00	82.17	82.33	58.29	74.29	**83.43**
	30%	59.15	68.26	81.64	74.05	81.82	82.21	82.35	68.72	72.35	**84.01**
	50%	63.53	73.90	82.22	79.01	81.76	81.48	81.32	78.06	61.85	**82.42**
	100%	79.75	79.75	79.75	79.75	79.75	79.75	79.75	79.75	79.75	79.75
la2	10%	51.43	55.23	**84.73**	66.75	83.56	83.68	83.74	52.65	78.44	83.90
	20%	57.34	63.16	83.75	73.30	83.52	83.32	83.41	65.27	73.67	**85.59**
	30%	64.72	69.57	83.10	74.05	83.94	84.03	84.15	78.80	76.67	**86.17**
	50%	68.01	74.40	83.57	79.63	83.11	83.44	83.32	82.80	68.82	**84.43**
	100%	81.98	81.98	81.98	81.98	81.98	81.98	81.98	81.98	81.98	81.98
rcv1	10%	56.21	56.19	**87.61**	83.29	87.35	87.45	87.49	57.66	60.96	87.26
	20%	60.44	60.63	87.96	84.35	**87.98**	87.88	87.83	63.02	65.69	87.74
	30%	62.46	62.70	88.09	83.98	88.03	87.94	87.87	67.73	68.55	**88.15**
	50%	69.14	69.39	88.02	84.08	88.13	88.09	88.15	71.47	72.95	**88.29**
	100%	86.84	86.84	86.84	86.84	86.84	86.84	86.84	86.84	86.84	86.84
rcv2	10%	55.97	56.03	87.58	83.89	**88.62**	88.01	88.08	55.94	59.16	87.46
	20%	59.68	59.76	88.04	84.97	**88.24**	87.96	88.04	61.42	62.54	87.79
	30%	63.00	63.16	88.23	84.79	88.15	87.82	87.89	66.97	67.86	**88.34**
	50%	67.80	67.66	88.12	84.49	88.12	87.72	87.64	74.64	72.43	**88.22**
	100%	86.96	86.96	86.96	86.96	86.96	86.96	86.96	86.96	86.96	86.96
rcv3	10%	55.94	56.00	88.04	82.92	**88.09**	88.02	87.97	56.38	60.11	87.69
	20%	59.93	59.91	**88.20**	84.19	88.08	88.07	88.12	63.57	65.35	87.81
	30%	62.85	62.83	87.92	83.86	87.92	87.89	87.85	67.13	67.89	**88.18**
	50%	67.93	67.84	87.63	84.37	87.67	87.36	87.44	73.06	73.52	**88.20**
	100%	87.10	87.10	87.10	87.10	87.10	87.10	87.10	87.10	87.10	87.10
rcv4	10%	53.56	53.73	88.05	84.64	**88.41**	88.33	88.36	57.01	59.76	87.75
	20%	58.31	58.34	88.21	85.40	88.33	**88.45**	88.40	60.36	62.42	87.91
	30%	60.89	60.87	87.82	85.78	88.08	88.08	88.01	65.11	66.64	**88.16**
	50%	69.01	69.02	87.76	85.24	87.27	87.86	87.92	71.79	72.85	**88.22**
	100%	87.28	87.28	87.28	87.28	87.28	87.28	87.28	87.28	87.28	87.28
tr41	10%	66.59	69.93	92.02	76.46	92.28	92.66	92.74	42.63	68.96	**92.83**
	20%	73.63	77.52	92.78	86.35	92.26	92.63	92.57	66.91	74.66	**93.00**
	30%	84.92	86.26	92.46	88.05	92.24	92.85	92.74	78.98	83.58	**93.46**
	50%	89.52	90.00	92.08	90.36	92.33	92.65	92.58	92.47	87.20	**93.04**
	100%	92.32	92.32	92.32	92.32	92.32	92.32	92.32	92.32	92.32	92.32

cont.

Data Set	VS in (%)	Accuracy									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (Proposed)
tr45	10%	72.27	75.47	**90.55**	83.56	90.13	89.80	89.85	66.91	79.73	89.45
	20%	78.76	78.65	**90.17**	86.24	90.12	90.02	90.12	73.14	81.47	89.86
	30%	85.14	85.20	90.28	83.69	89.60	89.50	89.42	83.26	89.23	**91.55**
	50%	90.10	89.28	91.23	86.07	89.28	89.63	89.50	91.11	83.27	**91.88**
	100%	88.98	88.98	88.98	88.98	88.98	88.98	88.98	88.98	88.98	88.98
wap	10%	49.92	54.44	78.06	57.48	77.01	**79.21**	79.00	36.98	41.43	77.88
	20%	52.52	63.47	78.39	63.05	76.96	**79.49**	79.37	47.41	48.28	78.19
	30%	60.18	67.86	78.71	65.32	76.80	79.23	**79.25**	56.23	54.26	78.23
	50%	65.80	71.28	**79.02**	72.92	76.33	78.87	78.91	64.67	61.69	78.66
	100%	74.37	74.37	74.37	74.37	74.37	74.37	74.37	74.37	74.37	74.37

methods.

The performances of TRL and the other term selection techniques using the accuracy of kNN classifier will be discussed now. It can be seen from Table 6.2 that the performances of kNN are better on the reduced set of terms than the complete set of terms for all the data sets (except fbis) using the proposed term selection framework TRL (even after removal of 90% terms). Note that for fbis data set the performance of TRL is degraded only when 90% terms have been removed. The performances of kNN classifier on the reduced set of terms created by BNS, CE, DF, MI, ODR are not better than the complete set of terms in most of the cases of Table 6.2. The performances of kNN classifier using the reduced sets of terms developed by CHI and GI are better than the complete set of terms for all the data sets except tr41. For tr41 CHI and GI are failed to retrieve significant information. The accuracy values of kNN classifier are better on the reduced set of terms produced by GR and IG than the complete set of terms for all the data sets. It may be noted that the proposed method performs better than the other competing methods in most of the cases in Table 6.2.

It can be seen that the proposed method has 360 comparisons with the other methods in Table 6.2 for 50% to 90% removal of terms from the actual vocabulary. TRL performed better than the other methods in 286 cases and in the remaining 74 cases other methods (e.g., CHI, GR etc.) beat the proposed method. It has been found using t-test that out of the 286 cases where TRL performed better than the other methods in Table 6.2, the differences are statistically significant in 252 cases for the level of significance 0.05. Out of the rest 74 cases where other methods have an edge over TRL, 60 differences are found to be statistically significant for the same

Table 6.3: Performance of Different Term Selection Techniques using F-measure of SVM Classifier

Data Set	VS in (%)	F-measure									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (proposed)
fbis	10%	0.388	0.388	**0.813**	0.778	0.808	0.802	0.804	0.520	0.661	0.798
	20%	0.465	0.465	0.814	0.786	0.810	0.810	0.812	0.686	0.719	**0.814**
	30%	0.523	0.523	0.813	0.785	0.820	**0.823**	0.821	0.809	0.745	0.815
	50%	0.609	0.609	0.827	0.761	0.827	0.828	0.826	**0.829**	0.800	0.810
	100%	0.776	0.776	0.776	0.776	0.776	0.776	0.776	0.776	0.776	0.776
la1	10%	0.574	0.577	0.887	0.716	0.885	0.887	**0.888**	0.686	0.703	0.881
	20%	0.63	0.631	0.896	0.785	0.895	0.896	0.897	0.716	0.718	**0.899**
	30%	0.66	0.662	0.899	0.829	0.891	0.894	0.895	0.776	0.706	**0.906**
	50%	0.739	0.74	0.905	0.868	0.905	0.904	0.904	0.802	0.696	**0.918**
	100%	0.892	0.892	0.892	0.892	0.892	0.892	0.892	0.892	0.892	0.892
la2	10%	0.622	0.617	0.886	0.710	**0.888**	0.887	0.889	0.520	0.717	0.882
	20%	0.641	0.636	0.890	0.745	0.886	0.890	0.892	0.668	0.745	**0.898**
	30%	0.715	0.713	0.903	0.863	0.899	**0.905**	0.904	0.781	0.726	0.902
	50%	0.762	0.751	0.914	0.889	0.902	0.906	0.904	0.825	0.756	**0.908**
	100%	0.887	0.887	0.887	0.887	0.887	0.887	0.887	0.887	0.887	0.887
rcv1	10%	0.534	0.549	0.891	0.864	0.887	**0.892**	**0.892**	0.503	0.556	0.881
	20%	0.573	0.581	0.894	0.869	0.893	0.895	**0.896**	0.628	0.641	0.890
	30%	0.612	0.614	0.895	0.868	0.892	0.893	0.891	0.666	0.649	**0.898**
	50%	0.639	0.641	0.894	0.874	0.893	0.895	0.893	0.721	0.687	**0.901**
	100%	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872	0.872
rcv2	10%	0.542	0.540	**0.891**	0.872	0.889	0.888	0.888	0.561	0.593	0.880
	20%	0.565	0.563	**0.894**	0.873	0.891	0.890	0.893	0.628	0.611	0.884
	30%	0.589	0.584	0.894	0.875	0.891	0.890	0.887	0.661	0.648	**0.894**
	50%	0.640	0.639	0.898	0.878	0.891	0.888	0.886	0.751	0.684	**0.895**
	100%	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884	0.884
rcv3	10%	0.542	0.543	0.883	0.867	**0.886**	0.882	0.885	0.570	0.602	0.874
	20%	0.579	0.579	0.890	0.873	**0.891**	0.888	0.890	0.631	0.638	0.880
	30%	0.593	0.592	0.894	0.869	0.890	0.891	0.890	0.647	0.654	**0.896**
	50%	0.651	0.651	0.891	0.870	0.893	0.891	0.887	0.728	0.709	**0.898**
	100%	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881	0.881
rcv4	10%	0.538	0.538	0.890	0.870	**0.891**	0.888	0.890	0.572	0.589	0.884
	20%	0.565	0.565	0.889	0.871	**0.892**	0.890	0.891	0.568	0.584	0.887
	30%	0.577	0.577	0.890	0.873	0.892	0.892	0.892	0.654	0.647	**0.900**
	50%	0.639	0.639	0.891	0.873	0.893	0.894	0.893	0.749	0.682	**0.909**
	100%	0.886	0.886	0.886	0.886	0.886	0.886	0.886	0.886	0.886	0.886
tr41	10%	0.707	0.718	0.929	0.842	0.898	0.909	0.909	0.758	0.812	**0.934**
	20%	0.770	0.782	0.925	0.876	0.902	0.914	0.912	0.788	0.801	**0.939**
	30%	0.881	0.869	0.949	0.931	0.950	0.948	0.945	0.819	0.846	**0.954**
	50%	0.924	0.922	0.953	0.928	0.954	0.953	0.951	0.934	0.879	**0.960**
	100%	0.932	0.932	0.932	0.932	0.932	0.932	0.932	0.932	0.932	0.932

cont.

