Investigating Usage of Term Proximity and Text Segmentation to Improve Text Document Clustering

Thesis submitted in partial fulfillment of the requirements for the degree of

MASTERS OF SCIENCE BY RESEARCH

in

COMPUTER SCIENCE

by

SHASHANK PALIWAL

200702043

shashank.paliwal@research.iiit.ac.in

CENTER FOR DATA ENGINEERING

International Institute of Information Technology

Hyderabad - 500 032, INDIA

JULY 2014
International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled “Investigating Usage of Term Proximity and Text Segmentation to Improve Text Document Clustering” by Shashank Paliwal, has been carried out under my supervision and is not submitted elsewhere for a degree.

Date

Adviser: Dr. Vikram Pudi
Dedicated to my parents Mr. Anil Kumar Paliwal and Mrs. Kalpna Paliwal & my elder brother Priyank Paliwal
Acknowledgments

I would like to take this opportunity to acknowledge and appreciate the efforts of the people who have helped me during my research and documenting this thesis. I am very grateful to my adviser Dr. Vikram Pudi without whom this thesis could have never been possible. I am indebted to the generous help he provided and his insightful comments about my work. I would especially like to thank him for the freedom he provides to his students and letting them follow their interests and at the same time guiding them in the right direction. I would also like to thank Dr. P. Krishna Reddy for motivating me towards research and helping me in the initial phases of my research work. Both of them are an immense source of inspiration to me.

I would like to offer my gratitude to my lab mates Aditya, Akshat, Mohak and Shubhankar for the fruitful discussions and constant motivation.

On a personal front, I would like to thank my family members who have been a source of motivation and strength for me. Finally, I would like to thank the almighty who instilled me with the required perseverance.
Abstract

Measuring inter-document similarity is one of the most essential steps in text document clustering. Most of the document clustering methods rely on the Bag-of-Words (BOW) model for representing the text documents. The BOW model assumes that the terms of a text document are independent of each other. We attempt to explain and address two of the major drawbacks of such single term analysis of the text.

First, it completely ignores the underlying (semantic) structure of a document. In the literature, sufficient efforts have been made to enrich BOW representation using phrases and n-grams like bi-grams and tri-grams. These approaches take into account dependencies between adjacent terms or a continuous sequence of terms only. However, while some of the dependencies exist between adjacent words, others are more distant. We make an effort to enrich traditional document vector by adding the notion of term-pair features. A Term-Pair feature is a pair of two terms of the same document such that they may be adjacent to each other or distant. We investigate the process of term-pair selection and propose a methodology to select potential term-pairs from the given document. Utilizing term proximity between distant terms also allows some flexibility for two documents to be similar if they are about similar topics but with varied writing styles.

Second major drawback of single term analysis of text is that it treats whole document as a single semantic unit and thus, ignores other semantic units like sentences, passages etc. A document is inherently an organized structure consisting of various text segments or passages. We attempt to take advantage of this underlying subtopic structure of text documents and investigate whether clustering of text documents can be improved if text segments of two documents are utilized while calculating similarity between them. We concentrate on examining effects of combining suggested inter-document similarities (based on inter-passage similarities) with traditional inter-document similarities following a simple approach for the same. Experimental results on standard data sets suggest improvement in clustering of text documents for both of the above mentioned approaches.
## Contents

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Introduction</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Text Mining</td>
<td>1</td>
</tr>
<tr>
<td>1.1.1 Text Document Clustering</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1.1 Document Representation</td>
<td>2</td>
</tr>
<tr>
<td>1.1.1.2 Similarity Measure</td>
<td>4</td>
</tr>
<tr>
<td>1.1.1.3 Clustering Algorithms</td>
<td>4</td>
</tr>
<tr>
<td>1.1.1.4 Evaluation Measures for Clustering Algorithms</td>
<td>6</td>
</tr>
<tr>
<td>1.2 Overview of Proposed Approaches</td>
<td>8</td>
</tr>
<tr>
<td>1.3 Contributions of Thesis</td>
<td>9</td>
</tr>
<tr>
<td>1.4 Organization of Thesis</td>
<td>10</td>
</tr>
<tr>
<td>2 Utilizing Term Proximity based features to improve text document clustering</td>
<td>11</td>
</tr>
<tr>
<td>2.1 Related Work</td>
<td>12</td>
</tr>
<tr>
<td>2.2 Basic Idea</td>
<td>13</td>
</tr>
<tr>
<td>2.2.1 Proposed Model</td>
<td>13</td>
</tr>
<tr>
<td>2.3 Similarity Computation between Documents</td>
<td>14</td>
</tr>
<tr>
<td>2.3.1 Term-Pair Feature Selection</td>
<td>15</td>
</tr>
<tr>
<td>2.3.1.1 Algorithm for building First Term-Pair Feature Set FTPairs</td>
<td>16</td>
</tr>
<tr>
<td>2.3.1.2 Algorithm for building Second Dynamic Term-Pair Feature Set SDTPairs</td>
<td>17</td>
</tr>
<tr>
<td>2.3.2 Enriching Original Document Vector with Term-Pair Feature</td>
<td>18</td>
</tr>
<tr>
<td>2.3.3 Similarity Computation</td>
<td>19</td>
</tr>
<tr>
<td>2.4 Experimental Results And Discussion</td>
<td>20</td>
</tr>
<tr>
<td>2.4.1 Data Sets</td>
<td>21</td>
</tr>
<tr>
<td>2.4.2 Evaluation Measure</td>
<td>21</td>
</tr>
<tr>
<td>2.4.3 Clustering Algorithm</td>
<td>21</td>
</tr>
<tr>
<td>2.4.4 Baseline Approach</td>
<td>21</td>
</tr>
<tr>
<td>2.4.5 Experiment 1</td>
<td>22</td>
</tr>
<tr>
<td>2.4.6 Experiment 2</td>
<td>22</td>
</tr>
<tr>
<td>2.5 Summary</td>
<td>23</td>
</tr>
<tr>
<td>3 Applying Text Segmentation and Inter-passage similarities to improve text document clustering</td>
<td>26</td>
</tr>
<tr>
<td>3.1 Related Work</td>
<td>27</td>
</tr>
<tr>
<td>3.2 Basic Idea</td>
<td>28</td>
</tr>
<tr>
<td>3.2.1 Text Segmentation</td>
<td>28</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------</td>
</tr>
<tr>
<td>3.3</td>
<td>Similarity Computation</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Inter Document Similarity</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Inter Passage Similarity</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Combined Similarity Measure</td>
</tr>
<tr>
<td>3.4</td>
<td>Experimental Results</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Data Sets</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Evaluation Measure</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Clustering Algorithm</td>
</tr>
<tr>
<td>3.4.4</td>
<td>Baseline Approach</td>
</tr>
<tr>
<td>3.4.5</td>
<td>Results</td>
</tr>
<tr>
<td>3.4.5.1</td>
<td>Graphs for selected values of parameters Window Size and similarity blend factor $\alpha$</td>
</tr>
<tr>
<td>3.5</td>
<td>Summary</td>
</tr>
<tr>
<td>4</td>
<td>Conclusions</td>
</tr>
<tr>
<td>4.1</td>
<td>Future Work</td>
</tr>
<tr>
<td>Bibliography</td>
<td></td>
</tr>
</tbody>
</table>
### List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Representation of a document in vector space model</td>
<td>3</td>
</tr>
<tr>
<td>1.2</td>
<td>Dendogram for hierarchical agglomerative clustering of three documents A, B and C</td>
<td>5</td>
</tr>
<tr>
<td>2.1</td>
<td>Variation in F-Scores of UW-Can with $\alpha$ for $tpw_1(t_i,t_j)$</td>
<td>23</td>
</tr>
<tr>
<td>2.2</td>
<td>Variation in F-Scores of UW-Can with $\alpha$ for $tpw_2(t_i,t_j)$</td>
<td>24</td>
</tr>
<tr>
<td>2.3</td>
<td>Variation in F-Scores of UW-Can with $\alpha$ for $tpw_3(t_i,t_j)$</td>
<td>24</td>
</tr>
<tr>
<td>2.4</td>
<td>Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for $tpw_1(t_i,t_j)$</td>
<td>25</td>
</tr>
<tr>
<td>2.5</td>
<td>Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for $tpw_2(t_i,t_j)$</td>
<td>25</td>
</tr>
<tr>
<td>2.6</td>
<td>Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for $tpw_3(t_i,t_j)$</td>
<td>25</td>
</tr>
<tr>
<td>3.1</td>
<td>Varying F-Scores of UW-Can for different values of Window Size with $\alpha = 0.45$ for non-overlapping text segments</td>
<td>33</td>
</tr>
<tr>
<td>3.2</td>
<td>Varying F-Scores of UW-Can for different values of $\alpha$ with Window Size = 225 for non-overlapping text segments</td>
<td>33</td>
</tr>
<tr>
<td>3.3</td>
<td>Varying F-Scores of UW-Can for different values of Window Size with $\alpha = 0.45$ for overlapping text segments</td>
<td>34</td>
</tr>
<tr>
<td>3.4</td>
<td>Varying F-Scores of UW-Can for different values of $\alpha$ with Window Size = 425 for overlapping text segments</td>
<td>34</td>
</tr>
<tr>
<td>3.5</td>
<td>Varying F-Scores of Mini 20-newsgroups for different values of Window Size with $\alpha = 0.45$ for non-overlapping text segments</td>
<td>35</td>
</tr>
<tr>
<td>3.6</td>
<td>Varying F-Scores of Mini 20-newsgroups for different values of $\alpha$ with Window Size = 150 for non-overlapping text segments</td>
<td>35</td>
</tr>
<tr>
<td>3.7</td>
<td>Varying F-Scores of Mini 20-newsgroups for different values of Window Size with $\alpha = 0.45$ for overlapping text segments</td>
<td>36</td>
</tr>
<tr>
<td>3.8</td>
<td>Varying F-Scores of Mini 20-newsgroups for different values of $\alpha$ with Window Size = 225 for overlapping text segments</td>
<td>36</td>
</tr>
</tbody>
</table>
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>Notations</td>
<td>15</td>
</tr>
<tr>
<td>2.2</td>
<td>Words present in Document 1 and 2 and their corresponding indices in respective documents after <strong>stop-words</strong> removal</td>
<td>18</td>
</tr>
<tr>
<td>2.3</td>
<td>Feature Weighting Schema. $tf_{t,d}$ is frequency of term $t$ in document $d$, $N$ is number of documents in corpus and $x_t$ is number of documents in which term $t$ occurs.</td>
<td>19</td>
</tr>
<tr>
<td>2.4</td>
<td>Data sets description</td>
<td>21</td>
</tr>
<tr>
<td>2.5</td>
<td>showing baseline value of F-score with traditional vector based approach, for both the data sets</td>
<td>22</td>
</tr>
<tr>
<td>2.6</td>
<td>Obtained F-score and percentage improvement over baseline approach with different term pair weights.</td>
<td>22</td>
</tr>
<tr>
<td>3.1</td>
<td>showing baseline value of F-score with traditional vector based approach, for both the data sets.</td>
<td>31</td>
</tr>
<tr>
<td>3.2</td>
<td>showing maximum improvement in terms of F-score over baseline approach with values of parameters like Window Size and Similarity Blend Factor.</td>
<td>32</td>
</tr>
<tr>
<td>3.3</td>
<td>showing average improvement in terms of F-score over baseline approach with values of parameters like Window Size and Similarity Blend Factor.</td>
<td>32</td>
</tr>
</tbody>
</table>
List of Algorithms

