

**GRAPH BASED TEXT REPRESENTATION FOR
DOCUMENT CLUSTERING**

ASMA KHAZAAL ABDULSAHIB

**MASTER OF SCIENCE (INFORMATION TECHNOLOGY)
SCHOOL OF COMPUTING
COLLEGE OF ARTS AND SCIENCES
UNIVERSITI UTARA MALAYSIA**

2015

PERMISSION TO USE

In presenting this dissertation in fulfillment of the requirements for a postgraduate degree from Universiti Utara Malaysia, I agree that the Universiti Library may make it freely available for inspection. I further agree that permission for the copying of this dissertation in any manner, in whole or in part, for scholarly purpose may be granted by my supervisor(s) or, in their absence, by the Dean of Awang Had Salleh Graduate School of Arts and Sciences. It is understood that any copying or publication or use of this dissertation or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to Universiti Utara Malaysia for any scholarly use which may be made of any material from my dissertation.

Requests for permission to copy or to make other use of materials in this dissertation, in whole or in part, should be addressed to:

Dean of Awang Had Salleh Graduate School of Arts and Sciences

UUM College of Arts and Sciences

Universiti Utara Malaysia

06010 UUM Sintok

ABSTARK

Kemajuan dalam teknologi digital dan World Wide Web telah membawa kepada peningkatan document digital yang digunakan untuk pelbagai tujuan seperti penerbitan dan Perpustakaan digital. Fenomena ini telah menimbulkan kesedaran untuk mewujudkan teknik-teknik yang lebih berkesan untuk membantu dalam pencarian dan pencapaian teks. Salah satu tugas yang paling diperlukan ialah pengkelompokkan yang boleh mengkategorikan dokumen secara automatik kepada kumpulan yang bermakna. Pengkelompokkan adalah satu tugas yang penting dalam perlombongan data dan pembelajaran mesin. Ketepatan kelompok bergantung erat pada pemilihan kaedah perwakilan teks. Kaedah tradisional memodelkan perwakilan dokumen teks dalam bentuk bag perkataan yang menggunakan teknik frekuensi istilah frekuensi dokumen indeks (TFIDF). Kaedah ini mengabaikan hubungan dan makna perkataan di dalam dokumen. Akibatnya masalah *sparsity* dan semantik yang lazim dalam dokumen teks tersebut tidak dapat diselesaikan . Dalam kajian ini , masalah *sparsity* dan semantik dikurangkan dengan mengusulkan kaedah perwakilan teks berdasarkan graf iaitu graf ketergantungan dengan tujuan untuk meningkatkan ketepatan pengkelompokkan dokumen. Skim perwakilan graf ketergantungan dihasilkan menerusi pengumpulan analisis sintaks dan semantik. Sampel daripada dataset 20 kumpulan berita telah digunakan dalam kajian ini. Dokumen-dokumen teks mengalami pra- pemprosesan dan *parsing* sintaks untuk mengenal pasti struktur ayat. Kemudian semantik perkataan dimodelkan menggunakan graf ketergantungan. Graf ketergantungan yang dihasilkan kemudian digunakan dalam proses analisis kelompok. Teknik K-means telah digunakan dalam kajian ini. Hasil kelompok berdasarkan graf ketergantungan dibandingkan dengan kaedah popular perwakilan teks iaitu TFIDF dan teks perwakilan berasaskan Ontologi. Hasil kajian menunjukkan bahawa graf ketergantungan menghasilkan keputusan baik yang melebihi kedua-dua TFIDF dan teks perwakilan berasaskan Ontologi. Ini membuktikan bahawa kaedah perwakilan teks yang dicadangkan mampu memberi hasil pengkelompokan dokumen yang lebih tepat.

ABSTRACT

Advances in digital technology and the World Wide Web has led to the increase of digital documents that are used for various purposes such as publishing and digital library. This phenomenon raises awareness for the requirement of effective techniques that can help during the search and retrieval of text. One of the most needed tasks is clustering, which categorizes documents automatically into meaningful groups. Clustering is an important task in data mining and machine learning. The accuracy of clustering depends tightly on the selection of the text representation method. Traditional methods of text representation model documents as bags of words using term-frequency index document frequency (TFIDF). This method ignores the relationship and meanings of words in the document. As a result the sparsity and semantic problem that is prevalent in textual document are not resolved. In this study, the problem of sparsity and semantic is reduced by proposing a graph based text representation method, namely dependency graph with the aim of improving the accuracy of document clustering. The dependency graph representation scheme is created through an accumulation of syntactic and semantic analysis. A sample of 20 news group, dataset was used in this study. The text documents undergo pre-processing and syntactic parsing in order to identify the sentence structure. Then the semantic of words are modeled using dependency graph. The produced dependency graph is then used in the process of cluster analysis. K-means clustering technique was used in this study. The dependency graph based clustering result were compared with the popular text representation method, i.e. TFIDF and Ontology based text representation. The result shows that the dependency graph outperforms both TFIDF and Ontology based text representation. The findings proved that the proposed text representation method leads to more accurate document clustering results.

KEYWORDS

Text Representation scheme, Dependency Graph, Document Clustering

ACKNOWLEDGEMENT

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

In the name of Allah the most gracious the most merciful First and foremost, Praise to Allah, Lord of the Worlds and prayers and peace are upon the master of messengers the Prophet Mohammed. Our leader in this life until the closing.

I would like to convey my deepest gratitude to my supervisor, Dr. SITI SAKIRA KAMARUDDIN for all continuous guidance and advices given to me in writing up of this dissertation.