Data Set	VS in (%)	F-measure									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (proposed)
tr45	10%	0.674	0.674	**0.916**	0.891	0.910	0.912	0.910	0.765	0.796	0.898
	20%	0.771	0.771	**0.914**	0.898	**0.914**	0.912	0.911	0.782	0.792	0.902
	30%	0.831	0.831	0.915	0.874	0.915	**0.918**	0.917	0.875	0.838	0.915
	50%	0.901	0.905	0.917	0.893	0.919	0.915	0.915	0.909	0.868	**0.920**
	100%	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882	0.882
wap	10%	0.643	0.597	0.843	0.640	0.842	0.843	**0.845**	0.408	0.449	0.831
	20%	0.735	0.704	0.849	0.735	**0.853**	0.848	0.850	0.551	0.489	0.839
	30%	0.768	0.757	**0.854**	0.770	0.850	0.851	0.849	0.659	0.538	0.848
	50%	0.807	0.799	**0.861**	0.847	0.853	0.850	0.849	0.730	0.609	0.847
	100%	0.784	0.784	0.784	0.784	0.784	0.784	0.784	0.784	0.784	0.784

level of significance. Thus in 80.76% cases TRL performs significantly better than the other methods.

6.3.2 Analysis of Results using SVM classifier

It can be seen from Table 6.3 that the performances of SVM are better on the reduced set of terms than the original one for all the data sets using the proposed TRL except for la1, la2, rcv3 and rcv4 data sets when 90% terms have been removed from the vocabulary. Hence it may be claimed from the experiments that the proposed method is able to retrieve significant information from the data. The performances of SVM are not better on the reduced set of terms than the complete set of terms for BNS, CE, DF, MI, ODR in most of the cases of Table 6.3. SVM shows better results on the reduced sets of terms produced by CHI, GI, GR and IG than the complete set of terms for all the data sets except la1, la2 and tr41. It may be noted that TRL performs better than the other competing methods in most of the cases in Table 6.3.

It can be seen that TRL has 360 comparisons with the other methods in Table 6.3 for 50% to 90% removal of terms from the actual vocabulary. The performance of TRL is better than the other methods in 279 cases and in the remaining 81 cases other methods (e.g., CHI, GR etc.) performed better than TRL. The differences are statistically significant by t-test in 249 out of 279 cases when TRL performed better than the other methods for the level of significance 0.05. The differences are statistically significant in 62 out of 81 cases when other methods performed better than TRL for the same level of significance. Thus TRL performs significantly better

Table 6.4: Performance of Different Term Selection Techniques using Accuracy (in %) of SVM Classifier

Data Set	VS in (%)	Accuracy									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (Proposed)
fbis	10%	41.29	41.29	**81.44**	77.84	80.79	80.31	80.38	52.90	68.20	79.74
	20%	47.25	47.25	81.28	76.58	80.87	81.31	81.36	68.85	73.00	**81.48**
	30%	52.61	52.61	81.40	76.50	81.97	**82.33**	82.21	80.99	74.86	81.72
	50%	60.82	60.82	82.74	76.14	82.81	**82.94**	82.78	82.92	80.10	81.48
	100%	77.81	77.81	77.81	77.81	77.81	77.81	77.81	77.81	77.81	77.81
la1	10%	55.71	55.93	**89.69**	72.69	89.48	89.60	89.64	69.69	68.66	89.46
	20%	63.82	63.85	90.51	78.46	90.45	90.72	90.79	73.51	73.03	**91.85**
	30%	65.73	65.94	90.79	83.86	90.01	90.58	90.64	78.59	71.41	**91.98**
	50%	73.62	73.68	91.38	87.80	91.48	91.51	91.51	82.18	71.38	**92.21**
	100%	90.84	90.84	90.84	90.84	90.84	90.84	90.84	90.84	90.84	90.84
la2	10%	62.72	62.06	**88.93**	76.20	88.80	88.77	88.84	54.26	72.13	88.73
	20%	64.67	64.18	89.53	79.50	89.87	89.66	89.74	65.81	73.61	**90.15**
	30%	71.97	71.86	90.34	86.25	90.31	**90.49**	90.36	78.23	72.77	90.20
	50%	76.80	75.73	90.43	88.86	90.45	90.55	90.44	82.46	75.45	**90.79**
	100%	89.27	89.27	89.27	89.27	89.27	89.27	89.27	89.27	89.27	89.27
rcv1	10%	54.88	55.88	89.64	86.43	88.99	89.78	**89.83**	57.90	61.27	89.40
	20%	57.79	58.40	89.78	86.87	89.88	89.91	**89.98**	63.41	64.89	89.63
	30%	61.29	61.37	89.88	86.77	89.53	89.73	89.60	67.82	65.98	**90.29**
	50%	64.52	64.70	89.78	87.37	89.83	89.98	89.86	73.37	69.45	**90.58**
	100%	87.79	87.79	87.79	87.79	87.79	87.79	87.79	87.79	87.79	87.79
rcv2	10%	54.98	54.76	89.70	87.17	**89.88**	89.70	89.74	57.26	60.33	89.38
	20%	57.65	57.43	90.29	87.27	90.33	**90.53**	90.50	63.26	61.97	90.21
	30%	59.39	59.39	90.53	87.47	90.43	90.14	90.04	67.57	65.59	**91.15**
	50%	64.45	64.14	90.18	87.76	89.78	89.29	89.17	76.10	70.15	**90.70**
	100%	88.87	88.87	88.87	88.87	88.87	88.87	88.87	88.87	88.87	88.87
rcv3	10%	54.81	54.93	88.69	86.72	88.70	88.62	**88.74**	57.90	61.32	88.48
	20%	58.55	58.55	89.34	87.27	89.34	89.60	**89.66**	63.95	64.15	89.52
	30%	60.33	60.33	89.68	86.92	89.43	89.53	89.44	65.98	66.63	**90.14**
	50%	65.80	65.84	89.53	87.02	89.58	89.51	89.38	73.82	71.44	**90.62**
	100%	88.67	88.67	88.67	88.67	88.67	88.67	88.67	88.67	88.67	88.67
rcv4	10%	54.62	54.62	89.94	87.17	**90.17**	90.04	90.11	58.03	60.16	89.87
	20%	57.12	57.04	90.33	87.86	90.56	90.53	**90.59**	61.30	62.74	90.24
	30%	58.58	58.52	90.37	87.91	90.72	90.70	90.66	66.71	65.17	**90.95**
	50%	64.93	64.91	90.55	87.96	90.93	90.94	90.81	75.59	69.09	**91.32**
	100%	89.31	89.31	89.31	89.31	89.31	89.31	89.31	89.31	89.31	89.31
tr41	10%	71.51	72.75	93.33	85.30	90.05	91.70	91.73	76.75	82.72	**93.97**
	20%	77.39	78.84	93.02	88.37	91.24	92.34	92.25	78.84	80.42	**94.73**
	30%	88.61	87.12	95.10	93.73	95.55	95.40	95.29	82.21	84.73	**95.78**
	50%	92.82	92.59	95.44	92.93	95.67	95.52	95.33	93.62	88.04	**95.89**
	100%	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85	93.85

cont.

Data Set	VS in (%)	Accuracy									
		BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (Proposed)
tr45	10%	68.11	68.11	**92.33**	90.04	91.59	91.73	91.68	74.35	80.72	90.44
	20%	77.39	77.39	**92.06**	90.28	92.04	91.59	91.48	78.84	79.42	90.72
	30%	83.76	83.76	92.02	87.97	92.02	**92.17**	92.10	87.82	84.05	92.02
	50%	90.43	90.86	92.17	89.71	92.17	92.02	92.05	91.44	87.39	**92.31**
	100%	88.76	88.76	88.76	88.76	88.76	88.76	88.76	88.76	88.76	88.76
wap	10%	64.80	60.19	84.71	64.35	84.87	84.85	**84.93**	41.47	45.32	84.21
	20%	73.91	70.70	85.12	73.97	**85.83**	85.63	85.70	55.57	49.48	84.94
	30%	77.24	76.15	**85.96**	77.56	85.76	85.89	85.84	66.15	54.48	85.28
	50%	81.08	80.32	**86.53**	85.32	85.89	85.45	85.38	73.58	61.21	85.32
	100%	78.30	78.30	78.30	78.30	78.30	78.30	78.30	78.30	78.30	78.30

than the other methods in 80.06% cases.