1. Extracting First Term-Pair Features ........................................... 16
2. Extracting Second Dynamic Term-Pair Features ............................ 17
Chapter 1

Introduction

The world contains unimaginably vast amount of information which is growing and expanding at an astonishing rate. With the large explosion in the world of web, a very large share of this information is present in digital form. The growth rate of the information hosted by the world wide web is astounding. In such a scenario, it becomes more and more important to devise effective tools to maintain and utilize this ever growing data efficiently. This need gave birth to the field of “Data Mining”.

Data mining can be defined as the nontrivial extraction of implicit, previously unknown, and potentially useful information from data. The goal of the process of knowledge discovery or data mining is to extract useful information from data by analyzing data from different perspectives and effectively summarize this information for further use. This data could be present in any form like text, numbers, tables to name of a few. There are various different branches of data mining which differ from each other, primarily, in view of the different kinds of data each of these branches focus upon. We provide a detailed description to text mining, which is one of the branches of data mining and focuses on textual data, in the next section.

1.1 Text Mining

Text mining is the automatic extraction of new, previously unknown information, for a particular purpose, from a usually large amount of unstructured text. Text mining deals with natural language text, which is the form in which most of the data is present on web. Various tasks come under the field of text mining such as text summarization, text categorization, document clustering, keyword extraction, document retrieval etc. In subsequent section, we discuss text document clustering in a detailed manner [45].
1.1.1 Text Document Clustering

Text clustering is one of the most important techniques of text mining. Given a large corpus of documents, document clustering aims to produce grouping or clusters of documents such that documents in same cluster are closely related to each other (i.e. ideally, they are all related to a common topic), while those documents which are are not related to each other should be present in different clusters. In contrast to text categorization, text document clustering is an unsupervised process. No external information is utilized to guide the clustering. On the other hand, text categorization is a supervised process and involves assignment of text documents to predefined categories.

A typical text document clustering algorithm involves following four steps:

1. Document representation.
2. Definition of a document similarity measure.
3. Applying a clustering Algorithm.
4. Evaluating the clustering results.

1.1.1.1 Document Representation

Most clustering algorithms (not limited to text document clustering algorithms) expect the data set to be represented using a set of feature vectors, while each feature vector separately represents one object of the data set. This step involves extraction of these features which together form the representative vector. There exist some methods such as principle component analysis [38], to reduce the dimension of this feature vector, ideally, without any loss of information. Selecting the correct set of features to represent a document is heavily dependent on the problem domain. The dimensionality of a feature vector affects the running time of the algorithm and hence its scalability.

“Vector Space Model” [37] is one of the most popular and widely used models for representing a text document. The most common Bag-of-Words model simply uses all terms in a document as features, and thus the dimension of the feature space is equal to the number of different terms in all of the documents (i.e. corpus). If a term is not present in the document, the terms component (weight of term for this document) of the document vector will be zero and if the term is present, the vector will bear some positive value for that feature, which will depend on the weighting scheme followed. Figure 1.1 visualizes representation of a text document in vector space model.

We, in all of our experiments, use vector space model to represent documents. There are various weighting schemes which are used to weigh these features.

Let $D$ be the whole document set and then $i^{th}$ document is represented using a vector $\vec{d}_i$. 

\[ \vec{d}_i = (w_{i,1}, w_{i,2}, w_{i,3}, \ldots, w_{i,t}) \quad (1.1) \]

where \( w_{i,j} \) is the weight of \( j^{th} \) term in \( i^{th} \) document and \( t \) is the total number of terms in corpus.

There are various weighting schemes utilized to calculate the weight of these features. In term frequency scheme, the weight of every term in the document vector it is present as a feature is the number of times it appears in that document (term frequency). In binary weighting scheme, it is either 1 or 0 depending on the fact whether that term appears in a document at all or not. The number of times it appears in a document is not considered. Probably, the most popular weighting scheme is TF-IDF scheme which combines term frequency with inverse document frequency.

\[ w_{i,j} = tf_{i,j} \cdot idf_j \quad (1.2) \]

where \( tf_{i,j} \) is the frequency of \( j^{th} \) term in \( i^{th} \) document and \( idf_j \) is defined as:

\[ idf_j = \log \left( \frac{N}{df_j} \right) \quad (1.3) \]

where \( df_j \) is the number of documents in corpus, the \( j^{th} \) term appears in.

**Figure 1.1** Representation of a document in vector space model

For creating an accurate representation of a document using a vector, it is very important to select right set of features. Not all the terms present in a document are suitable to be used as features. Therefore, a set of techniques, known as “Pre-Processing” techniques, are utilized to form the right of
features. Two of the most common and widely used pre-processing techniques after the tokenization of a text document, are stop-words removal and stemming. We apply both of the above mentioned pre-processing techniques to all the documents we use in our experiments.

1.1.1.2 Similarity Measure

One of the most important factors affecting the performance of any clustering algorithm is the similarity measure it uses. A similarity measure is required to calculate the extent of similarity between two documents or two clusters or a cluster and a document. A large number of similarity metrics have been proposed in the literature, some of the most common ones are euclidean distance, cosine similarity and jaccard measure [41]. Given two feature vectors \( \vec{x} \) and \( \vec{y} \), we need to calculate degree of similarity (or dissimilarity) \( Sim(x, y) \) between them.

1.1.1.3 Clustering Algorithms

Documents can be grouped into different clusters in a number of ways. Two of the most popular set of algorithms used to cluster text documents are hierarchical and partitional clustering algorithms. Both, hierarchical and partitional techniques result in disjoint clusters i.e. each document can belong to only one cluster. Algorithms which produce overlapping clusters i.e. a document can belong to more than one clusters [27] also exist [46]. Such algorithms are particularly useful where the documents to be clustered, contain information on multiple topics. We describe hierarchical and partitional clustering here briefly, as they have been widely used in literature because of their convenience and good performance.

1. Hierarchical Clustering

The basic idea behind hierarchical clustering [41] is to build a tree which after each level merges similar set of documents (in general data points) to form the next level of tree. Therefore, progressing in this fashion we get a tree which has a cluster containing all the documents at top and singleton clusters of each document at bottom. This tree is generally visualised using a structure, called “Dendogram”. This hierarchy can be built in both bottom up (known as Agglomerative Hierarchical clustering) and top down (known as Divisive Hierarchical clustering) fashion. Agglomerative hierarchical clustering is more commonly used as compared to divisive approach which has been rarely applied.