Next I would like to thank University Utara Malaysia (UUM) staff. Especially, School of Computing staff for their cooperation with me.

Especial thanks to my husband Amjad Majed and my kids (Noor, Mohammed and Haidr) for their constant support and encouraged me and sacrifice during the production of this dissertation. I would like to thank my father and mother for Permanent their prayer for me to finish this work. Lastly, I want to thank my country, Iraq for the material and moral support for getting the Master Certificate.

DEDICATIONS

To the Most Merciful...

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ
الرَّحْمَنُ (1) عَلَّمَ الْقُرْآنَ (2) خَلَقَ الْإِنْسَانَ (3) عَلَّمَهُ الْبَيَانَ (4). سورة الرحمن

In the name of Allah, Most Gracious, Most Merciful.

[1] (Allah) Most Gracious! [2] It is He Who has taught the Quran.

[3] He has created man: [4] He has taught him speech (and
Intelligence). *Quran* 55:1-4.

TABLE OF CONTENTS

PERMISSION TO USE	i
ABSTRAK	ii
ABSTRACT	iii
ACKNOWLEDGEMEN	iv
DEDICATION	v
TABLE OF CONTENTS	vi
LIST OF FIGURES	x
LIST OF TABLES	xii
LIST OF APPENDICES	xiii
C H A P T E R ONE: INTRODUCTION	1
1.1 DOCUMENT CLUSTERIN	1
1.2 TEXT REPRESENTATION SCHEME	2
1.3 DOCUMENT CLUSTERING TECHNIQUES	5
1.4 PROBLEM STATEMENT	8
1.5 RESEARCH QUESTATION	9
1.6 RESEARCH OBJECTIVES	9
1.7 SCOP OF THE STUDY	10
1.8 SIGNIFICANCE OF THE STUDY	10

CHAPTER TWO: LITERATURE REVIEW	11
2.1 INTRODUCTION	11
2.2 TEXT REPRESENTATION METHOD	14
2.2.1 TF-IDF	14
A. TF-IDF ALGORITHM PRINCIPLE	16
B. TF-IDF WEIGHTING METHODS	17
2.2.2 N-GRAM	20
2.2.3 ONTOLOGY	23
2.2.4 GRAPHBASEDTEXT REPRESENTATION	26
A. CONCEPTUAL GRAPHS	27
B. FORMALCONCEPT ANALYSIS	28
C. CONCEPT FRAMEGRAPHS	29
D. DEPENDENCYGRAPH	30
2.3 CLUSTERING METHODS	34
2.3.1 PARTITIONAL CLUSTERING	36
A. K-MEANS	38
B. SETTING UP	43
2.3.2 HIERARCHICAL CLUSTERING	46
2.3.3 DENSITY BASED CLUSTERING	47
2.3.4 GRID-BASED CLUSTERING	48
2.4 EVALUATION METHODS	51
2.5 SUMMARY	53

CHAPTER THREE: RESEARCH METHODOLOGY	54
3.1 THEORETICAL STUDY	54
3.2 RESEARCH DESIGN	54
3.2.1 DOCUMENT COLLECTION	55
3.2.2 DOCUMENT PRE-PROCESSING	56
A. SPLIT DOCUMENTS	57
B. STEMMING	57
C. TOKENIZATION	58
D. PART OF SPEECH TAGGING	59
3.2.3 TEXT REPRESENTATION SCHEME	60
A. PARSING	60
B. CONSTRUCT DEPENDENCY GRAPH	61
3.2.4 APPLY CLUSTERING ALGORITHM	62
3.2.5 EVALUATION AND RESULT ANALYSIS	62
CHAPTER FOUR: DEPENDENCYGRAPH BASED TEXT REPRESENTATION	64
4.1 INTRODUCTION	64
4.2 DOCUMENT PRE-PROCESSING	65
4.3 PARSING	70
4.4 GENERATING DEPENDENCY GRAPHS	71
4.5 ONTOLOGY BASED APPROACH FOR SEMANTIC ANALYSIS	77
4.6 SUMMARY	84

C H A P T E R FIVE: RESULTS AND DISCUSSION	85
5.1 INTRODUCTION	85
5.2 CLUSTERING RESULTS	85
5.2.1 TF-IDF	86
5.2.2 DEPENDENCY GRAPH	91
5.2.3 ONTOLOGY	95
5.3 RESULT EVALUATION	96
5.4 RESULT DISCUSSION	102
5.5 SUMMARY	103
C H A P T E R SIX: CONCLUSION AND FUTUER WORK	104
6.1 INTRODUCTION	104
6.2 RESEARCH CONTRIBUTION	104
6.3 FUTURE WORK	106
6.4 SUMMARY	107
REFERENCE	108

LIST OF FIGURES

Figure 1.1: Clustering Algorithm	6
Figure 2.1: K- means Algorithm	40
Figure 2.2 ,A: Determining the Number of Clusters	40
Figure 2.2, B: Randomly Guessing the K-means Center Location	40
Figure 2.2,C: Determining Which Center Each Data Point is Closest	41
Figure 2.2, D: Finding the Centroid of The Points Owned by a Center	41
Figure 2.2, E: Repeat Until Terminated	42
Figure 2.3: The Steps of K-means Algorithm	43
Figure 3.1: Research Design	55
Figure 3.2:The 20 Newsgroup	56
Figure 3.3: Dependency Graph Generated for Beijing is a big city. The city is very beautiful	62
Figure 4.1: Steps Building The Dependency Graph	64
Figure 4.2: Single Document Called alt.atheism	65
Figure 4.3: Split Single Document in Python	66
Figure 4.4: The Example Document After Splitting into Sentences	66
Figure 4.5: Porter Stemming Algorithm	67
Figure 4.6: Stemming Process	67
Figure 4.7: Tokenization Code in Python	68
Figure 4.8: Example Document After Tokenization Process	68