The performances of TRL and other term selection techniques, using the accuracy of SVM classifier will now be discussed. It can be seen from Table 6.4 that the performances of SVM are better on the reduced set of terms produced by TRL than the set of all terms for all the data sets except for la1, la2 and rcv3 data sets when 90% terms have been removed. Thus the results show that TRL is able to retrieve significant information from the data. The performances of SVM are not better on the reduced set of terms than the original one for BNS, CE, DF, MI, ODR in most of the cases of Table 6.4. SVM shows better results on the reduced sets of terms produced by CHI, GI, GR and IG than the complete set of terms for all the data sets except la1, la2 and tr41. Note that the proposed method performs better than the other competing methods in most of the cases in Table 6.4.

It can be observed from Table 6.4 that TRL has 360 comparisons with the other methods for 50% to 90% removal of terms from the actual vocabulary. The performance of TRL is better than the other methods in 279 cases and in the remaining 81 cases other methods performed better than TRL. The differences are statistically significant by t-test in 250 out of 279 cases when TRL performed better than the other methods for the level of significance 0.05. The differences are statistically significant in 62 out of 81 cases when other methods performed better than TRL for the same level of significance. Thus TRL performs significantly better than the other methods in 80.12% cases.

Table 6.5: Execution Time (in seconds) of Different Term Selection Techniques

Data Set	BNS	CE	CHI	DF	GI	GR	IG	MI	ODR	TRL (proposed)
fbis	45.58	48.09	50.35	9.42	45.30	52.47	52.26	47.11	45.22	49.44
la1	72.60	74.25	78.35	19.22	72.34	79.54	79.18	73.43	72.52	76.39
la2	72.22	74.34	78.42	19.04	72.14	79.41	79.06	73.27	72.16	76.11
rcv1	46.37	48.51	51.41	10.53	46.08	53.26	53.14	47.18	46.31	52.14
rcv2	46.35	48.50	51.40	10.52	46.08	53.26	53.14	47.18	46.30	52.14
rcv3	46.30	48.47	51.36	10.49	46.03	53.21	53.10	47.14	46.26	52.11
rcv4	46.55	49.10	51.58	11.12	46.26	53.45	53.32	47.36	46.51	52.31
tr41	16.48	15.52	18.55	4.12	16.24	19.58	19.39	17.14	17.02	17.45
tr45	17.15	16.07	19.12	4.33	16.56	20.17	20.01	17.36	17.28	18.12
wap	17.40	16.28	19.33	4.51	17.28	20.46	20.25	17.58	17.49	18.38

6.3.3 Time and Space Complexity of TRL

Algorithm 6 starts with N training documents and each document has n terms. The $P(t)$ and $P(C_i),\ i = 1, 2, ..., m$ values in equation 6.1 can be computed in $O(N)$ time. $TRL_{all}(t_i)$ for each $t_i \in T$ is computed in $O(N)$ time. The *for* loop of step 1 is executed n times. The sorting of $TRL_{all}(t_i)$'s can be done in $O(n \log n)$ time, in worst case. Thus to compute step 1 to step 8 the algorithm takes $(n \times O(N)) + O(n \log n)$ time, i.e., $O(nN + n \log n)$ time. Thus the time complexity of Algorithm 6 is $O(nN + n \log n)$ in worst case.

The data matrix to store N training documents requires $n \times N$ space. Hence the space complexity of Algorithm 6 is $O(nN)$.

DF thresholding is very popular as its computational complexity is approximately linear to the number of documents in the training set [131]. The probability computations in the evaluation function of the other eight methods have a time complexity of $O(nN)$ and space complexity of $O(nN)$ [131]. Thus to select γ terms from the vocabulary these methods takes $O(nN + n \log n)$ time in worst case, like the proposed method. The space complexity of each of these methods is $O(nN)$, same as TRL. Hence TRL has a time and space complexity comparable with the other methods, except DF.

Table 6.5 summarizes the time taken by different term selection algorithms used in the experiments for all the text data sets. The time shown here for each algorithm is the time to assign a score to each term as per the rule of the method. The time to select top γ terms from the entire set of terms and the time to perform text categorization by using either k-NN classifier or SVM classifier are not reported in

Table 6.5, as these times are same for all the methods in the present experimental setup. It can be seen from Table 6.5 that the processing time of TRL is less than the processing times of CHI, GR, IG. The processing time of DF is least among all the methods for each data set as it takes linear time to compute the score for each term. On the other hand the processing times of BNS, CE, GI, MI, ODR are less than the processing time of TRL for each data set. Note that the performance of TRL is better than BNS, CE, GI, MI and ODR.

6.4 Conclusions

Text categorization is a challenging task due to the high number of terms and the sparsity of data. An effective term selection method is thus needed to boost the performance of text categorization. An evaluation function for term selection in text categorization has been introduced in this study. The proposed TRL derives a similarity score between a term and a category and then finds the minimum value over all the categories. All the terms are ranked according to their scores and the top few terms are used for text categorization. The objective of TRL is to develop a subset of terms which consists of only the representative terms of each category. It therefore gives priority to those terms, which occur highly in a particular category.

The results show that the proposed TRL improves the performance of text categorization even after removal of 90% unique terms for all the data sets. It can be seen from the experimental analysis that TRL significantly outperforms the other methods in most of the cases. In real life, the text corpora are very diverse in nature. Hence it should not be claimed that TRL outperforms any other term selection approach for any type of corpus in practice. It may be considered as an alternative approach to the existing methods, as TRL outperforms several existing techniques. It may be noted from the analysis of processing time that the computational cost of the proposed TRL is comparable with the other existing term selection techniques, *viz.* IG, CHI, GR etc. Hence the proposed TRL may easily be applied to any high dimensional real life data set.

Chapter 7

Conclusions and Scope of Further Research

The conclusions drawn from the respective proposed methodologies and the experimental results have been presented in every chapter. In this chapter we compile them to provide an overall scenario of the contributions of the thesis.

Text mining refers to a system that identifies useful information from a huge amount of natural language text. Several specific learning methodologies have been proposed to extract useful information from the text data. In addition to that various existing pattern recognition algorithms have been applied on text data mining for meaningful knowledge discovery. The thesis deals with the supervised and unsupervised methodologies of text mining using the plain text data of English language only. In this thesis some problems have been pointed out about the existing supervised and unsupervised methodologies applied to or designed for text data only. Some new supervised and unsupervised methodologies have been proposed for effective mining of the text data after successfully overcoming those problems. The proposed techniques performed experimentally better than the existing methods of text mining on several well known text data sets from different sources e.g., Ruter newswire, TREC etc. All the methods have been designed for mining text data only, but those methods have the potential to find effective information from any type of data. In future the scope of applicability of the proposed methods will be extended to other fields such as web mining, social network mining etc.

In chapter 2, a new document similarity measure (*named as extensive similarity*) and an agglomerative hierarchical document clustering technique using a new cluster distance measure have been proposed. The clustering method is named as *Clustering Using Extensive Similarity (CUES)*. A histogram thresholding based method is introduced to fix the content similarity threshold. The threshold is used to define extensive similarity (specifically to restrict the low similarity values). The performance of CUES is compared with different partitional and hierarchical clustering algorithms and two types of spectral clustering methods using various well known text data sets in the experimental evaluation. The analysis shows that CUES has performed significantly better than the other methods. Cosine similarity is used in this chapter to find the content similarity between documents, since it can find the content similarity (even) between the documents with different lengths (the same is explained in detail in section 2.1). But one can use any other similarity or dissimilarity measure in the formula of distance function 2.1 to perform CUES as per requirement depending on the type of data sets. It may also be noted that document similarity can be measured by finding semantic relatedness between documents instead of content similarity. In future the performance of CUES will be studied by using semantic similarity between documents. The incoming and outgoing links play an important role in finding the similarity between web pages. The merit of extensive similarity can probably be extended to draw a relation between web pages using their incoming and outgoing links.

It has been observed that CUES has performed very well on several text data sets, but it may suffer from high computational time for some very large data sets. In chapter 3, another document clustering algorithm is proposed. The proposed one has low computational complexity than CUES and the performance of the proposed method is better than CUES and other existing document clustering techniques. A new distance function is proposed to find the distance between two clusters using the extensive similarity measure. The distance function creates the initial baseline clusters and these clusters determine the layout of the actual grouping of the corpus. It has been shown in the experiments that the proposed algorithm is able to find the actual grouping in most of the data sets. The same histogram thresholding based technique proposed in chapter 2 is used here to estimate the value of the content similarity threshold of extensive similarity. As the proposed method has outperformed the other methods, it may be claimed that the thresholding has been done

properly. The proposed technique can be used for clustering the data points of any type of data set.

A tweak on the kNN decision rule for text categorization is proposed in chapter 4. The proposed method is designed to enhance the certainty of a decision. Whenever we are taking a decision based on the supporting evidences regarding any matter in real life, it is necessary to have confidence on the decision, otherwise the decision may cause serious harm. The proposed method is developed along this general notion. From the empirical studies it has become clear that the proposed method has performed significantly better than various other classifiers for text categorization. The proposed algorithm is applied for text categorization, but it can be easily extended for any type of data sets. Moreover, the intuition can be used in any decision making system.