Agglomerative Hierarchical Clustering: Agglomerative clustering starts with each of the document being treated as an individual cluster. In the following steps, it merges two clusters (at every step)
which are most similar. The steps are repeated until, either a desirable number of clusters is reached or a single cluster consisting of all the documents is obtained. Different metrics can be defined for measuring inter-cluster similarities. Three such metrics are single linkage, complete linkage and average linkage.

We use Hierarchical agglomerative clustering with complete linkage to cluster documents for all our experiments as it tends to produce tight clusters with small diameter.

**Divisive Hierarchical Clustering:** Divisive clustering follows a top down approach i.e. it starts with a one large cluster comprising of all the documents that are to be clustered. Then it divides this cluster into two smaller clusters, and this process is continued recursively until, either a desired number of clusters is reached or the number of clusters becomes equal to the number of documents i.e each cluster has only one document and no cluster can be divided further. Therefore, at any intermediate level, a divisive clustering algorithm has to select a cluster which is to be divided into two sub-clusters and then divide all the data points present inside original cluster among the two sub-clusters such that similarity

---

1. In case of text document clustering, data points which are to be clustered are text documents.
between them is minimum.

2. Partitional Clustering

Partitional Clustering algorithms [41] are not hierarchical but iterative in nature i.e. they normally require the user to give a desired number of clusters, say $k$, as input and then in first iteration, they divide the data points into these $k$ clusters and then depending on some clustering criterion, relocate the data points among these clusters until a locally optimal clustering is attained. Perhaps, the most popular partitional clustering algorithm is “k-means” algorithm.

1.1.4 Evaluation Measures for Clustering Algorithms

Measures used to evaluate clusters generated as output of clustering algorithms can be divided into following two classes [41]:

1. Internal Measures

These measures evaluate the clustering on the basis of the data that was clustered and are related to the inter/intra cluster similarity. Intra cluster similarity refers to a measure of similarity between objects present inside the same cluster while inter cluster similarity refers to a measure of similarity between different clusters. A clustering with higher intra cluster similarity and lower inter cluster similarity is considered better as compared to a clustering with lower intra cluster similarity and higher inter cluster similarity. The objective function for any internal measure is one which maximizes intra cluster similarity and minimizes inter cluster similarity.

2. External Measures

These measures judge a given set of clusters by comparing them to the predetermined benchmark or “true” classes. These benchmarks or true classes are often created using human judgement. Therefore, external measures can only be applied to evaluate clustering of a labelled data. Three of the most popular external measures are described below:

• (A) Purity Purity is a simple evaluation measure which is computed by assigning each cluster to the class which is most frequent in the cluster and then, adding the number of correctly assigned objects for all the clusters and dividing by the total number of objects in the data set.

$$Purity(\Omega, C) = \frac{1}{N} \sum_{i=1}^{K} \max_{j=1}^{I} |w_i \cap c_j|$$

(1.4)
where

- $\Omega = \omega_1, \omega_2, \ldots, \omega_K$ is the set of clusters obtained.
- $\omega_i$ is the $i^{th}$ cluster.
- $C = c_1, c_2, \ldots, c_J$ is the set of predetermined classes.
- $c_j$ is the $j^{th}$ class.

**• (B) Entropy** Entropy provides a measure of the homogeneity of un-nested clusters or clusters at a particular level of hierarchical clustering. Higher the homogeneity, lower is the entropy and vice versa. Entropy of a cluster containing only one document or object is zero as such a cluster is perfectly homogeneous. In other words, entropy measure how the various semantic classes are distributed within each cluster. Entropy for a cluster say $\omega_j$ belonging to a set of clusters represented by $\Omega$, is calculated using the following formula:

$$E(\omega_j) = -\sum_{i=1}^{K} p_{ij} \log(p_{ij})$$  \hspace{1cm} (1.5)

where

- $i$ varies from 1 to $K$ which is the total number of labelled classes and $p_{ij}$ is the probability of an object from cluster $\omega_j$ belonging to the $i^{th}$ class from the set of labelled classes.

Total entropy of a set of clusters $\Omega$ is given by:

$$E(\Omega) = \sum_{j=1}^{m} \left( \frac{N_j}{N} \ast E_i \right)$$  \hspace{1cm} (1.6)

where

- $m$ is the number of clusters present in $\Omega$.
- $N_j$ is the number of objects present in cluster $\omega_j$.
- $N$ is the total number of data objects.

**• (C) F-Measure** F-measure combines precision and recall by calculating their harmonic mean. Let there be a class $i$ and cluster $j$, then precision and recall of cluster $j$ with respect to class $i$ are as follows:
\[ Precision(i, j) = \frac{n_{ij}}{n_j} \quad Recall(i, j) = \frac{n_{ij}}{n_i} \quad (1.7) \]

where

- \( n_{ij} \) is the number of documents belonging to class \( i \) in cluster \( j \).
- \( n_i \) is number of documents belonging to class \( i \).
- \( n_j \) is the number of documents in cluster \( j \).

Then F-score of class \( i \) is the maximum F-score it has in any of the clusters:

\[ F_{score}(i) = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (1.8) \]

The overall F-score for clustering is the weighted average of F-score for each class \( i \):

\[ F_{overall} = \frac{\sum_i (n_i \cdot F(i))}{\sum_i n_i} \quad (1.9) \]

where \( n_i \) is the number of documents belonging to class \( i \).

We use F-measure score to evaluate the quality of clustering for all of our experiments.

### 1.2 Overview of Proposed Approaches

Two of the major drawbacks of following the Bag-of-Words model (traditional vector space model) are as follows:

1. Bag-of-Words model assumes that terms in a document are independent of each other and the dependencies between the set of terms present in a document are completely ignored. Term-Term proximity helps in identifying the context of the document. This underlying semantic structure present in a text document is completely ignored by this model.

2. Bag-of-Words model also assumes whole document as one single passage where as most of the authors divide text documents into more than one semantic units like sentences, passages etc. A text document is inherently an organized structure and consists of many sentences and passages. This
information is not captured and whole of the document is considered as one complete passage or a single semantic unit.

We propose two approaches to improve the performance of text document clustering which aim to reduce the effects of the above mentioned drawbacks of Bag-of-Words model to some extent. The first approach introduces a new set of features called “Term-Pair” features which are weighed according to a measure of term proximity between a pair of terms. The purpose of adding these new set of features is to capture the dependency between a pair of terms (can be adjacent or distant) rather than treating them as two unrelated entities. The second approach attempts to utilize the inherent subtopic structure present in text documents. We suggest an approach to measure similarity between two documents on the basis of similarity between their respective text segments or passages. We combine this similarity measure with the traditional tf-idf based inter-document similarity measure. For both the approaches, vector based representation is the basic representation technique as our aim is not to present an alternative clustering algorithm, but to investigate whether clustering of text documents represented using vector space model can be improved or not using term proximity and text segmentation. We also conduct several experiments to validate our proposed approaches.

1.3 Contributions of Thesis

We briefly highlight the major contributions of our thesis which are as follows:

1. We highlight the two major drawbacks of representing text documents using Bag-of-Words model and suggest innovative approaches to lower their detrimental affect on the quality of text document clustering.

2. We introduce the notion of term proximity from the view point of text document clustering. To measure term proximity between a pair of terms and to incorporate this measure into the feature vector representing the document, we develop a new set of features called “Term-Pair” features, which help in creating a better representation of the document and thus, as a result we get a more accurate measure of similarity or dissimilarity between text documents.

3. We also propose an efficient approach to generate a set of filtered term-pair features, out of the large number of term-pair features possible, as it would be highly inefficient to select all the possible pair of terms and form a feature comprising them. We, then, add these features to the traditional document vector.
4. We also discuss the utilization of text segmentation to improve the text document clustering and suggest an intuitive algorithm which first, divides documents into text windows or segments, and then incorporates these inter-segment similarities into the final inter-document similarity measure.

5. We perform experiments for all the suggested approaches on standard data sets and analyse the results to conclude whether the clustering improves or not.

1.4 Organization of Thesis

In this chapter, we have covered introduction to the contributions of this thesis. Rest of the thesis is organized in the following manner:

In Chapter 2, we look at the usage of term proximity based features to help improve the text document clustering.

In Chapter 3, we discuss the possibility of usefulness of text segmentation in improving the text document clustering and also, present an algorithm for the same.

In Chapter 4, we summarize the work presented and also suggest some ideas for the future work.

We also provide the related publications and references in their respective sections.
Chapter 2

Utilizing Term Proximity based features to improve text document clustering

Text documents are often represented as a vector where each term is associated with a weight. The Vector Space Model [37] is a popular method that abstracts each document as a vector with weighted terms acting as features. Most of the term extraction algorithms follow “Bag of Words” (BOW) representation to identify document terms. For the sake of simplicity, the BOW model assumes that words are independent of each other but this assumption does not hold true for textual data. Single term analysis is not sufficient to successfully capture the underlying (semantic) structure of a text document as it completely ignores the semantic association between terms present in a document. Proximity between terms is a very useful information which if utilized, helps to go beyond the Bag of Words representation.