Figure 4.9: Pos-Tagging Code in Python	69
Figure 4.10: Example Document After Pos-Tagging	69
Figure 4.11: Example Document After Parsing	71
Figure 4.12: Dependency Graph for The Example Document	72
Figure 4.13: The Reduction in Document Size	73
Figure 4.14: Dependency Graph Structure	74
Figure 4.15: Three Sentences Connect to Each Other in Dependency Graph	76
Figure 4.16: Ontology Algorithm	78
Figure 4.17: The Simple Text in English	80
Figure 4.18: The Resulted Hierarchy After Processing The Text	80
Figure 4.19: The Example as Dependency Graph	82
Figure 4.20: Represent Text as a Tree	83
Figure 5.1: Confusion Matrix	86
Figure 5.2, A: The Process Tokenization, Stop words and Stemming	88
Figure 5.2, B: The Process of Clustering Raw Text Using tf-idf	89
Figure 5.3: Process Clustering Dependency Graph	92
Figure 5.4: Precision Score of tf-idf, Dependency Graph, and Ontology	99
Figure 5.5: Recall Score of tf-idf, Dependency Graph, and Ontology	100
Figure 5.6: F-measure Score of tf-idf, Dependency Graph, and Ontology	101
Figure 5.7: Accuracy Score of tf-idf, Dependency Graph, and Ontology	101

LIST OF TABLES

Table 2.1: Summary of Text Representation Scheme	33
Table 2.2: Summary of Clustering Methods	50
Table 4.1: Size Documents Before and After Constructing DG	81
Table 5.1: The Value of Confusion Matrix	90
Table 5.2: Average of Experimental Results of The DG	93
Table 5.3: TP, FP, FN and TN of Dependency Graph	94
Table 5.4: Precision, Recall, F-measure and Accuracy of TFIDF Representation	96
Table 5.5: Precision, Recall, F-measure and Accuracy of DG Representation	97
Table 5.6: Precision, Recall, F-measure and Accuracy of Ontology Representation	98

List of Appendices

Appendix A: Size Documents before and after Construct Dependency Graph 120

CHAPTER ONE

INTRODUCTION

1.1 DOCUMENT CLUSTERING

Document clustering is considered a vital technology in the era of internet. It's an essential technique in mining underlying structures in text document data sets. Furthermore, this is a very interesting research topic that has influenced a number of researchers and practitioners from a number of fields, including data mining, machine learning, and information retrieval due to its fundamental role in many of real-world applications (Andrews & Fox, 2007). Text clustering means finding the groups that are related to each other. These groups are collected together in an unstructured formal document. In fact, clustering becomes very famous for its ability to offer an exceptional way of digesting in addition to generalize a good quantity of information. The extracting appropriate feature is considered the basis of clustering. Clustering text documents into category groups is a necessary step in the mining of abundance text data on the Web, indexed and retrieval or incorporate information systems and extract proper feature (concept) of a problem area. Text documents are often represented as high-dimensional, sparse vectors and complex semantics (Dhillon, et al., 2001& Jing, et al., 2005).

In existing clustering methods, a document is often represented as “bag of words” (in BOW model), N-grams (in suffix tree document model), or TF-IDF without considering the natural language relationships between the words (Wang et al.,2011).

This led to sparsity problem. The sparsity problem can be resolved by using an alternative method in order to improve the text representation schemes. Existing text representation methods also ignores important semantic links between words and/or word meanings and this led to semantic problem. A graph based text representation is proposed to solve these problems because previous studies demonstrated that it is able to capture relations between words and improve the performance of similarity measure between texts (Y. Wang et al., 2011). Representation of text document into the appropriate representation considers the main key of clustering.

1.2 TEXT REPRESENTATION SCHEME

Previous studies have shown the using suitable text representation algorithms is a key factor to improve the performance and thus will enhance the clustering results where Bloehdorn, (2005) has identified that the use of bag of words (BOW) representation affects the clustering results where, in most cases the results are unsatisfactory as it forgo relationships between necessary terms that do not co-occur literally. BOW representations are also referred to as term frequency inverse document frequency (TF-IDF) representation. Similarly, another opinion favors this subject where Harish et al. (2010) illustrates that BOW representation scheme has its own drawbacks, which includes high dimensionality of the representation, loss of correlation with adjacent words and loss of semantic relationship that exist among the terms in a document.

One of the most popular approaches to text representation is a graph model used to present context information in the text has been envisioned as a suitable solution in order to solve complex textual representations.

Graphs are mathematical structures which consist of nodes and edges which link nodes together. Mining patterns in graphs have become an important issue in real applications, such as sociology, biology, neuroscience, information management, bioinformatics and web mining (Qu et al., 2008).

Compared with other types of text representation scheme, Graph model is more efficient because it is characterized by its ability to capture the relation between words in text (Y. Wang et al., 2011). Researchers have used graph instead of other methods of ways the text representation.

There are several types of graphs to represent text. Where Sowa, (1984) suggests Conceptual Graph Model (CGM) which is more ability to be visualize for understanding. Most researchers are paying great attention to employing (CGM) for classification work or extraction features (Montes et al., 2000 ; Schenker, Last, Bunke, & Kandel, 2003).