In chapter 5 the extensive similarity is used instead of cosine similarity on the TkNN decision rule for text categorization. The use of extensive similarity enriches the quality of text categorization and thus the proposed extensive similarity based decision rule has performed better than TkNN and the other classifiers. This method once again shows the effectiveness of extensive similarity. It has become clear that extensive similarity is useful for supervised techniques also. In future the merit of extensive similarity will be explored in various other supervised methodologies where similarity or dissimilarity measures play significant role.

In chapter 6, a supervised term selection technique is presented. The aim of the proposed term selection method is to give stress on the representative terms of each category (even if the data is unevenly distributed). Therefore the proposed technique gives high priority to the terms which occur highly in each category irrespective of their occurrence in the entire corpus. The intuition behind the proposed evaluation function is justified in the chapter. The experimental analysis shows that the method has performed better than several other methods. In future this method will be applied on all the proposed supervised and and unsupervised methodologies presented in the previous chapters to enrich their performances by discarding irrelevant or redundant terms from the corpus.

Appendix A

Description of Porter's Stemming Method

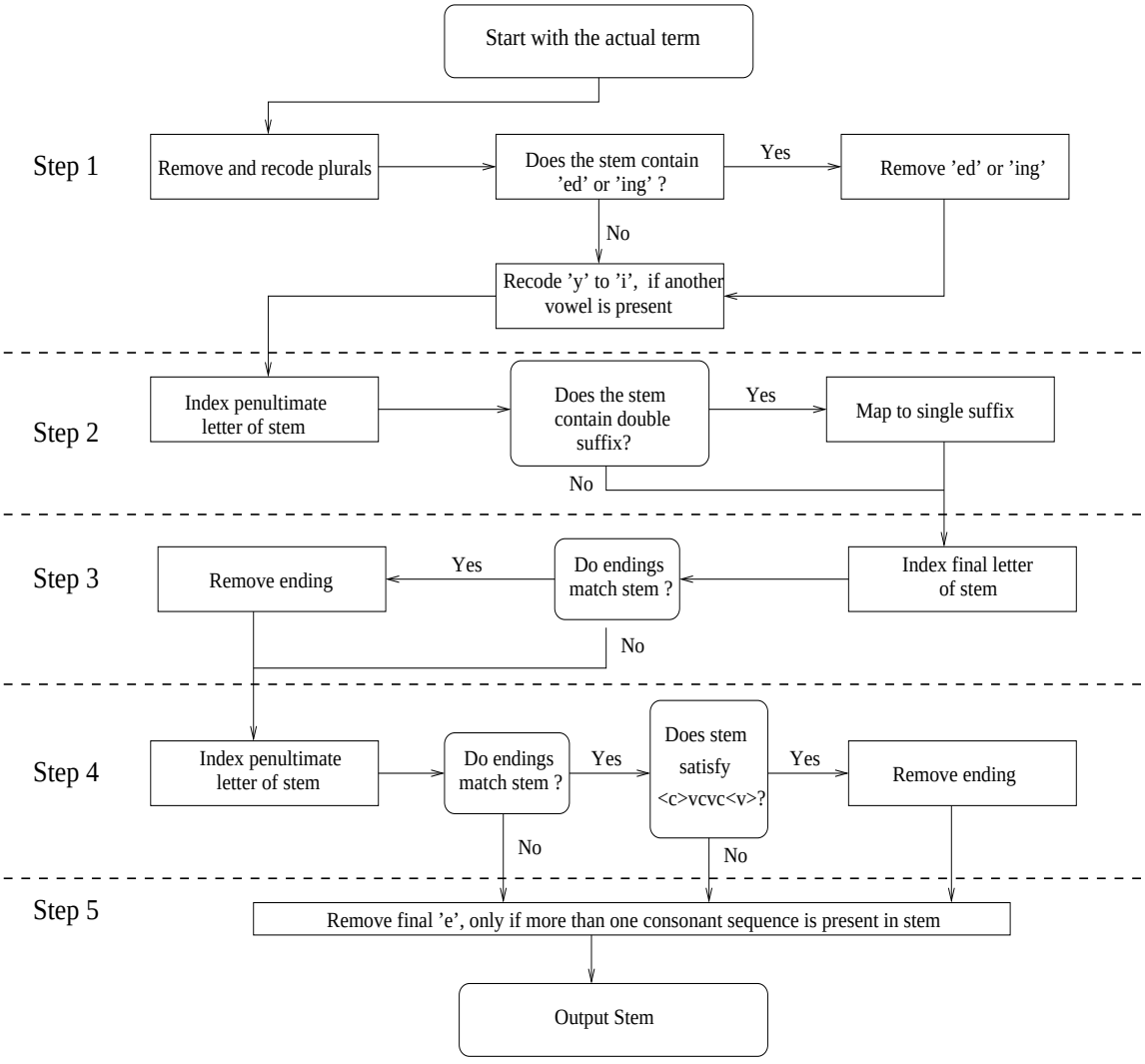

Figure A.1: Steps of the Porter Stemmer Algorithm

Appendix B

Discussion on Implementation of Different Existing Clustering Techniques

The performance of new clustering algorithms proposed in chapter 2 and chapter 3 are compared with bisecting k-means clustering, k-means clustering, buckshot clustering, k nearest neighbors based clustering, single-link and average link hierarchical clustering, simple and kernel based spectral clustering and non negative matrix factorization based clustering techniques using f-measure and normalized mutual information in each of these chapters. All of these techniques are clearly described in chapter 1 respectively in section 1.3.2.2, section 1.3.2.1, section 1.3.2.3, section 1.3.2.4 of the previous version of the thesis. The experimental setup is described in section 2.4.1 of chapter 2 and section 3.3 of chapter 3. It has been clearly described in these sections how different parameters of each of the competing algorithms are chosen for fair comparison with the proposed techniques.

The text data sets used in the experiments are nothing but collection of documents, where each document contains a set of terms. The clustering algorithms can be performed on these data sets after creating the term-document matrix (i.e., tf-idf matrix) from the raw text data. The tf-idf matrix of each text data set has been developed using C++ programming language on a Linux workstation. The codes for some of the clustering algorithms mentioned above are available on either Mat-

lab[1] or R[2] statistical toolbox. As far as the knowledge of the author goes, similar codes are not available publicly on any reliable source. It may be noted that the codes available for different clustering algorithms in Matlab or R can only generate the clusters. No code is available for implementing the cluster validity index (e.g., f-measure) in Matlab or R to evaluate the quality of the clusters. Hence, the codes of every aspect of each method have been written in C++ programming language.

The codes written for different clustering techniques have been executed on some well known data sets, downloaded from UCI repository[3] prior to applying them on text data sets. The names of the data sets are ecoli, glass, iris, spambase and yeast. The number of data points and number of classes of these data sets are mentioned in Table B.1 and Table B.2. The desired number of clusters produced by different algorithms are given in these tables. The number of clusters are equal to the number of actual classes of each data set. The codes available in Matlab toolbox for k-means clustering, single-link and average link hierarchical clustering and kernel based spectral clustering have also been executed for each data set. The non negative matrix factorization based clustering technique is executed in R for each data set. The results are reported using f-measure and Normalized Mutual Information (NMI) as cluster validity measure respectively in Table B.1 Table B.2. The codes for implementation of f-measure and NMI have been manually verified on iris data set.

It can be observed from these tables that the developed code and the code available in Matlab or R for each algorithm are producing more or less same results for each data set. The performance of the developed codes and the existing codes are equal for single link, complete link and spectral clustering algorithms for all the data sets in Table B.1 and Table B.2. The performance of the developed codes and the existing codes are close to each other for the other algorithms i.e., k-means and NMF based clustering techniques in Table B.1 and Table B.2. Sometimes the performance of the developed codes are slightly better than the existing codes and for the other cases the existing codes are showing better results than the developed ones for k-means and NMF based clustering algorithms. The reason is that these

[1]http://in.mathworks.com/
[2]http://www.rstudio.com/
[3]http://archive.ics.uci.edu/ml/

Table B.1: Performance of Different Clustering Techniques on Various UCI Data Sets using F-measure

Data Sets	ND[4]	NC	NCT	F-measure									
				KM		SLHC		ALHC		SCK		NMF	
				AVC[5]	MVC[6]	AVC	MVC	AVC	MVC	AVC	RVC[7]	AVC	MVC
Ecoli	336	8	8	0.635	0.630	0.405	0.405	0.405	0.405	0.761	0.761	0.520	0.516
Glass	214	7	7	0.525	0.530	0.397	0.397	0.397	0.397	0.547	0.547	0.320	0.318
Iris	150	3	3	0.840	0.846	0.494	0.494	0.494	0.494	0.841	0.841	0.687	0.683
Spam-base	4601	2	2	0.625	0.618	0.679	0.679	0.679	0.679	0.585	0.585	0.610	0.619
Yeast	1484	10	10	0.208	0.204	0.351	0.351	0.351	0.351	0.294	0.294	0.205	0.198

[4] ND stands for Number of Data. NC stands for Number of Classes. NCT stands for Number of Clusters developed by each algorithm. KM, SLHC, ALHC, SCK, NMF stand for - k-Means clustering, Single-Link Hierarchical Clustering, Average-Link Hierarchical Clustering, Spectral Clustering using Kernel and Non-negative Matrix Factorization based clustering techniques respectively. [5] AVC stands for Author Version of Code i.e., the code is implemented by the author [6] MVC stands for Matlab Version of Code i.e., the code is available in Matlab toolbox [7] RVC stands for R Version of Code i.e., the code is available in R toolbox

Table B.2: Performance of Different Clustering Techniques on Various UCI Data Sets using NMI

Data Sets	ND[8]	NC	NCT	Normalized Mutual Information (NMI)									
				KM		SLHC		ALHC		SCK		NMF	
				AVC	MVC	AVC	MVC	AVC	MVC	AVC	RVC[7]	AVC	MVC
Ecoli	336	8	8	0.588	0.582	0.448	0.448	0.448	0.448	0.695	0.695	0.402	0.399
Glass	214	7	7	0.434	0.438	0.403	0.403	0.403	0.403	0.436	0.436	0.084	0.079
Iris	150	3	3	0.831	0.836	0.315	0.315	0.315	0.315	0.766	0.766	0.530	0.528
Spam-base	4601	2	2	0.012	0.008	0.054	0.054	0.054	0.054	0.093	0.093	0.002	0.006
Yeast	1484	10	10	0.040	0.035	0.222	0.222	0.222	0.222	0.045	0.045	0.034	0.029

[8] All the acronyms are same as mentioned in Table B.1

algorithms follow some random heuristics at some stage. It may be noted that these algorithms are executed 10 times to reduce the effect of random initialization of seed points and for each execution they have been iterated 100 times to reach a solution (if they have not converged automatically) for both Matlab implementation and the implementation developed by the author.