Most of the work which has been done in the direction of capturing term dependencies in a document is through finding matching phrases between the documents. According to [47], a “phrase” is an ordered sequence of one or more words. Phrases are less sensitive to noise when it comes to calculating document similarity as the probability of finding matching phrases in non related documents is low [18]. But as phrase is an ‘ordered sequence’, it is not flexible enough to take different writing styles into account. Two documents may be on same topic but due to varied writing styles there may be very few matching phrases or none in worst case scenario. In such cases, phrase based approaches might not work or at best be as good as a single term analysis algorithm.

Measuring term dependency through phrases or n-grams includes dependency only between adjacent terms. However, genuine term dependencies do not exist only between adjacent words. They may also occur between more distant words such as, for example, between “powerful” and “computers” in a sample sentence like “powerful multiprocessor computers”. This work is targeted in the direction of capturing term dependencies between adjacent as well as distant terms. Proximity could be viewed as indirect measure of dependence between terms. [3] shows that term dependencies between terms are
strongly influenced by proximity between them. The intuition behind the work presented in this chapter is that if two words have some proximity between each other in one document and a similar proximity in the other document, then a combined feature of these two words when added to the original document vector should contribute to similarity between these two documents. We also suggest a feature generation process to limit the number of pairs of words to be considered for inclusion as a feature. Cosine similarity is then used to measure similarity between the two document vectors and finally Hierarchical Agglomerative Clustering (HAC) algorithm with complete linkage is used to cluster documents.

To the best of our knowledge, no work has been done so far to utilize term proximity between distant terms for improved clustering of text documents. In the subsequent sections, we introduce a new kind of feature called Term-Pair feature. A Term-Pair feature consists of a pair of terms which might be adjacent to each other as well as distant and is weighed on the basis of a term proximity measure between the two terms. With the help of different weights, we show how clustering is improved in a simple yet effective manner. We also discuss how from the large number of such possible features, only the most important ones are selected and remaining ones are discarded.

2.1 Related Work

Many Vector Space Document based clustering models make use of single term analysis only. To further improve clustering of documents and rather than treating a document as a “Bag Of Words”, including term dependency while calculating document similarity has gained attention. Most of the work dealing with term dependency or proximity in text document clustering techniques includes usage of phrases or n-gram models. [18] introduces a new document representation model called the Document Index Graph which indexes documents on the basis of phrases rather than on single term only and then, form clusters based on the phrases shared between documents. [10] and [47] do so with the use of suffix tree. [47] introduced suffix tree clustering which first identifies base clusters and then combines these base clusters into final clusters. They define base cluster as a set of documents which share a common phrase and use suffix tree structure to form this set of base clusters. In [4] and [9], the authors talk about the usage of bi-grams to improve text classification. [4] incorporate bigrams into the document representation based on distributional clusters of unigrams. [9] makes use of both unigrams and bigrams as document feature and extract top-scored features using various feature selection methods. [34] propose a feature induction technique using which they aim to improve text categorization. They rank the bigrams using a string distance measure which is similar to the string kernel [31]. In all of the above mentioned clustering techniques, semantic association between distant terms has been ignored or
is limited to words which are adjacent or a sequence of adjacent words. [1], [2] analyze the existing approaches for calculating inter-document similarity and document clustering.

Most of the existing information retrieval models are primarily based on various term statistics. In traditional models - from classic probabilistic models [13], [16] through vector space models [37] to statistical language models [28], [33] - these term statistics have been captured directly in the ranking formula. The idea of including term dependencies between distant words (distance between term occurrences) in measurement of document relevance has been explored in some of the works by incorporating these dependency measures in various models like in vector space models [15] as well as probabilistic models [40]. In literature efforts have been made to extend the state-of-the-art probabilistic model BM25 to include term proximity in calculation of relevance of document to a query [40], [35]. [20] makes use of distance based relevance formulas to improve quality of retrieval. [48] proposes a new proximity based language model and studies the integration of term proximity information into the unigram language modeling. We aim to make use of term proximity, between adjacent and distant words, in calculation of similarity between text documents represented using vector space model.

2.2 Basic Idea

The basis of the work presented in this chapter is measuring proximity among words which are common in two documents and then, utilize this measure to improve their clustering. This in turn conveys that two documents will be considered similar if they have many words in common and these words appear in ‘similar’ proximity of each other in both the documents.

2.2.1 Proposed Model

A Term-Pair feature is a feature whose weight is a measure of proximity between the pair of terms. These terms may be distant i.e. one appears after certain number of terms from other or be adjacent to each other. Since it is unclear what is the best way to measure proximity, we use three different proximity measures and normalize these measures using a common measure, so that these measures can be used as weights for the term-pair features. All these measures are independent of other relevance factors like Term Frequency(TF) and Inverse Document Frequency(idf). We choose term as a segmentation unit i.e. we measure the distance between two term occurrences based on the number of terms between two occurrences after stop words removal. $\text{dist}(t_i, t_j)$ refers to number of terms which occur between terms $t_i$ and $t_j$ in a document after stop words removal.

Let D be a document set with N number of documents:
\[ d_n = \{ t_1, t_2, t_3, \ldots, t_m \} \]

Where \( d_n \) is the \( n^{th} \) document in corpus and \( t_i \) is \( i^{th} \) term in document \( d_n \).

In this work, we use three kinds of proximity measures to compute term-pair weights for terms \( t_i \) and \( t_j \):

1. \( \text{dist}_{\text{min}}(t_i, t_j) \)

where \( \text{dist}_{\text{min}}(t_i, t_j) \) is the minimum distance expressed in terms of number of words between terms \( t_i \) and \( t_j \) in a document \( d \). Distance of 1 corresponds to adjacent terms. The intuition behind this measure is that if two terms appear very close to each other, then it can be said that the two words are related to each other in such a way that they contribute to a similar context of the document.

2. \( \text{dist}_{\text{avg}}_{\text{min}}(t_i, t_j) \)

where \( \text{dist}_{\text{avg}}_{\text{min}}(t_i, t_j) \) is the average of the shortest distance between each occurrence of the least frequent term and the nearest occurrence of the other term. Suppose \( t_i \) occurs less frequently than \( t_j \), then \( \text{dist}_{\text{avg}}_{\text{min}}(t_i, t_j) \) is the average of minimum distance between every occurrence of \( t_i \) and nearest occurrence of \( t_j \). This measure primarily searches for a closer relationship between the two terms as compared to previous measure \( \text{dist}_{\text{min}}(t_i, t_j) \) i.e. a low value of \( \text{dist}_{\text{avg}}_{\text{min}}(t_i, t_j) \) would indicate towards the presence of a phrase.

3. \( \text{dist}_{\text{avg}}(t_i, t_j) \)

where \( \text{dist}_{\text{avg}}(t_i, t_j) \) is the difference between average positions of all the occurrences of terms \( t_i \) and \( t_j \). This measure indicates the difference between the positions where each of the term tends to occur. This measure takes into consideration all the occurrences of both the terms while calculating the value.

[14]

**Example 1:** Let in a document, \( t_1 \) occurs at \( \{2, 6, 10\} \) positions and \( t_2 \) occurs at \( \{3, 4\} \) positions. Then,

- \( \text{dist}_{\text{min}}(t_1, t_2) = (3-2) = 1.0 \)
- \( \text{dist}_{\text{avg}}_{\text{min}}(t_1, t_2) = ((3-2)+(6-4))/2 = 1.5 \)
- \( \text{dist}_{\text{avg}}(t_1, t_2) = ((2+6+10)/3) - ((3+4)/2) = 2.5 \)

### 2.3 Similarity Computation between Documents

Computing similarity between two documents consists of three steps which are as follows:

1. For every document, we form a set of highly ranked terms which are discriminative and help distinguish the concerned document from other documents.
2. We form an enriched document vector by adding term proximity based features to traditional single term document vectors.

3. We compute similarity between two documents using cosine similarity measure.

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$D$</td>
<td>Set of Documents in a data set</td>
</tr>
<tr>
<td>$d_i$</td>
<td>$i^{th}$ document of set $D$</td>
</tr>
<tr>
<td>$Dterms_i$</td>
<td>Set of terms belonging to $d_i$</td>
</tr>
<tr>
<td>$HRterms_i$</td>
<td>Set of Highly Ranked terms for $d_i$</td>
</tr>
<tr>
<td>$FTpairs_i$</td>
<td>Set of First Term-Pair features for $d_i$</td>
</tr>
<tr>
<td>$SDTpairs_{(i,j)}$</td>
<td>Set of Second dynamic Term-Pair features for $d_i$ when $d_j$ is encountered</td>
</tr>
<tr>
<td>$CTerms_{(i,j)}$</td>
<td>Set of terms which are common between $d_i$ and $d_j$</td>
</tr>
</tbody>
</table>

Table 2.1 Notations

2.3.1 Term-Pair Feature Selection

In document retrieval framework, the most obvious choices of term pairs for measuring proximity are terms present in the query. However, while calculating similarity between two documents, out of the large combinations of term pairs possible, only the most important ones should be selected. Also, unlike in document retrieval framework, not only strong dependencies are important but weak dependencies are in document clustering framework also important and can not be neglected as they too might contribute to similarity or dissimilarity between the documents as per the case.