Another type is Formal Concept Analysis (FCA) was discovered by Rudolf Wille in the early 80s (Wille, 1982) at the beginning of the eighties of this century, in spite of this it has grown the last ten years in the international community applications in many disciplines, for example, in information retrieval, Software Engineering, Linguistics, AI and Psychology.

FCA is considered a method for analyzing the data and the representation of knowledge and information management (Stumme, G., 2002). Dependency graph is a kind of representation scheme of the text which can be defined in linguistic terms, as a method to visualize the structure of a sentence to show how different words connect with each other using direct link called dependencies. This flexible approach to dependencies allows to model relationships between word segments, between phrases or entire words.

Dependency graph can also be defined as a directed graph representing dependencies of many objects towards each other. It is possible to derive an evaluation order or the lack of an evaluation order that respects the given dependencies from the dependency graph. Recent studies prove that this pattern has advantages i.e able to offers knowledge more efficiently than other type like CG (Qu et al., 2008) . In addition to its ability to discover causal relationships, it can improve the performance of similarity measure between texts (Y. Wang et al., 2011).

Dependence graph is a method for visualizing the structure of the sentence by demonstrating how different words correlate with each other by using the directed links called dependencies. In most variations of dependency grammar the nodes of a graph consist of words. That is, only allows links between words. When there is a correlation between the two words in the dependency graph, one is known as the head and the other is known as the dependent.

Dependency graphs are used in: automated software installation, and are computed compiler technology and the implementation of the official language, for example, medium assembly instructions are used to determine the optimal order of the instructions or instruction scheduling dependency graphs for transactions (Balmas, Francoise, 2001). As there are limits to satisfactory techniques for representing text, researchers look for substitute plans that either develop the present approaches or propose a novel integration of mining skills. This study focuses on the initial substitute and enriches a graph-based text representation of document clustering and to decrease parts of the restrictions of the TF-IDF based representation.

1.3 DOCUMENT CLUSTERING TECHNIQUES

Document clustering techniques usually rely on four modules: document representation model, similarity measure, clustering model and the clustering algorithm which generates clusters based on the document representation model (Hammouda & Kamel, 2004). Among all these modules, the document representation model is very fundamental and crucial for the clustering results. The present technology of clustering basically concentrates on the calculation of weightiness. To get more precise text clustering, some features are very necessary such as weight.

Tar and Nyunt (2011) Ensure that the selection of feature is very important for the process of clustering. The abundant of feature may vague the outcomes of clustering. Document clustering aims to automatically divide documents into groups based on similarities of their contents. Each group (or cluster) consists of documents that are similar among themselves (have high intra-cluster similarity) and dissimilar to

documents of other groups (have low inter-cluster similarity). This can be illustrated in Figure 1.1.

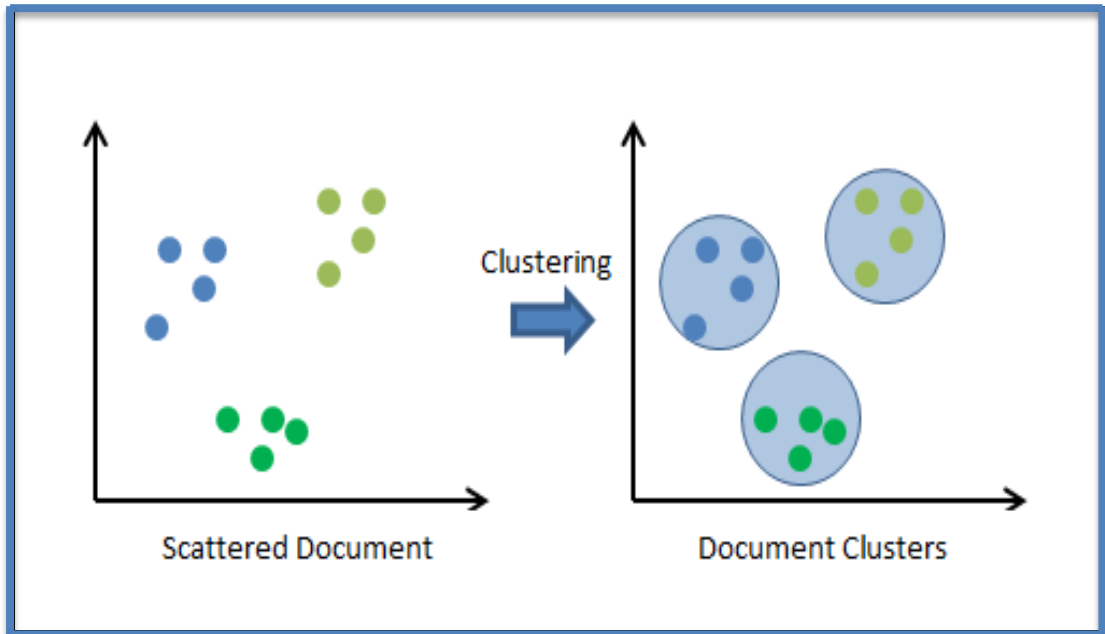


Figure 1.1: Clustering Algorithm

In this scenario, simply recognize the 3 clusters into which the information can be separated; the likeness standard is distance: two things or more will be close if they have the same feature of the cluster in regard to the distance i. e. (Geometrical distance). In fact, this is called “distance-based clustering”.

In addition to this sort, there is conceptual clustering is a sort of clustering where two items are from the same cluster. This essentially means, the method of gathering the groups based on suitability not on the similarity (Matteucci, 2008).