Hence it may be implied from the above observations that the codes developed by the author are producing as optimized results as the existing implementations. Regarding the other clustering techniques like buckshot, bisecting k-means and kNN clustering, the codes for their implementation on any reliable source could not be found.

Appendix C

Discussion on Term Relatedness

In this section we shall justify the claims that have been made in chapter 6 for all possible TRL values between a term and a category. Let $t_1, t_{2a}, t_{2b}, t_{2c}, t_{2d}$ and t_{2e} be the terms obtained respectively by case 1, case 2(a), case 2(b), case 2(c), case 2(d) and case 2(e) and all these terms belong to the same category i.e., C_i. Let $t_{3a}, t_{3b}, t_{3c}, t_{3d}$ and t_{3e} be the terms obtained respectively by case 3(a), case 3(b), case 3(c), case 3(d) and case 3(e) and all these terms belong to the same C_i.

Claim 1: t_1 gets highest preference among all the terms.

Justification: The minimum TRL value of a term t and a category c is 0 and it can be obtained only when $P(t, C_i) = P(t) = P(C_i)$ and for all the other cases the TRL values are greater than 0. t_1 is such a term where $TRL(t_1, C_i) = 0$. Note that the minimum TRL value indicates the optimum term. The terms of all the sub-cases of case 2 and case 3 can be obtained by using either TF or TRF or TCR. It has been explained in section 6.2 that the values of TF, TRF and TCR lie between $(0, 1)$ whenever they are applied to find TRL between a term and a category. Thus the TRL values of all the sub-cases of case 2 and case 3 lie between $(0, 1)$. Hence t_1 gets highest preference among all the terms.

Claim 2(a): t_{2a} gets higher preference than t_{2b} by TRL when $P(t_{2a}, C_i) = P(t_{2b}, C_i)$.

Justification: As $1 + P(t_{2b}) > 1 + P(C_i)$ we have

$$\frac{1 + P(t_{2a}, C_i)}{1 + P(C_i)} = \frac{1 + P(t_{2b}, C_i)}{1 + P(C_i)} > \frac{1 + P(t_{2b}, C_i)}{1 + P(t_{2b})}$$

$$\therefore TRL(t_{2a}, C_i) = 1 - \frac{1 + P(t_{2a}, C_i)}{1 + P(C_i)} \times E(C_i) \;<\; 1 - \frac{1 + P(t_{2b}, C_i)}{1 + P(t_{2b})} \times E(C_i)$$
$$= TRL(t_{2b}, C_i)$$

Hence the terms of case 2(a) gets higher preference than the terms of case 2(b) for the same $P(C_i)$ and $P(t, C_i)$ values.

Claim 2(b): If $P(t_{2b}) = P(t_{2c})$ then t_{2b} gets higher preference than t_{2c} by TRL.

Justification: It can be seen that $P(t_{2b}, C_i) > P(t_{2c}, C_i)$, since they both exist in the same category C_i and $P(t_{2b}, C_i) = P(C_i)$ and $P(t_{2c}, C_i) < P(C_i)$.

$$\begin{aligned}
\therefore TRL(t_{2b}, C_i) &= 1 - \frac{1 + P(t_{2b}, C_i)}{1 + P(t_{2b})} \times E(C_i) \\
&< 1 - \frac{P(t_{2b}, C_i)}{P(t_{2b})} \times E(C_i) = 1 - \frac{P(t_{2b}, C_i)}{P(t_{2c})} \times E(C_i) \\
&< 1 - \frac{P(t_{2c}, C_i)}{P(t_{2c})} \times E(C_i) \\
&= 1 - \frac{P(t_{2c}) - P(t_{2c}, \overline{C_i})}{P(t_{2c})} \times E(C_i) \\
&= TRL(t_{2c}, C_i)
\end{aligned}$$

Hence for the same $P(C_i)$ and $P(t)$ values the terms of case 2(b) get higher preference than the terms of case 2(c).

Claim 2(c): If $P(t_{2b}) = P(t_{2d})$ then t_{2b} gets higher preference than t_{2d} by TRL.

Justification: It can be seen that $P(t_{2b}, C_i) > P(t_{2d}, C_i)$, since they both exist in the same category C_i and $P(t_{2b}, C_i) = P(C_i)$ and $P(t_{2d}, C_i) < P(C_i)$.

$$\begin{aligned}
\therefore TRL(t_{2b}, C_i) &= 1 - \frac{1 + P(t_{2b}, C_i)}{1 + P(t_{2b})} \times E(C_i) \\
&< 1 - \frac{P(t_{2b}, C_i)}{P(t_{2b})} \times E(C_i) = 1 - \frac{P(C_i)}{P(t_{2b})} \times E(C_i) \\
&< 1 - \frac{P(C_i) - P(t_{2d}, C_i)}{P(t_{2d}) - P(t_{2d}, C_i)} \times E(C_i)
\end{aligned}$$

$$\Rightarrow TRL(t_{2b}, C_i) < 1 - \frac{P(C_i) - P(t_{2d}, C_i)}{P(t_{2d}) - P(t_{2d}, C_i)} \times \frac{P(t_{2d}) - P(t_{2d}, \overline{C_i})}{P(t_{2d})} \times E(C_i)$$
$$= TRL(t_{2d}, C_i)$$

as the multiplication of two fractional values is less than their individual values (e.g., $0.4 \times 0.3 = 0.12$ is less than both 0.4 and 0.3). Hence for the same $P(C_i)$ and $P(t)$ values the terms of case 2(b) get higher preference than the terms of case 2(d).

Claim 2(d): t_{2a} gets higher preference than t_{2e} by TRL, if $P(t_{2a}) = P(t_{2e})$.

Justification: Note that $P(t_{2a}) = P(t_{2a}, C_i)$ and $P(t_{2a}, C_i) > P(t_{2e}, C_i)$ as both of t_{2a} and t_{2e} belong to C_i.

$$\therefore TRL(t_{2a}, C_i) = 1 - \frac{1 + P(t_{2a}, C_i)}{1 + P(C_i)} \times E(C_i)$$
$$< 1 - \frac{P(t_{2a}, C_i)}{P(C_i)} \times E(C_i) = 1 - \frac{P(t_{2e})}{P(C_i)} \times E(C_i)$$
$$< 1 - \frac{P(t_{2e}) - P(t_{2e}, C_i)}{P(C_i) - P(t_{2e}, C_i)} \times E(C_i)$$
$$< 1 - \frac{P(t_{2e}) - P(t_{2e}, C_i)}{P(C_i) - P(t_{2e}, C_i)} \times \frac{P(t_{2e}) - P(t_{2e}, \overline{C_i})}{P(t_{2e})} \times E(C_i)$$
$$= TRL(t_{2e}, C_i)$$

as the multiplication of two fractional values is less than their individual values. Thus the terms of case 2(a) get higher preference than the terms of case 2(d) for the same $P(C_i)$ and $P(t, C_i)$ values.

Claim 2(e): t_{2c} gets higher preference than t_{2d} by TRL when $P(t_{2c}, C_i) = P(t_{2d}, C_i)$.

Justification: $TRL(t_{2c}, C_i) = 1 - \frac{P(t_{2c}) - P(t_{2c}, \overline{C_i})}{P(t_{2c})} \times E(C_i)$
$$< 1 - \frac{P(t_{2d}) - P(t_{2d}, \overline{C_i})}{P(t_{2d})} \times E(C_i)$$
$$\left[\because P(t_{2c}, C_i) = P(t_{2d}, C_i) \text{ and } P(t_{2d}) > P(t_{2c})\right]$$

$$\Rightarrow TRL(t_{2c}, C_i) < 1 - \frac{P(C_i) - P(t_{2e}, C_i)}{P(t_{2e}) - P(t_{2e}, C_i)} \times \frac{P(t_{2d}) - P(t_{2d}, \overline{C_i})}{P(t_{2d})} \times E(C_i)$$
$$= TRL(t_{2d}, C_i)$$

since the multiplication of two fractional values is less than their individual values. Therefore the terms of case 2(c) get higher preference than the terms of case 2(d) for the same $P(C_i)$ and $P(t, C_i)$ values.