Due to including term dependencies between distant terms, the possible number of set of term pairs are $mC_2$, if we assume a document $d_i$ to have $m$ unique words. To select only those combinations which are useful in our calculation of similarity, we first sort terms of a document on the basis of tf-idf weights and then consider only highly ranked words. If this set of highly ranked words is denoted by $HRTerms$, then for a pair of words to be considered for inclusion as a feature in document vector, at least one of the words must belong to this set $HRTerms$. 

15
If the set of words which are common in both the documents is represented by $CTerms$ then for a pair of terms $t_i$ and $t_j$ to be considered as a feature if:

1. Both $t_i$ and $t_j$ belong to $HRTerms$, then $(t_i,t_j)$ is a feature (Algorithm 1), or

2. $t_i$ and $t_j$ belong to $CTerms$ and $t_i$ or $t_j$ belongs to $HRTerms$, then $(t_i,t_j)$ is a feature (Algorithm 2).

**Algorithm 1**: Extracting First Term-Pair Features

**Input**: $Dterm_i$

**Output**: $HRTerms_i$ and $FTPairs_i$

$FTPairs_i = \{\}$

$HRTerms_i = \{\}$

**for each** term $t \in Dterm_i$

**if** $(tf-idf \ weight(t) \geq \text{Minimum \ Threshold})$

$HRTerms_i = HRTerms_i \cup \{t\}$

**end if**

**end for**

$HRTerms_i = \{t_1,t_2,t_3,\ldots,t_k\}$

**for** $(m = 1; m < k; m++)$

**for** $(n = m; n \leq k; n++)$

$FTPairs_i = FTPairs_i \cup \{<t_m,t_n>\}$

**end for**

**end for**

### 2.3.1.1 Algorithm for building First Term-Pair Feature Set $FTPairs$

Algorithm 1 first forms a set of highly ranked words for a document $d_i$ consisting of all those terms which have a tf-idf weight higher than a certain user-specified minimum threshold. Now, all the possible combination of pairs of terms from $HRTerms_i$ are then added to set $FTPairs$. If $HRTerms_i$ consists of $k$ terms then $FTPairs$ will consist of $kC_2$ Term-Pair Features. The complexity of part of algorithm generating $HRTerms_i$ is $O(n)$ and part of algorithm building $FTPairs$ is $O(n^2)$.  

16
### Algorithm for building Second Dynamic Term-Pair Feature Set SDTpairs

Algorithm 2 extracts Term-Pair features dynamically before computing the similarity between two documents $d_i$ and $d_j$. These features are primarily based on set of common words $CTerms_{(i,j)}$ between the two documents. Although all the pair of terms from this set are candidates for Term-Pair features, only those pairs are finally added as features of whose at least one term belongs to set of highly ranked terms for respective documents. It is important to note that $SDTPairs_{(i,j)}$ is different from $SDTPairs_{(j,i)}$ since set $HRterms$ is different for both the documents. The complexity of Algorithm 2 is $O(n^2)$.

<table>
<thead>
<tr>
<th>Algorithm 2: Extracting Second Dynamic Term-Pair Features</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong> $Dterms_i$, $Dterms_j$, $HRterms_i$, $HRterms_j$, $FTPairs_i$ and $FTPairs_j$</td>
</tr>
<tr>
<td><strong>Output:</strong> $SDTPairs_{(i,j)}$ and $SDTPairs_{(j,i)}$</td>
</tr>
</tbody>
</table>

- $SDTPairs_{(i,j)} = \emptyset$ \hspace{1cm} \$Set of Second dynamic Term-Pair features for $d_i$ when $d_j$ is encountered$
- $SDTPairs_{(j,i)} = \emptyset$ \hspace{1cm} \$Set of Second dynamic Term-Pair features for $d_j$ when $d_i$ is encountered$
- $CTerms_{(i,j)} = \emptyset$ \hspace{1cm} \$Set of terms which are common between $d_i$ and $d_j$
- $CTerms_{(i,j)} = Dterms_i \cap Dterms_j$
- $CTerms_{(i,j)} = \{t_1, t_2, t_3, ..., t_l\}$

for (int $m = 1; m \leq l; m + +$)
    for (int $n = 1; n \leq l; n + +$)
        if ($t_m \in HRterms_i$ || $t_n \in HRterms_i$)
            if $< t_m, t_n > \notin FTPairs_i$
                $SDTPairs_{(i,j)} = SDTPairs_{(i,j)} \cup \{< t_m, t_n >\}$
            end if
        end if
        if ($t_m \in HRterms_j$ || $t_n \in HRterms_j$)
            if $< t_m, t_n > \notin FTPairs_i$
                $SDTPairs_{(j,i)} = SDTPairs_{(j,i)} \cup \{< t_m, t_n >\}$
            end if
        end if
    end for
end for

**Example 2:**

*Document 1 = “Document clustering techniques mostly rely on single term analysis of text.”*

*Document 2 = “Traditional data mining techniques do not work well on text document clustering.”*
<table>
<thead>
<tr>
<th>Word</th>
<th>Positions in Document 1</th>
<th>Positions in Document 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>analysis</td>
<td>6</td>
<td>-</td>
</tr>
<tr>
<td>clustering</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>data</td>
<td>-</td>
<td>2</td>
</tr>
<tr>
<td>document</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>mining</td>
<td>-</td>
<td>3</td>
</tr>
<tr>
<td>single</td>
<td>4</td>
<td>-</td>
</tr>
<tr>
<td>techniques</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>term</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>text</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>traditional</td>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2.2 Words present in Document 1 and 2 and their corresponding indices in respective documents after **stop-words** removal

Considering that all of the words shown in bold belong to the set of highly ranked words for both the documents, term pairs that will be a part of document vectors as term-pair feature for Document 1 as per possible cases mentioned above are:

1. According to **Algorithm 1**: \{document,single\}, \{document,term\}, \{document,analysis\}, \{single,term\}, \{single,analysis\}, \{term,analysis\}.

2. According to **Algorithm 2**: \{document,clustering\}, \{document,techniques\}, \{document,text\}.

Only the terms which remained (see Table 2.2) after stop word removal \(^1\) are considered for inclusion as term-pair features. And \{document,clustering,techniques,text\} is the set of terms which are common in both the documents.

### 2.3.2 Enriching Original Document Vector with Term-Pair Feature

So the term pairs obtained from the above step along with their respective term proximity weights (tpw) are added as features to original document vector with unigrams or single terms and their respective tf-idf weights. These weights are shown in Table 2.3.

We use “span length” to normalize the term-pair weights. **Span Length** is the number of terms present in the document segment which covers all occurrences of common set of words between two documents. The word “span” has been used here in a similar sense as used by [43], [20].

[43] defines two kind of proximity mechanisms namely span-based approaches and distance aggregation based approaches. Span-based approaches measure proximity based on the length segment

---

\(^1\)Treating \{ mostly, rely, on, of, do, not, work, well, on \} as stop-words.
Table 2.3 Feature Weighting Schema. $tf_{t,d}$ is frequency of term $t$ in document $d$, $N$ is number of documents in corpus and $x_t$ is number of documents in which term $t$ occurs.

covering all the query terms and distance aggregation based approaches measure proximity by aggregating pair-wise distances between query terms. Span-based approaches do not consider the internal structure of terms and calculate proximity only on the basis of text spans. According to their definitions, $dist_{min}(t_i, t_j)$, $dist_{avg,min}(t_i, t_j)$, $dist_{avg}(t_i, t_j)$ belong to class of distance aggregation while span length is itself a proximity measure belonging to class of span-based approaches. However, we treat span length simply as a normalization factor and not as a proximity measure. The reason behind this treatment of span-length as a simple normalization factor and not a proximity measure is the basic difference between query-document similarity and document-document similarity. When all the query terms appear in a small span of text, it is reasonable to assume that such a document is more relevant to query. However, query terms are generally more closely related as compared to words common between two documents. It would not be reasonable, in our opinion to assume that when a set of common words between two documents appear in a small text span then this contributes to similarity between two documents. There might be very few words common between two documents and thus these words can occur in a very small text span in one or both of the documents but this does not make those two documents similar. Utilizing span-length as a normalization factor caters to the above mentioned problem and helps to normalize such proximities.

In Example 2, for document 1 span-length\(^2\) is (7-1)=6 since document is the first word and text is the last word in Document 1 which are common between the two documents. Similarly, for Document 2 span-length\(^3\) is (7-4)=3.