There are a number of clustering methods assessable in the literature. These are classified into some classes : Partitioning Clustering, Hierarchical Clustering, Density Based Clustering, and Grid Based Clustering (Tasdemir & Merényi, 2011) .
Partitional clustering: introduce this document as approaches of the non-hierarchical cluster. These approaches attempt a flat partitioning of a collection of documents into a predefined number of disjoint clusters (Kaur, 2013).

Hierarchical clustering it produces a series of partitions. Are often the similarities between each couple from documents is deposited in resemblance matrix. Density based clustering: Density based techniques are one of the highly used clustering strategies for clustering tasks in a number of different fields, including spatial, graphic, and network clustering (Huang et al. 2013; Klusch et al. 2003). The main idea behind density based clustering is to group objects as per the similarity of their neighborhood (similar objects).

Grid based clustering the mining of spatial datum targets this kind of algorithms. Their main feature is quantizing the space in a limited figure of cells, and then execution the whole processes on the quantized space (Halkidi et al. 2001).

Regardless of the clustering technique that is chosen to cluster text, previous studies have shown that the most important factor in clustering method is the text representation scheme. The selection of the text representation method has a significant impact on the accuracy of the clustering results (Hotho, Staab, & Stumme, 2003). Where studies have stated that knowledge appliance in the document representation enhances the purity values.

1.4 PROBLEM STATEMENT

In text clustering applications text are usually represented with the popular Term Frequency (TF), and Inverse Document Frequency (IDF) or vector that based on representation which treats the document and the query as vectors of term weighting . The clusters in the text group are classified and organized by some subsections of terms which make the matrix sparse. Sparsity also depends on the concept of semantics since text documents commonly include different terms to express the same message.

The sparsity problem urges the need to find an alternative method in order to perform improvement of text representation schemes. The problem related to the TF-IDF is it ignores the semantic relationship that exists among the term in a document. In this dissertation, proposed to solve the sparsity and semantic problem related to the TF-IDF using a graph based text representation with the aim of improving the text clustering results.

A graph based text representation is proposed to solve this problem because previous studies demonstrated that it is able to capture relation between words and improve the performance of similarity measure between texts (Y. Wang et al., 2011).

1.5 RESEARCH QUESTIONS

The main research question in this study relates to how the sparsity problem associated with TF-IDF representation can be reduced. In order to address this question the following sub-questions are formulated:

1. How to represent text into graph based representation to reduce the sparsity and capture the semantics of text?
2. Can a graph based text representation method produce good clustering results?

1.6 RESEARCH OBJECTIVES

This research aims to propose a graph based text representation scheme for text document clustering. There are two sub- objectives for this study:

1. To represent text documents using graph based representation to reduce the sparsity and semantic problem associated with text documents.
2. To evaluate the graph based text representation by analyzing the clustering result using precision, recall, f-measure, and accuracy.

1.7 SCOPE OF THE RESEARCH

This research focuses on graph based text representation schema, using the standard text document clustering methods and a popular variant of 20 Newsgroup dataset. This set comprises of a good number of the news stories area and classified by a good and a strong network of news over the 1987. The subjects were marked by specialists to classify these stories. The data is selected due to 20 Newsgroup is a publicly available data and are used by many researches in a text document clustering (Lan, Tan, Low & Sung, 2005).

1.8 SIGNIFICANCE OF THE STUDY

Cluster analysis is considered as significant activity in human beings life. Its analysis has been used in many forms. These forms are recognition of pattern, then the analysis of giving date, in addition to image processing, finally the study of the market. The contributions of this research lies in the method of constructing a graph based text representation which includes extracting noun phrases and chooses the most frequent phrases (top sentences) and convert them into meaningful sentences using the graph based text representation. This research hopes to improve the text clustering performance.

CHAPTER TWO

LITERATURE REVIEW

2.1 INTRODUCTION

Sparsity is a very familiar problem in the field of statistical modeling. Structural sparsity has elicited an increasing amount of attention recently. Sparsity is one of the core characteristics of real-world data. Clustering is a technique utilized to solve the sparsity problem. Text documents are usually represented as high-dimensional and sparse vectors with a few thousand dimensions, and sparsity of 95% to 99% is typical. A large number of text documents are currently on the increase (Dhillon & Modha, 2001). For instance, the World Wide Web contains nearly 1 billion pages; this number is increasing rapidly (www.alexa.com). The IBM patent server consists of more than 2 million patents (www.patents.ibm.com), and the Lexis-Nexis database contains more than 2 billion documents (www.lexisnexis.com).

Moreover, a large amount of text data exists in the private internal network of companies and technical publishing houses and in the archives of media companies. In this regard, the application of machine learning and statistical algorithms (e.g., clustering, classification, main component analysis, and differentiation analysis) to text data sets is of interest. Clustering is regarded as an impartial technique that yields optimal results. It identifies homogenous groups that have the same feature and meaning and arranges them in unstructured groups of documents. The technique is popular because it provides a large amount of information.

In the case of data in lexeme space, some of the test data may not be present in the training data. Here, the clustering process allows for data transfer from lexeme space to cluster space to reduce data sparsity. When the data in cluster space are used, it is highly unlikely to encounter such data from the test side that are not present in the training data. The reason for this is that the number of unique clusters is less than the number of unique words. Thus, transferring data from lexeme space to cluster space through clustering reduces data sparsity to a large extent (Popat, 2013).