Claim 3(a): $t_{3a}, t_{3b}, t_{3c}, t_{3d}$ **and** t_{3e} **get lower preference than** $t_{2a}, t_{2b}, t_{2c}, t_{2d}$ **and** t_{2e} **in** C_i**.**

Justification: It has been justified in Claim 2(d) that t_{2a} gets higher preference than t_{2e}, if $P(t_{2a}) = P(t_{2e})$. Claim 2(b) states that t_{2b} gets higher preference than t_{2c}, if $P(t_{2b}) = P(t_{2c})$. Hence we have to justify the following statements to establish claim 3(a).

(i) t_{3a} **gets lower preference than** t_{2c} **by TRL.**

$$TRL(t_{2c}, C_i) = 1 - \frac{P(t_{2c}) - P(t_{2c}, \overline{C_i})}{P(t_{2c})} \times E(C_i)$$
$$TRL(t_{3a}, C_i) = 1 - \frac{1 + P(t_{3a}, C_i)}{1 + P(C_i)} \times E(C_i)$$

$P(C_i) >> P(t_{3a})$ and $P(t_{3a}) = P(t_{3a}, C_i)$ then $1 + P(t_{3a}, C_i) << 1 + P(C_i)$ and $\frac{1 + P(t_{3a}, C_i)}{1 + P(C_i)} << 1$ i.e., close to 0. As a result $TRL(t_{3a}, C_i)$ is close to 1.

$P(t_{2c}, C_i) < P(C_i) = P(t_{2c})$ and $P(t_{2c})$ is close to $P(t_{2c}, C_i)$. Therefore $\frac{P(t_{2c}) - P(t_{2c}, \overline{C_i})}{P(t_{2c})}$ is close to 1. Hence $TRL(t_{2c}, C_i) < TRL(t_{3a}, C_i)$ and thus t_{3a} gets lower preference than t_{2c} by TRL.

(ii) t_{3a} **gets lower preference than** t_{2d} **by TRL.**

$$TRL(t_{2d}, C_i) = 1 - \frac{P(C_i) - P(t_{2d}, C_i)}{P(t_{2d}) - P(t_{2d}, C_i)} \times \frac{P(t_{2d}) - P(t_{2d}, \overline{C_i})}{P(t_{2d})} \times E(C_i)$$

$$TRL(t_{3a}, C_i) = 1 - \frac{1 + P(t_{3a}, C_i)}{1 + P(C_i)} \times E(C_i)$$

It has been shown that $TRL(t_{3a}, C_i)$ is close to 1. $P(t_{2d}, C_i) < P(C_i) < P(t_{2d})$ and $P(t_{2d}, C_i)$ and $P(t_{2d})$ both are close to $P(C_i)$. Thus $P(t_{2d})$ is close to $P(t_{2d}, C_i)$. Therefore $\frac{P(t_{2d}) - P(t_{2d}, \overline{C_i})}{P(t_{2d})}$ and $\frac{P(C_i) - P(t_{2d}, C_i)}{P(t_{2d}) - P(t_{2d}, C_i)}$ both are close to 1 and $TRL(t_{2d}, C_i)$ becomes close to 0. Hence $TRL(t_{2d}, C_i) < TRL(t_{3a}, C_i)$ and t_{3a} gets lower preference than t_{2c}.

(iii) t_{3a} gets lower preference than t_{2e} by TRL.

$$TRL(t_{2e}, C_i) = 1 - \frac{P(t_{2e}) - P(t_{2e}, C_i)}{P(C_i) - P(t_{2e}, C_i)} \times \frac{P(t_{2e}) - P(t_{2e}, \overline{C_i})}{P(t_{2e})} \times E(C_i)$$

$$TRL(t_{3a}, C_i) = 1 - \frac{1 + P(t_{3a}, C_i)}{1 + P(C_i)} \times E(C_i)$$

$P(t_{2e}, C_i) < P(t_{2e}) < P(C_i)$ and $P(t_{2e}, C_i)$ and $P(t_{2e})$ both are close to $P(C_i)$. Thus $P(t_{2e})$ is close to $P(t_{2e}, C_i)$. Therefore $\frac{P(t_{2e}) - P(t_{2e}, \overline{C_i})}{P(t_{2e})}$ and $\frac{P(t_{2e}) - P(t_{2e}, C_i)}{P(C_i) - P(t_{2e}, C_i)}$ both are close to 1 and as a result $TRL(t_{2e}, C_i)$ is close to 0. Note that $TRL(t_{3a}, C_i)$ is close to 1 and thus $TRL(t_{2e}, C_i) < TRL(t_{3a}, C_i)$

Claim 3(b): t_{3a} gets higher preference than t_{3b} by TRL when $P(t_{3a}, C_i) = P(t_{3b}, C_i)$.

Claim 3(c): If $P(t_{3b}) = P(t_{3c})$ then t_{3b} gets higher preference than t_{3c} by TRL.

Claim 3(d): If $P(t_{3b}) = P(t_{3d})$ then t_{3b} gets higher preference than t_{3d} by TRL.

Claim 3(e): t_{3a} gets higher preference than t_{3e} by TRL, if $P(t_{3a}) = P(t_{3e})$.

Claim 3(f): t_{3c} gets higher preference than t_{3d} by TRL, if $P(t_{3c}, C_i) = P(t_{3d}, C_i)$.

Justification: Claim 3(b) can be justified in the same way as claim 2(a) has been justified. Claim 3(c), claim 3(d), claim 3(e) and claim 3(f) can be justified respectively in the same way as claim 2(b), claim 2(c), claim 2(d) and claim 2(e) have been justified.

Bibliography

[1] C. C. Aggarwal and C. Zhai. A survey of text clustering algorithms. *Mining Text Data*, pages 77–128, 2012.

[2] H. Al-Mubaid and S. A. Umair. A new text categorization technique using distributional clustering and learning logic. *IEEE Transactions on Knowledge and Data Engineering*, 18(9):1156–1165, 2006.

[3] N. Ali and N. Ibrahim. Porter stemming algorithm for semantic checking. In *Proceedings of the International Conference on Communications and Information Technology (ICCIT'2012)*, pages 253–258, 2012.

[4] C. J. Alpert and S. Z. Yao. Spectral partitioning: The more eigenvectors, the better. In *Proceedings of the ACM/IEEE Design Automation Conference*, pages 195–200, 1995.

[5] N. O. Andrews and E. A. Fox. Recent developments in document clustering. Technical report, Verginia Tech., USA, 2007.

[6] D. Arthur, B. Manthey, and H. Roglin. Smoothed analysis of the k-means method. *Journal of ACM*, 58(5), 2011.

[7] S. Banerjee, K. Ramanathan, and A. Gupta. Clustering short texts using wikipedia. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'07*, pages 787–788, 2007.

[8] L. Baoli, L. Qin, and Y. Shiwen. An adaptive k-nearest neighbor text categorization strategy. *ACM Transactions on Asian Language Information Processing*, 3(4):215–226, 2004.

[9] T. Basu and C. A. Murthy. Effective text classification by a supervised feature selection approach. In *Proceedings of the IEEE International Conference*

on Data Mining Workshops, ICDMW'12, pages 918–925, Brussels, Belgium,, 2012.

[10] T. Basu and C. A. Murthy. A feature selection method for improved document classification. In *Proceedings of the International Conference on Advanced Data Mining and Applications, ADMA'12*, pages LNCS 7713, 296–305, Nanjing, China, 2012.

[11] T. Basu and C. A. Murthy. Cues: A new hierarchical approach for document clustering. *Journal of Pattern Recognition Research*, 8(1):66–84, 2013.

[12] T Basu and C. A. Murthy. A similarity based supervised decision rule for qualitative improvement of text categorization. *communicated to Fundamenta Informaticae*, April, 2014.

[13] T. Basu and C. A. Murthy. A supervised term selection technique for effective text categorization. *communicated to Journal of Classification, Springer*, April, 2014.

[14] T. Basu and C. A. Murthy. A similarity assessment technique for effective grouping of documents. *under revision in Information Sciences*, March, 2013.

[15] T. Basu and C.A. Murthy. Towards enriching the quality of k-nearest neighbor rule for document classification. *International Journal of Machine Learning and Cybernetics (in Press)*, DOI: 10.1007/s13042-013-0177-1, 2013.

[16] T. Basu, C.A. Murthy, and H. Chakraborty. A tweak on k-nearest neighbor decision rule. In *Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV'13*, pages 929–935, Las Vegas, USA, 2012.

[17] P. Berkhin. Survey of clustering data mining techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

[18] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

[19] D. Boley, M. Gini, R. Gross, E. H. Han, K. Hastings, G. Karypis, V. Kumar, B. Mobasher, and J. Moore. Document categorization and query generation

on the world wide web using webace. *Journal Artificial Intelligence Review - Special Issue on Data Mining on The Internet*, 3(5-6):365–391, 1999.

[20] R. G. Brown. *Smoothing, Forecasting and Prediction of Discrete Time Series*. Prentice-Hall, 1962.

[21] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[22] D. Cai, X. He, and J. Han. Document clustering using locality preserving indexing. *IEEE Transaction on Knowledge and Data Engineering*, 17(12):1624–1637, 2005.

[23] T. H. Cao, T. M. Tang, and C. K. Chau. Text clustering with named entities. *Data Mining: Foundations and Intelligent Paradigms*, 23:267–287, 2012.

[24] J. J. Carlson, M. R. Muguira, J. B. Jordan, G. M. Flachs, and A. K. Peterson. Final report: Weighted neighbor data mining. Technical report, SANDIA Report, SAND 2000-3122, 2000.

[25] C. Carpineto, S. Osinski, G. Romano, and D. Weiss. A survey of web clustering engines. *ACM Computing Surveys*, 41(3), 2009.

[26] C. C. Chang and C. J. Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):1–27, 2011.