2.3.3 Similarity Computation

We use cosine similarity to measure similarity between enriched document vectors. Cosine similarity between two document vectors $\vec{d}_1$ and $\vec{d}_2$ is calculated as

\[^2\text{Position of word “document” is 1 and position of “text” is 7 in Document 1.}\]

\[^3\text{Position of word “techniques” is 4 and position of “clustering” is 4 in Document 2.}\]
Sim(\vec{d}_1, \vec{d}_2) = \frac{\vec{d}_1 \cdot \vec{d}_2}{|\vec{d}_1||\vec{d}_2|} \tag{2.1}

where (\cdot) indicates the vector dot product and |\vec{d}| indicates the length of the vector \vec{d}. Each of the feature in document vector is weighted using tf-idf scheme.

\[ tf-idf \text{ weight} = \log(1 + tf_{(t,d)}) \times \log\left(1 + \frac{N}{x_t}\right) \tag{2.2} \]

where \(tf_{(t,d)}\) is term frequency of term \(t\) in document \(d\) and \(N\) is the total number of documents in corpus and \(x_t\) is the number of documents in which term \(t\) occurs.

2.4 Experimental Results And Discussion

We conduct experiments to investigate the effectiveness of using term-pair features in improving text document clustering. It is important to note that we do not apply any kind of dimensionality reduction on original document vector which consists of only single term features, since our aim is to investigate whether term proximity based features can be successfully utilized to improve clustering or not. In other words, we want to credit any improvement or deterioration in clustering to the suggested similarity measure.

No kind of bound has been kept on maximum number of hops between the pair of terms which combine to form a term-pair feature as the terms which are close to each other in one document but far away from each other in other documents are the ones which contribute to dissimilarity between the two documents and it is important to keep them. Experimental results also support this and unlike in relevance model, where a limit is generally kept on the distance between two terms in terms of number of words which occur between them.

To form the set \(HRTerms\) of highly ranked words, we sort terms on the basis of their tf-idf weights. These are the words which are discriminative and help to distinguish a document from other unrelated documents. If the average of tf-idf weight of all the words in a document is denoted by \(avg\), then all the words whose tf-idf weight is greater than \((\beta \times avg)\) belong to set \(HRTerms\). Here \(\beta\) is a user-defined value such that \(0 \leq \beta \leq 1\).
2.4.1 Data Sets

We used two data sets, out of which one is a web document data set\(^4\), manually collected and labeled from Canadian websites and second is a collection of articles posted on various USENET newsgroups. It is a subset of full 20-newsgroup data set. It is available from the UCI KDD archive\(^5\). Average length of a document in UW-Can data set is much greater than that of a document from mini 20-newsgroup data set. Table 2.4 describes the used data sets.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Name</th>
<th>Type</th>
<th>Number of Documents</th>
<th>Classes</th>
<th>Average number of words per document</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>UW-Can</td>
<td>HTML</td>
<td>314</td>
<td>10</td>
<td>469</td>
</tr>
<tr>
<td>2</td>
<td>Mini 20-newsgroups</td>
<td>USENET</td>
<td>2000</td>
<td>20</td>
<td>151</td>
</tr>
</tbody>
</table>

Table 2.4 Data sets description

For UW-Can data set, we kept value of $\beta$ as 0.7 and for Mini 20-newsgroups as 0.6. Lesser value of $\beta$ for Mini 20-newsgroups is in accordance with the average length of document for this data set, which is lesser than the average length of a document from UW-Can data set.

2.4.2 Evaluation Measure

We use F-measure score to evaluate the quality of the clustering. F-measure combines precision and recall by calculating their harmonic mean. A more detailed description has been provided in section 1.1.5.4 where along with F-score, various other clustering evaluation measure have been explained.

2.4.3 Clustering Algorithm

For clustering, we use Hierarchical Agglomerative Clustering with complete linkage with the help of a java based tool\(^6\).

2.4.4 Baseline Approach

We choose traditional tf-idf weighting based single term approach as our baseline approach since our aim is to investigate whether clustering can be improved by utilizing Term-Pair features or not for a set of documents represented using traditional vector space model, as suggested by us and vector space

\(^4\)Link to web data set: http://pami.uwaterloo.ca/hammouda/webdata
\(^5\)Link to mini newsgroup data set: http://kdd.ics.uci.edu/
\(^6\)Link to download tool: http://www.cs.umb.edu/smimarog/agnes/agnes.html
model based approaches are still one of the most widely used approaches. We perform two different experiments using different document vectors.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Baseline Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UW-Can</td>
<td>0.7782</td>
</tr>
<tr>
<td>Mini 20-newsgroups</td>
<td>0.35126</td>
</tr>
</tbody>
</table>

Table 2.5 showing baseline value of F-score with traditional vector based approach, for both the data sets.

### 2.4.5 Experiment 1

We add term-pair features to original document vector consisting of individual terms. As mentioned earlier, we do not apply any dimensionality reduction technique to original document vector. The F-scores using different term-pair weights are tabulated in Table 2.6.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Term Pair Weight</th>
<th>F-score</th>
<th>% improvement over baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>UW-Can</td>
<td>tpw((t_i, t_j)) = (d_{\min}(t_i, t_j)) / (S_L)</td>
<td>0.840</td>
<td>7.91</td>
</tr>
<tr>
<td>UW-Can</td>
<td>tpw((t_i, t_j)) = (d_{\min,\text{avg}}(t_i, t_j)) / (S_L)</td>
<td>0.81</td>
<td>4.08</td>
</tr>
<tr>
<td>UW-Can</td>
<td>tpw((t_i, t_j)) = (d_{\text{avg}}(t_i, t_j)) / (S_L)</td>
<td>0.799</td>
<td>2.67</td>
</tr>
<tr>
<td>Mini 20-newsgroups</td>
<td>tpw((t_i, t_j)) = (d_{\min}(t_i, t_j)) / (S_L)</td>
<td>0.384</td>
<td>9.32</td>
</tr>
<tr>
<td>Mini 20-newsgroups</td>
<td>tpw((t_i, t_j)) = (d_{\min,\text{avg}}(t_i, t_j)) / (S_L)</td>
<td>0.373</td>
<td>6.19</td>
</tr>
<tr>
<td>Mini 20-newsgroups</td>
<td>tpw((t_i, t_j)) = (d_{\text{avg}}(t_i, t_j)) / (S_L)</td>
<td>0.379</td>
<td>7.89</td>
</tr>
</tbody>
</table>

Table 2.6 Obtained F-score and percentage improvement over baseline approach with different term pair weights.

### 2.4.6 Experiment 2

To further investigate the significance of Term-Pair features, we combined term-pair features vector based cosine similarity with traditional single term feature vector based cosine similarity using weighted average of two similarities. If \(Sim_{tpf}(d_1, d_2)\) represents cosine similarity between document vectors consisting of only term-pair features and \(Sim_{tf-idf}(d_1, d_2)\) represents cosine similarity between document consisting of only traditional single term, tf-idf weighted features, then combined similarity is given by equation (2.3):

\[
Sim(d_1, d_2)) = \alpha \ast Sim_{tpf}(d_1, d_2) + (1 - \alpha) \ast Sim_{tf-idf}(d_1, d_2)
\]  

(2.3)
where $\alpha$ is the similarity bend factor and its value lies in the interval $[0,1]$.\cite{18}

We experimented with three moderate values of $\alpha$ - 0.4, 0.5, and 0.6 for each of the three Term-Pair weights. Figure 2.1, 2.2, 2.3 show the results with respective curves for UW-Can data set while 2.4, 2.5, 2.6 for mini 20-newsgroups.

For UW-Can, the obtained F-Scores however, are mostly less than the baseline score. For $tpw_1(t_i,t_j)$, F-scores improve as value of $\alpha$ is increased from 0.4 to 0.6. Best scores were obtained with $tpw_1(t_i,t_j)$ with $\alpha$ as 0.6 where the F-Score is 0.7783 marginally improving in comparison to the baseline score for UW-Can.

However, for mini 20-newsgroups better results were obtained and the F-score improved for all the term-pair weights $tpw_1(t_i,t_j)$, $tpw_2(t_i,t_j)$ and $tpw_3(t_i,t_j)$ as $\alpha$ was increased from 0.4 to 0.6. And all of the results were better than baseline score. Maximum improvement, here too, was obtained with $tpw_1(t_i,t_j)$ with $\alpha$ as 0.6 where the F-Score is 0.421 and percentage improvement of 19.85% which is much larger than that was obtained for Experiment 1. This large percentage improvement can also be attributed to the low baseline score for mini 20-newsgroups data set.

The results for both the experiments agree with experiments in relevance model where too proximity measure based on minimum pair distance generally also performs well in relevance model as reported by \cite{43} and \cite{14}.