Several researchers addressed this issue in recent years. Jing, Ng, and Huang (2007) attempted to solve the problem of data sparsity encountered in high-dimensional data clustering; they added a new step to the k-means algorithm, where the weights of all dimensions in each cluster account are calculated automatically and separately. The experiments conducted with actual and synthetic data show that the new algorithm achieves better results than other subspace clustering algorithms and can deal with large data sets.

Also Luo, Ding, and Huang (2010) have shown that sparse representation not only contributes to the simplification of data models but also helps in the discovery of predictive patterns in data. Their study helped promote the understanding and interpretation of underlying physical, biological, and other natural processes.

This work focused on determining sparse solutions to the vastly under-determined systems of linear equations. Popat (2013) employed Brown's clustering algorithm (Brown et al., 1992), the predictive exchange algorithm (Uszkoreit and Brants, 2008), and the cross-

lingual clustering algorithm (Tackstrom et al., 2012) to address the sparsity problem in different NLP applications.

The second topic addressed by many researchers is semantic relations. Semantic relation is defined by researchers as the relationship between concepts or meanings. Rosario and Hearst (2004) established five graphical models and a neural network for semantic relation classification tasks and addressed the problem of distinguishing among many various relations that exist between two semantic entities; addressing this problem is difficult but important in understanding natural language. The authors also studied the role of extracted from bioscience text.

Davidov and Rappoport (2008) showed that the semantic relationship in classification and the semantic relationship between the components of noun and nominal compounds in general are difficult to categorize rigorously.

Han, Sun, and Zhao (2011) reiterated the need to identify the semantic relationship of entities in a document that are linked with one another by exploiting the relationship among them. This task is achieved through entities that have a common link in the same document. The study showed that identifying such semantic relationship results in an improvement in entity linking accuracy. This chapter presents a review of literature on text representation and clustering methods.

2.2 TEXT REPRESENTATION METHOD

Conventional text stands for models that aim to determine if the document has keywords that appear many times. However, a document is changed to a vector in a text document ($t_1, t_2 \dots t_n$). A group of documents is introduced as a matrix, where every line denotes a phrase or word in the text of the document.

Clusters are classified by different subsections of words that make the matrix sparse, where sparsity depends on many different semantics of the clusters. In clustering sparse text data, the feature weights are utilized to find clusters from a subspace of the document vector space and identify key words that represent the semantic of clusters (Jain, 2010).

Sparse and different types of documents are necessary in designing text algorithms. This subject has been discussed in data retrieval literature. Many methods have been recommended to optimize text representation and consequently achieve the precise linking of a text with a question (Mahdabi & Crestani). Several of these techniques may be utilized to increase the representation of the document for clustering (Aggarwal & Zhai, 2012). The following subsection provides a description of several text representation schemes, such as term frequency and inverse document frequency (TF-IDF), N-gram, ontology, and graph-based text representation.

2.2.1 TF-IDF

TF-IDF is normally applied in typical text information retrieval (IR) in combination with a similar function to estimate the relevance of a set of documents to a query (Schedl, 2012). In TF-IDF, the frequency of each lexicon becomes normal through IDF. IDF reduces the weight of words that always occur in the collection. As a result, the

consequence of popular words is reduced. This condition ensures that the linking of the documents becomes more active by using words with very low frequency.

The TF-IDF weighing scheme is utilized to transform a document from a bag of words to a numeric vector to make each word in the vocabularies represent a dimension; in addition, each document is represented in this high dimensional space. Supposing that we have to select a document that is most suitable for the query “brown cow” among a collection of documents and texts in English. A simple method to begin this task is to eliminate documents that do not contain all three words, namely, “in,” “brown,” and “cow”.

However, this task still leaves numerous documents. For further recognition, the number of times that each chapter in every document occurs must be counted and then summed up altogether. The resulting sum is called the number of times (frequency) the term occurs in a document.

Nevertheless, the term “the,” being very common, would result in an emphasis on improper documents that occur frequently in the use of the word “to” without providing sufficient weight to the clear terms “brown” and “cow”.

Unlike the less common words “brown” and “cow,” the term “the” is not a good keyword to distinguish related and non-related documents and terms. Thus, the inverse document frequency factor is included to reduce the terms that occur very frequently in the document set and increase the weight of the terms that seldom occur. The disadvantage of TF-IDF is that it disregards significant semantic links between words and/or word meanings. This condition generates many problems, such as sparsity.

A. TF-IDF ALGORITHM PRINCIPLE

To establish an effective mechanism for clustering, the word that is used many times must be normal in spite of the frequency of appearance in the document. The common representation method utilized for text processing is vector space-based TF-IDF representation (Salton & McGill, 1983). The TF-IDF algorithm represents the importance of a word. However, TF and IDF are associated with word importance (Huang, X., Wu, Q., 2013; IEEE).

Naturally, if a word appears repeatedly in a document, it can be considered important and would be given a high score. If a word appears in several documents, it is not a unique identifier; hence, the word gives a low score. Common words, such as “the” and “for,” that appear in many documents are scaled down. Words that appear frequently in a single document are scaled up. TF represents the number of times a word appears in a document. IDF is a measure of the capability of a word to distinguish between categories.