[27] J. Chen, H. Huanga, S. Tiana, and Y. Qua. Feature selection for text classification with naive bayes. *Expert Systems with Applications*, 36(3):5432–5435, 2009.

[28] L. Chen, G. Guo, and K. Wang. Class dependent projection based method for text categorization. *Pattern Recognition Letters*, 32(11):1493–1501, 2011.

[29] H. Chim and X. Deng. A new suffix tree similarity measure for document clustering. In *Proceedings of the International Conference on World Wide Web, WWW'07*, pages 121–130, 2007.

[30] G. G. Chowdhury. Natural language processing. *Annual Review of Information Science and Technology*, 37:51–90., 2003.

[31] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[32] D. R. Cutting, D. R. Karger, J. O. Pedersen, and J.W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'93*, pages 126–135, 1993.

[33] B. V. Dasarathy. *Nearest Neighbor NN Norms: NN Pattern Classification Techniques*. McGraw-Hill Computer Science Series. IEEE CS Press, 1991.

[34] B. V. Dasarathy and B. V. Sheela. Visiting nearest neighbors: A survey of nearest neighbour classification techniques. In *Proceedings of the International Conference on Cybernetics and Society*, pages 630–635, 1977.

[35] A. Dasgupta, P. Drineas, B. Harb, V. Josifovski, and M. W. Mahnoney. Feature selection methods for text classification. In *Proceedings of International Conference on Knowledge Discovery and Data Mining, SIGKDD'07*, pages 230–239, San Jose, USA, 2007.

[36] S. Dasgupta and V. Ng. Towards subjectifying text clustering. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'10*, pages 483–490, NY, USA, 2010.

[37] M. Dash and H. Liu. Feature selection for clustering. In *Proceedings of the Pacific Asia Conference on Knowledge Discovery and Data Mining Workshop, PAKDD'00*, pages 110–121, 2000.

[38] F. Debole and F. Sebastiani. Supervised term weighting for automated text categorization. In *Proceedings of the ACM Symposium On Applied Computing*, pages 784–788, Melbourne, Australia, 2003.

[39] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice Hall, first edition, 1982.

[40] I. S. Dhillon, S. Mallela, and R. Kumar. A divisive information-theoretic feature clustering algorithm for text classification. *Journal of Machine Learning Research*, 3:1265–1287, 2003.

[41] A. Dhurandhar and A. Dobra. Probabilistic characterization of nearest neighbor classifiers. *International Journal of Machine Learning and Cybernetics*, 4(4):259–272, 2012.

[42] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. J. Wiley and Sons, 1973.

[43] S. A. Dudani. The distance weighted k nearest neighbor rule. *IEEE Transactions on Systems, Man, Cybernetics, Part A*, SMC-6(4):325–327, 1976.

[44] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD'96)*, pages 226–231, 1996.

[45] G. Felici, F. Sun, and K. Truemper. A method for controlling errors in two-class classification. In *Proceedings of Computer Software and Applications Conference, COMPSAC'99*, pages 186–191, 1999.

[46] G. Feng, J. Guo, B. Y. Jing, and L. Hao. A bayesian feature selection paradigm for text classification. *Information Processing and Management*, 48(2):283–302, 2012.

[47] M. Filipponea, F. Camastrab, F. Masullia, and S. Rovettaa. A survey of kernel and spectral methods for clustering. *Pattern Recognition*, 41(1):176–190, 2008.

[48] E. Fix and J. L. Hodges. Discriminatory analysis, non-parametric discrimination: Consistency properties. Technical report, 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.

[49] G. Forman. An extensive empirical study of feature selection metrics for text categorization. *The Journal of Machine Learning Research*, 3(1):1289–1305, 2003.

[50] W. B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, 1992.

[51] J. H. Friedman. Flexible metric nearest neighbor classification. Technical report, Dept. of Statistics, Stanford University, USA, 1994.

[52] J.H. Friedman. On bias, variance, 0-1 loss, and the curse of dimensionality. *Data Mining and Knowledge Discovery*, 1:55–77, 1997.

[53] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. New York Academic Press, 1990.

[54] K. Fukunaga and L. D. Hostetler. K-nearest neighbor bayes risk estimation. *IEEE Transactions on Information Theory*, 21(3):285–293, 1975.

[55] L. Galavotti, F. Sebastiani, and M. Simi. Feature selection and negative evidence in automated text categorization. In *Proceedings of the Knowledge Discovery and Data Mining Workshop on Text Mining, KDD'00*, 2000.

[56] A. K. Ghosh. On nearest neighbor classification using adaptive choice of k. *Journal of Computational and Graphical Statistics*, 16(2):482–502, 2007.

[57] C. A. Glasbey. An analysis of histogram-based thresholding algorithms. *Graphical Models and Image Processing*, 55(6):532–537, 1993.

[58] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer. Knn model-based approach in classification. In *On the Move to Meaningful Internet Systems, LNCS vol. 2888*, pages 986–996. Springer, 2003.

[59] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer. Using knn model for automatic text categorization. *Soft Computing*, 10(5):423–430, 2006.

[60] K. M. Hammouda and M. S. Kamel. Efficient phrase-based document indexing for web document clustering. *IEEE Transaction on Knowledge and Data Engineering*, 16:1279–1296, 2004.

[61] J. A. Hartigan and M. A. Wong. A k-means clustering algorithm. *Journal of the Royal Statistical Society (Applied Statistics)*, 28(1):100–108, 1979.

[62] A. Hotho, A. Nurnberger, and G. Paa. A brief survey of text mining. *LDV Forum - GLDV Journal for Computational Linguistics and Language Technology*, 20(1):19–62, 2005.

[63] C. W. Hsu, C. C. Chang, and C. J. Lin, editors. *A Practical Guide to Support Vector Classification*. http://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf, 2006.

[64] A. Huang. Similarity measures for text document clustering. In *Proceedings of the New Zealand Computer Science Research Student Conference, Christchurch, New Zealand*, pages 49–56, 2008.

[65] A. Huang, D. Milne, E. Frank, and I. H. Witten. Clustering documents with active learning using wikipedia. In *Proceedings of the IEEE International Conference on Data Mining, ICDM'08*, pages 839–844, 2008.

[66] T. Huang, Y. Yua, G. Guo, and K. Li. A classification algorithm based on local cluster centers with a few labeled training examples. *Knowledge-Based Systems*, 23(6):563–571, 2010.

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

[68] L. Jiang, Z. Cai, D. Wang, and H. Zhang. Bayesian citation-knn with distance weighting. *International Journal of Machine Learning and Cybernetics*, 5(2):193–199, 2014.

[69] L. Jing, M. K. Ng, and J. Z. Huang. Knowledge based vector space model for text clustering. *Knowledge and Information Systems*, 25(1):35–55, 2010.

[70] A. G. Jivani. A comparative study of stemming algorithms. *International Journal of Computer Technology and Applications*, 2(6):1930–1938, 2011.

[71] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning, ECML'98*, pages 137–142, Berlin, Germany, 1998.

[72] G. Karypis and E. H. Han. Centroid-based document classification: Analysis and experimental results. In *Proceedings of the Fourth European Conference on the Principles of Data Mining and Knowledge Discovery, PKDD'00*, pages 424–431, Lyon, France, 2000.

[73] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley and Sons, 1990.

[74] D. Koller and M. Sahami. Toward optimal feature selection. In *Proceedings of the International Conference on Machine Learning, ICML'96*, pages 284–292, 1996.

[75] H. P. Kriegel and M. Pfeifle. Density based clustering of uncertain data. In *Proceedings of the ACM International Conference on Knowledge Discovery and Data Mining (KDD'05)*, pages 672–677, 2005.

[76] W. Lam and C. Y. Ho. Using a generalized instance set for automatic text categorization. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'98*, pages 81–89, 1998.

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

[78] A. N Langville, C. D. Meyer, and R. Albright. Initializations for the non-negative matrix factorization. In *Proceedings of the Conference on Knowledge Discovery from Data, KDD'06*, 2006.

[79] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. *In Advances in Neural Information Processing Systems*, 13:556–562, 2001.

[80] E. L. Lehmann. *Testing of Statistical Hypotheses*. New York: John Wiley, 1976.

[81] E. Leopold and J. Kindermann. Text categorization with support vector machines. how to represent texts in input space? *Machine Learning*, 46(1-2):423–444, 2002.

[82] D. D. Lewis, R. E. Shapire, J. P. Callan, and R. Papka. Training algorithms for linear text classifiers. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR'96*, pages 298–306, 1996.

[83] S. Li, R. Xia, C. Zong, and C. Huang. A framework of feature selection methods for text categorization. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2*, pages 692–700, Stroudsburg, PA, USA, 2009.

[84] R. F. Ling. A probability theory of cluster analysis. *Journal of the American Statistical Association*, 68(341):159–164, 1973.

[85] B. Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Springer, 2007.

[86] T. Liu, S. Liu, Z. Chen, and W. Ma. An evaluation on feature selection for text clustering. In *Proceedings of the International Conference on Machine Learning, ICML'03*, pages 488–495, 2003.

[87] X. Liu, X. Yong, and H. Lin. An improved spectral clustering algorithm based on local neighbors in kernel space. *Computer Science and Information Systems*, 8(4):1143–1157, 2011.

[88] P. Majumder, M. Mitra, S. K. Parui, G. Kole, P. Mitra, and K. Datta. Yass: Yet another suffix stripper. *ACM Transactions on Information Systems (TOIS)*, 25(4):18–38, 2007.

[89] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, New York, 2008.