![Figure 2.1 Variation in F-Scores of UW-Can with $\alpha$ for $tpw_1(t_i,t_j)$](image)

2.5 Summary

In this chapter, we highlight one of the major drawbacks of Bag-of-Words model that is it ignores the semantic relation between words. These semantic relationships between words present in a text document, if not ignored, can help in figuring out the context of a document more accurately and thus,
in turn, representing document more accurately. For capturing these relationships between words, we propose a new set of features called, “term-pair” features which weigh the proximity between a pair of terms. We also suggest three different weighting schemes and a normalizing factor for the same, and techniques to select candidate set of term-pair features. Experimental results show that clustering of text documents is significantly improved when term pair features are utilized as compared to tradition inter-document similarity consisting of only tf-idf weighted single term features.
Figure 2.4 Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for \( tpw_1(t_i,t_j) \)

Figure 2.5 Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for \( tpw_2(t_i,t_j) \)

Figure 2.6 Variation in F-Scores of Mini 20-newsgroups with $\alpha$ for \( tpw_3(t_i,t_j) \)
Chapter 3

Applying Text Segmentation and Inter-passage similarities to improve text document clustering

One of the most popular methods to represent text documents is using the Vector Space Model [37] which forms a vector comprising of terms present in the document as features and an associated weight for that feature or term. While such a representation is simple and easy to understand, it suffers from some problems. One of the major drawbacks is that it considers whole document as a single passage. The compressed feature vectors lose the word distribution over text segments in the documents. According to [21], different text segments represent different topics. Two documents may talk about related topics and share common vocabulary, but they could still be judged dissimilar because of other unrelated topics present in the two documents, if one or both of the documents consist of varying topics. Utilizing semantically independent units of text rather than treating whole document as one single passage, might help in reducing the defiling effect of drifting topics (as text segments on similar topics would be judged similar while those on unrelated topics as dissimilar) and varying length problem (as text segments are going to be of comparable sizes). Most of these documents do not particularly deal with a single topic, which makes it difficult to classify them under a single category. Such a scenario, thus gives rise to the need for clustering methods which can classify documents on the basis of topic on which the document is primarily written i.e. theme of most of the passages or segments which combine together to form the whole document. It is our intuition that calculating document-document similarity with the help of text segments of a particular length may help in improving quality of clustering by solving varying length problem and drifting topics problem to a small extent.

The primary aim of the work presented in this chapter is to investigate whether segmenting a document into various independent units could help in improving the clustering of text documents represented using vector space model or not. So, we present a simple algorithm to efficiently calculate inter-passage similarities between text segments of two different documents and then effectively inte-
grate these values with those obtained from considering each document as a single semantic unit, to obtain better clustering of text documents. Throughout this chapter, we use text segments or text windows interchangeably and assume them to be same i.e. a segment of document consisting of a particular number of words which we refer to as Window Size.

### 3.1 Related Work

Many vector space model based clustering approaches make use of single term analysis only. Efforts have been made to enrich this representation by utilizing the inherent structure present in a text document. A text document consists of various passages and considering whole document as one concrete unit might not be the correct representation of a document as some of the passages or segments in a text document might be deviating from the core topic of the document. A number of text segmentation methods have been proposed such as Text Tiling [22], C99 [11] and Seg-Gen [29], and most of them rely on the statistical approaches. These methods analyse the distribution of words in a text and then locate blocks of text which are thematically less cohesive.

In the field of information retrieval, problem of passage retrieval has been extensively studied. Passage retrieval is the task of retrieving only those segments of text which are relevant to a particular information need. It has been extensively utilized in the field of information retrieval to improve the quality of retrieval as even a full document which contains content related to a user’s query, might also contain content irrelevant to a user’s information need. [36][25]. Passage retrieval has also been utilized to improve performance of question answering systems [44]. [7] utilizes segmentation of web pages to improve the quality of web search. They use a vision based page segmentation technique proposed in [6][5] which takes various visual cues into account to segment a text document and then, the document is passed to a normalization procedure.

In [12], fragments of legal text documents are clustered. However, no segmentation algorithm is needed as legal documents are decomposable. [26] proposes passage-based text categorization model, which segments a document and then, passage categories are merged into document categories to achieve final categorization of documents. Perhaps, the more closely related works are [42] and [30]. In [30], authors evaluate the impact of text segmentation on query specific clustering of text documents and suggest that better results can be obtained when such a clustering is applied to the document set retrieved as the response from an IR system. [42] focuses on soft clustering of multi-topic documents using text segments by proposing an approach based on individually modeling the documents into text segment groups which are cohesive according to the document topics. Our work is different from [42]
from two aspects majorly. First, our focus is not on multi topic documents and second, we attempt to investigate effects on hard clustering, if similarity between text segments is also included in combined similarity between two documents, while [42] attempts to improve soft clustering of multi-topic documents utilizing each text segment as an independent semantic unit.

3.2 Basic Idea

The basis of this work is the intuition that two documents should be considered more similar for the purpose of clustering, if the set of common terms between the two documents are contained in a small region as compared to two other documents in which these terms are highly scattered across the documents. Traditional vector space model based techniques ignore the density of region in which these common terms fall and thus judge many similar (dissimilar) documents as dissimilar (similar).

3.2.1 Text Segmentation

Text segments can be categorized into three kinds of passages: discourse, semantic, and window. Discourse passages rely on the logical structure of the documents marked by punctuation. Semantic passages are obtained by partitioning a document into topics or sub-topics according to its semantic structure (e.g. TextTiling [23]). The third type of passages which are fixed-length passages or windows, are defined to contain a fixed number of words and were introduced in [8]. For the sake of simplicity, we use the fixed length passages in our experiments. We use both non-overlapping and overlapping passages to investigate effect of combining inter-document and inter-passage similarities on text document clustering. Example: Document = The flash washes out the photos, and the camera takes very long to turn on. Window Size = 4

1. Non-Overlapping Passages are following

<table>
<thead>
<tr>
<th>Passage 1: The flash washes out</th>
<th>Passage 2: the photos and the</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passage 3: camera takes very long</td>
<td>Passage 4: to turn on</td>
</tr>
</tbody>
</table>

2. Overlapping Passages with size of overlap = (Window size / 2) are following

<table>
<thead>
<tr>
<th>Passage 1: The flash washes out</th>
<th>Passage 2: washes out the photos</th>
<th>Passage 3: the photos and the</th>
</tr>
</thead>
<tbody>
<tr>
<td>Passage 4: and the camera takes</td>
<td>Passage 5: camera takes very long</td>
<td>Passage 6: very long to turn</td>
</tr>
<tr>
<td>Passage 7: to turn on</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.3 Similarity Computation

Let $D$ be a document set with $N$ number of documents:

$$D = d_1, d_2, d_3, ... d_N$$

Where $d_n = t_1, t_2, t_3, ... t_m$ and $d_n$ is the $n^{th}$ document in corpus and $t_i$ is $i^{th}$ term in document $d_n$.

3.3.1 Inter Document Similarity

We calculate inter document similarity by calculating cosine similarity between two document vectors with each feature weighted using tf-idf method.

$$tf-idf weight : log(1 + tf_{(t,d)}) * log \left(1 + \frac{N}{x_t}\right)$$  (3.1)

where $tf_{(t,d)}$ is term frequency of term $t$ in document $d$ and $N$ is the total number of documents in corpus and $x_t$ is the number of documents in which term $t$ occurs.

We use cosine similarity to measure similarity between enriched document vectors. Cosine similarity between two document vectors $d_1^r$ and $d_2^r$ is calculated as

$$Sim_d(d_1^r, d_2^r) = \frac{d_1^r \cdot d_2^r}{|d_1^r||d_2^r|}$$  (3.2)

3.3.2 Inter Passage Similarity

A window or passage too is represented using a feature vector with terms present in the passage being its features and tf-idf weighting scheme used to feature them. However, for weighting terms of passages, each passage is considered as a single document and all the passages of a single document, together are treated as the full corpus.

Let a document $d_1$ be divided into a set of passages or windows as $\{P_1, P_2, P_3...P_r\}$ and another document $d_2$ be divided into $\{P'_1, P'_2, P'_3...P'_s\}$ and assuming $r > s$ i.e. $d_1$ has more number of passages than $d_2$, then inter-passage similarity for $d_1$ and $d_2$ is given by equation (3.3):

$$Sim_p(d_1^r, d_2^r) = \frac{\sum_{i=1}^{r} \max(\text{Sim}(P_i, P'_j))}{r}$$  (3.3)

where $j$ varies from 1 to $s$ and $\text{Sim}(P_i, P'_j)$ is cosine similarity between feature vectors of two passages.

Therefore, inter-passage similarity would be influenced by different word distributions over text segments.
3.3.3 Combined Similarity Measure

Let inter-document similarity for two documents \(d_1\) and \(d_2\) is represented by \(Sim_d(d_1, d_2)\) and inter-passage similarity as \(Sim_p(d_1, d_2)\), then combined or effective similarity between them is given by equation (3.4):

\[
Sim(d_1, d_2)) = \alpha \times Sim_p(d_1, d_2) + (1 - \alpha) \times Sim_d(d_1, d_2)
\]  

(3.4)

where \(\alpha\) is the similarity bend factor and \(0 \leq \alpha \leq 1\). [18]

3.4 Experimental Results

We conduct experiments to investigate the effectiveness of using both inter-document and inter-passage similarities together in improving text document clustering. The experiments are conducted for two types of fixed-length passages i.e. overlapping and non-overlapping. It is important to note we do not apply any kind of dimensionality reduction on original document vector which consists of only single term features since our aim is to investigate whether inter-passage similarities can be successfully utilized to improve clustering or not. In other words, we want to credit any improvement or deterioration in clustering to the suggested similarity measure.