The IDF of a word is obtained from the total amount of documents that contain the word divided by the total number of documents after applying the quotient logarithm. The importance of a word (T_i) in a document can be expressed as:

$$TF_{i,j} = \frac{N_{i,j}}{\sum_k N_{k,j}}. \quad (2.1)$$

Equation (1) shows the calculation of word importance (T_i) for document j . $N_{i,j}$ stands for the number of times the word is repeated in document j . $\sum_k N_{k,j}$ is the total number of occurrences of words in document j .

$$IDF_i = \log \frac{|D|}{|\{j: t_i \in d_j\}|} \quad (2.2)$$

In Equation (2), $|D|$ is the sum of the documents. The denominator is the total number of documents that contain the lexicon. The calculated inverse document frequency indicates that the number of documents that contain a word is small. Thus, IDF is small, indicating that the word has a very good discriminate capability.

$$W = TF \times IDF \quad (2.3)$$

The result of Equation (1) is multiplied by that of Equation (2) to obtain the result of Equation (3), which represents the weight of words (X. Huang & Wu, 2013).

B. TF-IDF WEIGHTING METHODS

TF-IDF is a commonly utilized method in the field of text mining. The benefit of using this approach is that it allows for the estimation of the significance of a word in a document. Word frequency in a document is basically the number of repeated words in the document. IDF involves a computation of the general significance of a term. Mathematically, the TF-IDF weight of a term is expressed as:

$$W_i = TF_i \times \log(D/DF_i). \quad (2.4)$$

In the equation above, TF_i stands for the expression frequency of term i in a document, D is the figure of documents in the body, and DF_i stands for the document frequency or number of documents that contain term i . Therefore, $\log(D/DF_i)$ stands for inverse document frequency.

When the weighting approach is used, a significant increase occurs in the repetition of a term in the specified document; however, this increase is compensated by the repetition of the term in the special corpus. The weighting approach is known to improve the precision of clustering (Li & Liu, 2012).

TF-IDF weighting can be derived from the regularity information matrix. After determining the weight vectors, they are applied to data frequency and presence. TF-IDF is a term-weighting function utilized broadly for document representation (Aggarwal & Zhai, 2012). One of the key disadvantages of this scheme is that it disregards important semantic links between words and/or word meanings and compares documents based solely on word frequencies.

Several of the current approaches that attempt to address this issue either depend on alternate representation schemes or are based on probabilistic models (Ramos, 2003; Pandit, 2008). A number of studies have attempted to represent text through the use of TF-IDF (F. Liu, Pennell, Liu, & Liu, 2009). Such studies include those conducted by Ramos (2003), Yun-tao et al. (2005), Pandit (2008), Liu, M. and Yang J. (2012), and Huang, X. and Wu, Q. (2013).

According to Ramos (2003), in studying the importance of TF-IDF, specifying the word in the corpus of documentation is perhaps more suitable when used in a question; TF-IDF can be utilized to calculate the importance of each word in the document according to the percentage of the inverse frequency of the word in a required document in proportion to the documents they appear in (in vocabularies with towering numbers). TF-IDF considers the vigorous connection of words with the specified document they appear in, indicating that if that particular lexicon appears in the question, it can be of use to the speaker. This condition provides proof that this uncomplicated algorithm professionally sorts the related lexicons that can improve question retrieval.

Yun-tao, Ling, and Yong-cheng (2005) proved that the new and reinforced TF-IDF approach, which employs confidence, support, and characteristic words, improves the recall and precision of text classification. Synonyms determined by a lexicon are processed in the improved TF-IDF approach to yield encouraging results. The new TFIDF approach improves the precision and recall of text clustering compared with conventional TF-IDF.

Pandit (2008) utilized a neural network-based learning component with TF-IDF. He restricted his study to the use of inductive methods and attempted to enrich the TF-IDF vector with additional words (in vector space) that are within the context of the document. These words are acquired by utilizing a neural network-based component trained in (document, word) pairs to determine whether the word in the pair is within the context of the document.

A number of studies have been conducted on the use of dimensionality reduction techniques, such as probabilistic latent semantic analysis (Hofmann, 2001), which seeks a k-generative model for word occurrences in a document. This method basically attempts to replace the vector-space model with a latent-space model.

M. Liu and Yang (2012) recommended an upgrade of TF-IDF weighting in the vector space model for improvement of classification accuracy by introducing a new parameter to represent the in-class characteristic; the new weighting method is called TF-IDF-CF based on TF-IDF. Traditional TF-IDF employs a term weighting scheme that is based on term ontology (Punitha, 2012). The reason for this use is that TF-IDF only pays attention to the repeated words in the document and disregards the other factors that may influence the word weights.

Huang, X. , and Wu, Q., (2013) proposed an improved TF-IDF algorithm to solve the low accuracy of micro-blog commercial word extraction and application it in term weight calculation. The possibility of applying the improved algorithm involved classifying a large amount of micro blog information into certain patterns and then assigning term weights for the classes under the Hadoop distributed framework by using the improved TF-IDF algorithm. The results indicate that the application of the improved TF IDF algorithm in micro-blog commercial word extraction is effective and enforceable.

2.2.2 N-GRAM

The N-gram model has been successfully applied in speech recognition and other applications for statistical modeling of data sequences. The model contains compact, flexible, and fast decoding algorithms. According to Galescu and Allen (2001), an N-gram model can predict words and syntactic/semantic tags simultaneously.

The N-gram method can be perceived as moving a window of size n in a word with length k . The number of N-gram that can be captured through this method is $(k-n+1)$. For example, the word “profitable” has length $k=10$; therefore, for a 5-gram word sequence, where $n=5$, we have $(10-5+1)=6$ (i.e., “profi,” “rofit,” “ofita,” “fitab,” “itable,” and “table”). If a 4-gram word is necessary, we have $(10-4+1)=7$ (i.e., “prof,” “rofi,” “ofit,” “fita,” “itab,” “tabl,” and “able.” An N-gram that is too small tends to capture more similarity but fails to capture semantic similarity. An N-gram that is too large fails to capture the similarities between similar but different words.