[90] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[91] J. R. Millar, G. L. Peterson, and M. J. Mendenhall. Document clustering and visualization with latent dirichlet allocation and self-organizing maps. In *Proceedings of the Twenty-Second International FLAIRS Conference*, pages 69–74. AAAI press, 2009.

[92] D. Mladenic and M. Grobelnik. Feature selection on hierarchy of web documents. *Decision Support Systems - Web Retrieval and Mining*, 35(1):45–87, 2003.

[93] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.

[94] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Proceedings of Neural Information Processing Systems (NIPS'01)*, pages 849–856, 2001.

[95] J. Novovicova and A. Malik. Information-theoretic feature selection algorithms for text categorization. In *Proceedings of the International Joint Conference on Neural Networks*, pages 3272–3277, Montreal, Canada, 2005.

[96] N. Oikonomakou and M. Vazirgiannis. A review of web document clustering approaches. *Data Mining and Knowledge Discovery Handbook*, 6:931–948, 2010.

[97] T. Pahikkala, A. Airola, F. Gieseke, and O. Kramer. Unsupervised multi-class regularized least-squares classification. In *Proceedings of the IEEE International Conference on Data Mining, ICDM'12*, pages 585–594, 2012.

[98] T. Pahikkala, S. Pyysalo, J. Boberg, J. Jarvinen, and T. Salakoski. Matrix representations, linear transformations, and kernels for disambiguation in natural language. *Machine Learning*, 74(2):133–158, 2009.

[99] T. Pahikkala, S. Pyysalo, F. Ginter, J. Boberg, J. Jarvinen, and T. Salakoski. Kernels incorporating word positional information in natural language disambiguation tasks. In *Proceedings of the Eighteenth International Florida Artificial Intelligence Research Society Conference*, pages 442–447. AAAI Press, 2005.

[100] C. D. Paice. An evaluation method for stemming algorithms. In *Proceedings of the International conference on Research and development in Information Retrieval, SIGIR'94*, pages 42–50, 1994.

[101] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[102] P. Pantel and D. Lin. Document clustering with committees. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'02*, pages 199–206, 2002.

[103] I. Pilaszy. Text categorization and support vector machines. In *Proceedings of the Sixth International Symposium of Hungarian Researchers on Computational Intelligence*, 2005.

[104] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[105] Z. Qiu, B. Wu, B. Wang, C. Shi, and L. Yu. Collapsed gibbs sampling for latent dirichlet allocation on spark. *Journal of Machine Learning Research*, Workshop and Conference Proceedings 36:17–28, 2014.

[106] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[107] C. R. Rao, S. K. Mitra, A. Matthai, and K. G. Ramamurthy, editors. *Formulae and Tables for Statistical Work*. Statistical Publishing Society, Calcutta, 1966.

[108] C. J. Van Rijsbergen. *Information Retrieval*. Butterworths, London, Second Edition, 1979.

[109] B. D Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge, 1996.

[110] T. Salakoski, F. Ginter, S. Pyysalo, and T. Pahikkala, editors. *Advances in Natural Language Processing*. LNCS 4139, Springer, 2006.

[111] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval*. McGraw Hill, 1983.

[112] G. Salton, A. Wong, and C. Yang. A vector space model for automatic indexing. *Communications of ACM*, 18(11):613–620, 1975.

[113] Satu Elisa Schaeffer. Graph clustering. *Computer Science Review*, I:27–64, 2007.

[114] W. Shang, H. Huang, H. ZHU, Y. Lin, Y. Qu, and Z. Wang. A novel feature selection algorithm for text categorization. *Expert System with Applications*, 33(1):1–5, 2007.

[115] R. Sibson. Slink: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[116] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *Proceedings of the Text Mining Workshop, ACM International Conference on Knowledge Discovery and Data Mining (KDD'00)*, 2000.

[117] C. J. Stone. An asymptotically optimal window selection rule in kernel density estimates. *Annals of Statistics*, 12:1285–1297, 1984.

[118] A. Strehl and J. Ghosh. Cluster ensembles - a knowledge reuse framework for combining multiple partitions. *The Journal of Machine Learning Research*, 3:583–617, 2003.

[119] F. Tan, X. Fu, Y. Zhang, and A. G. Bourgeois. A genetic algorithm-based method for feature subset selection. *Soft Computing 12:*, 12(2):111–120, 2007.

[120] TREC, editor. *Text REtrieval conference*. http://trec.nist.gov.

[121] E. Tsivtsivadze, T. Pahikkala, and J. Boberg. Kernels for text analysis. In *Advances of Computational Intelligence in Industrial Systems*, pages 81–97. Studies in Computational Intelligence, Volume 116, Springer, 2008.

[122] A. K. Uysal and S. Gunal. A novel probabilistic feature selection method for text classification. *Knowledge-Based Systems*, 36:226–235, 2012.

[123] N. X. Vinh, J. Epps, and J. Bailey. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *The Journal of Machine Learning Research*, 11:2837–2854, 2010.

[124] J. Wang, S. Wu, H. Q. Vu, and G. Li. Text document clustering with metric learning. In *Proceedings of the 33rd International Conference on Research and Development in Information Retrieval, SIGIR'10*, pages 783–784, 2010.

[125] J. Wang, Y. Zhou, L. Li, B. Hu, and X. Hu. Improving short text clustering performance with keyword expansion. *Advances in Intelligent and Soft Computing*, 56:291–298, 2009.

[126] P. Willet. Recent trends in hierarchical document clustering: A critical review. *Information Processing and Management*, 24:577–97, 1988.

[127] I. H. Witten, E. Frank, and M. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier, third edition, 2011.

[128] W. Xu, X.Liu, and Y.Gong. Document clustering based on non-negative matrix factorization. In *Proceedings of the International Conference on Research and Development in Information Retrieval, SIGIR'03*, pages 267–273, Toronto, Canada, 2003.

[129] J. Yang, Y. Liu, X. Zhu, Z. Liu, and X. Zhang. A new feature selection based on comprehensive measurement both in inter-category and intra-category for text categorization. *Information Processing and Management*, 48(4):741–754, 2012.

[130] Y. Yang. An evaluation of statistical approaches to text categorization. *Information Retrieval, Kluwer Academic Publishers*, 1(1-2):69–90, 1999.

[131] Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the International Conference on Machine Learning, ICML'97*, pages 412–420, 1997.

[132] J. Yu, S. Mohan, D. Putthividhya, and W. K. Wong. Latent dirichlet allocation based diversified retrieval for e-commerce search. In *Proceedings of the Seventh ACM International Conference on Web Search and Data Mining*, pages 463–472, New York, NY, USA, 2014.

[133] O. Zamir and O. Etzioni. Web document clustering: A feasibility demonstration. In *Proceedings of the International ACM Conference on Research and Development in Information Retrieval, SIGIR'98*, pages 46–54, 1998.

[134] J. Zhang, L. Chen, and G. Guo. Projected-prototype based classifier for text categorization. *Knowledge-Based Systems*, 49:179–189, 2013.

[135] W. Zhang, T. Yoshida, and X. Tang. Text classification based on multi-word with support vector machine. *Knowledge-Based Systems*, 21(8):879–886, 2008.

[136] Z. Zheng, X. Wu, and R. Srihari. Feature selection for text categorization on imbalanced data. *ACM SIGKDD Explorations Newsletter - Special Issue on Learning from Imbalanced Datasets*, 6(1):80–89, 2004.

[137] Y. Zhu, L. Jing, and J. Yu. Text clustering via constrained non-negative matrix factorization. In *IEEE International Conference on Data Mining, ICDM'11*, pages 1278–1283, 2011.

List Of Related Articles Of The Author

- T Basu, C. A. Murthy and H. Chakraborty, *A Tweak on k-Nearest Neighbour Decision Rule*, **published** in proceedings of the 16^{th} International Conference on. Image Processing, Computer Vision, and Pattern Recognition (**IPCV 2012**), Las Vegas, USA, pp. 929-935, July 2012.

- T. Basu and C. A. Murthy, *Effective Text Classification by a Supervised Feature Selection Approach*, **published** in proceedings of the 12^{th} International Conference on Data Mining Workshops (**ICDMW 2012**), IEEE CS Press, pp. 918-925, Brussels, Belgium, December, 2012.

- T. Basu and C. A. Murthy, *A Feature Selection Method for Improved Document Classification*, **published** in proceedings of the 8^{th} International Conference on Advanced Data Mining and Applications (**ADMA 2012**), LNCS vol. 7713 pp. 296-305, Nanjing, China, December, 2012.

- T. Basu and C. A. Murthy, *Towards Enriching the Quality of k-Nearest Neighbour Decision Rule for Document Classification*, **published** in the International Journal of Machine Learning and Cybernetics (**IJMLC**), Springer, vol. 5(6), pp. 897-905, 2014.

- T. Basu and C. A. Murthy, *CUES: A New Approach for Document Clustering*, **published** in the Journal of Pattern Recognition Research (**JPRR**), vol. 8(1), pp. 66-84, 2013.

- T. Basu and C. A. Murthy, *A Similarity Assessment Technique for Effective Grouping of Documents*, in **Information Sciences**, Elsevier, vol. 311, pp. 149-162, 2015.

- T. Basu and C. A. Murthy, *A Similarity based Supervised Decision Rule for Qualitative Improvement of Text Categorization*, **under revision** in **Fundamenta Informaticae**, IOS Press.

- T. Basu and C. A. Murthy, *A Supervised Feature Selection Technique for Effective Text Categorization*, **communicated to** International Journal of Machine Learning and Cybernetics (**IJMLC**), Springer.