3.4.1 Data Sets

We use the same two data sets which we used in the experiments of Chapter 2. First one is a web document data set\(^1\), manually collected and labeled from Canadian web sites and second is a collection of articles posted on various USENET newsgroups. It is a subset of full 20-newsgroup data set. It is available from the UCI KDD archive\(^2\). While web data set has moderate overlap between different classes, mini 20-newsgroup data set has varying overlap between different classes. Average length of a document in UW-Can data set is much greater than that of a document from mini 20-newsgroup data set.

\(^1\)Link to web data set: http://pami.uwaterloo.ca/hammouda/webdata
\(^2\)Link to mini newsgroup data set: http://kdd.ics.uci.edu/
3.4.2 Evaluation Measure

We use F-measure score to evaluate the quality of the clustering. A more detailed description has been provided in section 1.1.5.4 where along with F-score, various other clustering evaluation measure have been explained.

3.4.3 Clustering Algorithm

For clustering, we use Hierarchical Agglomerative Clustering with complete linkage with the help of a java based tool\(^3\).

3.4.4 Baseline Approach

We choose traditional tf-idf weighting based single term approach as our baseline approach since our aim is to investigate whether clustering can be improved or not by combing traditional inter-document similarities with inter-passage similarities for a set of documents represented using traditional vector space model, as suggested by us and vector space model based approaches are still one of the most widely used approaches.

<table>
<thead>
<tr>
<th>Data set</th>
<th>Baseline Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>UW-Can</td>
<td>0.7782</td>
</tr>
<tr>
<td>Mini 20-newsgroups</td>
<td>0.35126</td>
</tr>
</tbody>
</table>

Table 3.1 showing baseline value of F-score with traditional vector based approach, for both the data sets.

3.4.5 Results

Results have been summarized in Table 3.3. For experiments with non-overlapping segments, we obtained maximum improvement of 7.39 % and 10.86 % in F-Score for UW-Can dataset and mini 20-newsgroups data set respectively. For experiments with over-lapping segments, we obtain maximum improvement of 10.04 % for UW-Can data set and 7.02 % for mini 20-newsgroup data set. For every experiment with overlapping segments, size of overlap is equal to half of window size.

\(^3\)Link to tool: http://www.cs.umb.edu/smimarog/agnes/agnes.html
Table 3.2 showing maximum improvement in terms of F-score over baseline approach with values of parameters like Window Size and Similarity Blend Factor.

3.4.5.1 Graphs for selected values of parameters Window Size and similarity blend factor $\alpha$

For all the experiments, similarity blend factor assumes only five values i.e. 0.4, 0.45, 0.5, 0.55 and 0.6 as we want to be moderate, so that the effectiveness of our method could be judged fairly. Similarity blend factor of 0.45 performs best for most of the experiments with both the data sets as evident from Fig 2, Fig 4, Fig 6 and Fig 8. If Fig 1 and Fig 5 are compared with Fig 3 and Fig 7, it is clear that a larger value of window size is required for better performance when dealing with overlapping windows. Window sizes used for mini 20-newsgroups are smaller as compared to those used for UW-Can which is in accordance with their average document length. Performance is reduced if larger windows are used for smaller documents.

Table 3.3 showing average improvement in terms of F-score over baseline approach with values of parameters like Window Size and Similarity Blend Factor.

3.5 Summary

In this chapter, we highlight the second major drawback of Bag-of-Words model, which is its assumption of whole text document as a one single passage, where as most of the text documents consist of multiple passages or semantic units. We propose that this inherent segmentation in text documents,
Figure 3.1 Varying F-Scores of UW-Can for different values of Window Size with $\alpha = 0.45$ for non-overlapping text segments

Figure 3.2 Varying F-Scores of UW-Can for different values of $\alpha$ with Window Size = 225 for non-overlapping text segments
Figure 3.3 Varying F-Scores of UW-Can for different values of Window Size with $\alpha = 0.45$ for overlapping text segments

Figure 3.4 Varying F-Scores of UW-Can for different values of $\alpha$ with Window Size $= 425$ for overlapping text segments
Figure 3.5 Varying F-Scores of Mini 20-newsgroups for different values of Window Size with $\alpha = 0.45$ for non-overlapping text segments

Figure 3.6 Varying F-Scores of Mini 20-newsgroups for different values of $\alpha$ with Window Size = 150 for non-overlapping text segments
Figure 3.7 Varying F-Scores of Mini 20-newsgroups for different values of Window Size with $\alpha = 0.45$ for overlapping text segments

Figure 3.8 Varying F-Scores of Mini 20-newsgroups for different values of $\alpha$ with Window Size = 225 for overlapping text segments
if utilized, can help in obtaining better clustering results. We also present a methodology for combining traditional tf-idf based inter-document similarity with newly suggested inter-passage similarity. We conduct experiments with different sizes of text segments and values of similarity blend factor. Experimental results with the combined similarity measure show significant improvement over the traditional tf-idf based similarity measure, used as baseline approach.
Chapter 4

Conclusions

This thesis involved a research on text mining, and in particular text document clustering. In the past, some effort has been expended to bring forth a document representation richer than the one based on simple Bag-of-Words model which is very popular and widely used. In this thesis, we studied and covered in detail, two major drawbacks of following Bag-of-Words model to represent and utilize text documents, from the perspective of clustering. First, it completely ignores the dependency between terms and assumes each term to be a single independent entity. Second, it treat each document as one complete passage and term distribution among different segments of a text document.

We have suggested alternative approaches which might help in eradicating these issues to some extent. In second chapter, we extended the notion of term dependency and proximity to text document clustering which, in literature, has been proposed to improve the performance of information retrieval systems. We add a new set of features called “Term-Pair” features, which are weighed on the basis of a measure of proximity between a pair of terms. These terms can either be adjacent or distant and thus, are not to be confused with bi-grams. We conducted experiments with the enriched document vectors to validate our approach. In third chapter, we propose to divide a text document into multiple text segments and then, exploit these segments while calculating inter-document similarities. We experimented with different sizes of text segments and values of similarity blend factor which influences the contribution of traditional tf-idf based similarities and similarities based on different text segments of respective documents.

It is to be kept in mind that the purpose of the work presented in this thesis, is not suggesting surrogate clustering algorithm for text documents but to determine whether or not, text document clustering of the documents represented using vector space model could be improved or not, using term-pair features and inter-passage similarities while using conventional tf-idf based approach as the basic underlying algorithm. The basic intuition behind both the present approaches is to utilize the innate structure present
in text documents to improve their clustering, in addition to the primary content of a text document. While, both the approaches might not provide best results as compared to the state of the art approaches but they do achieve a significantly better quality of clustering as compared to traditional tf-idf based approaches. And thus, we can deduce that the results are definitely promising and encouraging enough to investigate more in suggested directions. Based on the improvements obtained, it is our intuition that if such simple approaches can improve the clustering then a more elaborate and complete approach can prove to be very useful and produce much better clustering.

4.1 Future Work

The work presented in this thesis suggests new notions, rather than improving performance of existing models. Thus, both of the approaches introduced are intrinsically investigatory and there are quite a lot of possibilities which are worth exploring. Some of the future research directions, in which work presented in this thesis can be continued, are mentioned below:

1. While we have used vector space model as the basic foundation above which our approaches have been constituted, it would be interesting to develop and experiment with a model which is more pertaining to the notion of term proximity and inter-passage similarities. Such a model, if built, is more likely to result in better performance in comparison to vector space model.

2. For the experiments conducted in this thesis, we have applied traditional pre-processing techniques like stemming and stop word removal, more evolved pre-processing such as text summarization [17] and other machine learning based techniques [24] can help to create a better representation of the document and reduced dimensionality should also increase the efficiency of clustering algorithms.

3. We have formed the set of candidate terms for Term-Pair features on the basis of their respective tf-idf weights. While, tf-idf weight of a word does represent the measure of uniqueness it brings to a document, a key word extraction algorithm [19][32] can help us in selecting better candidate terms for forming term-pair features and thus, could help directly in improving the quality of clustering. Feature selection methods like Mutual Information can also be utilized to form a set of candidate terms or even the set of final term-pair features.

4. We have used a fixed length text window based approach to divide our document into various segments. Discourse and Semantic partitioning of a text document can also be utilized to divide
a document into segments and the results can be compared with the results obtained through the fixed length partitioning scheme.

5. For calculating inter-document similarities between two documents represented using feature vectors both, the similarity measure (similarity between document feature vectors or between inter-passage similarities) and the weighting scheme employed to weigh the features (term-pair features or a term belonging to particular segment of a document), affect the quality of final clustering to a large extent. Therefore, one direction in extending this work is to experiment with different similarity calculation strategies and weighting scheme. We also believe these experiments will help us understand the inherent structure of text documents better.
Related Publications


Bibliography