Moreover, an N-gram is an N-character list with a long string. The idiom comprises the idea of any repeated group of characters in the string (e.g., an N-gram composed of character number one and three of a word).

The word “TEXT” is composed of the following N-grams.

Bi-grams: _T, TE, EX, XT, T_

Tri-grams: _TE, TEX, EXT, XT_, T__

Quad-grams: _TEX, TEXT, EXT_, XT__ , T___

Generally, a string with length k padded with blanks has $k+1$ bi-grams, $k+1$ tri-grams, $k+1$ quad-grams, and so on (Cavnar & Trenkle, 1994).

Al-Ramahi and Mustafa (2012) demonstrated the application of the N-gram approach to document matching at two levels: word and entire document levels. The same can also be applied at the gateway level.

All technologies discussed in the paper applied bi-gram computation; the application of tri-gram computation may result in improving several or all of the strategies employed because tri-grams enforce further restrictions on matching in the bi-gram process. The study of Kyle, Crossley, Dai, and McNamara (2013) showed the usefulness key of the N-gram method. N-grams were divided into categories based on their syntactic, rhetorical, semantic, and grammatical features for a cohesive NLI.

With regard to the length of N-grams, they found that in spite of initially considering the words with N-1-10 grams length, N-grams longer than 5 grams were identified as a key. The longest N-gram that remained after removal based on prompt was *singed iteration* 4 grams. This result suggests that a threshold of 4 grams (or maybe even 3 grams) is possibly beneficial for future investigations.

N-grams have been proven to have practical advantages for several reasons, namely, drafting potential models for them is easy, they are easy to extract from a corpus, and they provide useful probability estimates for alternative readings of the input (Manjula K.S. et al., 2013).

In this method, a term is decomposed into fragments of size *n*. A matching algorithm is then designed to compare these fragments and identify their similarity and dissimilarity. The advantages of N-gram method are as follows: it involves uniform length, computation time is short, and the method can prevent noisy text data in terms of spelling errors (Agyemang, Barker, & Alhajj, 2005).

A disadvantage of the N-gram method is that it interferes with two words and fail to provide information on the relations between these words. For example, N-grams “ew Y” and “w Yo” from the words “New York” often exist together. Another disadvantage is that it is contrary to the system that handles the input as a single word (Náther, 2005). The other drawbacks of N-gram includes the need for a large memory to store a large number of n-gram sequences for each considered word (Agyemang, Barker, & Alhajj, 2005).

2.2.3 ONTOLOGY

Ontology is broadly interpreted as a dynamic, controlled vocabulary to describe a plurality of objects or concepts contained in a process or domain and their interrelationships. Ontology may be further thought of as a formal knowledge-representation system that comprises rules for the manner by which knowledge is represented along with rules that enable automated reasoning with regard to the objects or concepts represented (Gardner, 2007). An ontology term is a single-named concept that describes an object or entity. A concept may, for example, comprise a collection of words and associated relevance weights, co-occurrence, and word localization statistics that describe a topic.

Users face the challenge of organizing, analyzing, and searching for significant numbers of documents because of the ever increasing amounts of textual information available electronically. The automatic classification of text documents into predefined objective categories or clusters of documents with related similar content is a promising approach to address this complexity. Numerous machine learning algorithms for supervised and unsupervised text categorization have been proposed in the past decades.

However, to date, existing text classification systems normally employ the bag-of-words model of information retrieval; in the model, single words or word stems are utilized as features to represent document content (Salton, Singhal, Mitra, & Buckley, 1997). Several researchers regard this issue of principle knowledge exploitation presented in textual documents as an important issue in constructing systems for knowledge management and relevant tasks.

Various disciplines, scientific or otherwise, a number of resources (e.g., data management systems) may exist to represent cumulative knowledge gathered for different specialty areas within each discipline. Several existing systems, for instance, may use separate ontologies for each area of specialty within a particular discipline.

A drawback associated with such ontology systems, however, is the inability to link different areas of specialty within a discipline. Another drawback associated with existing ontology systems is the inability to map ontology terms to infer linkages to related or referenced entities in wider data sources. While existing ontology systems generally offer descriptions of a particular discipline syntactically (in terms of nouns or entity names), they often lack a particularly well-developed set of semantic relationships (in terms of verbs or relationship between entities) built into the ontology.

Yet another drawback associated with existing ontology systems is a general lack of names in the ontology that correspond to all the information or concepts in a document set. A number of studies have dealt with this type of text representation.

For instance, Bloedorn, Cimiano, Hotho, and Staab (2005) presented the ontology-based text mining framework (OTTO), which revolves around the KANO model, to determine the interaction between ontologies. The researchers established an approach that exploits current ontologies through the use of their lexica and concept hierarchies to improve results in both supervised and unsupervised settings; Reuters-21578 corpus using WordNet was employed as ontology.

Gardner (2007) focused on an information management system and provided a method to integrate structured and unstructured data using an ontological approach, which further comprises processes for creating, validating, augmenting, and combining ontologies for life science informatics and other disciplines.

Tar and Nyunt (2011) proposed a system that utilizes the concept of weight for text clustering. It was developed with a k-means algorithm based on the principle of ontology. The system is used to identify irrelevant or redundant features that may reduce strength, thereby achieving more accurate document clustering.

Ma et al. (2012) developed a text mining approach based on ontology to collect novel research proposals based on the similarity in the areas of research. The approach proved to be effective and efficient in compiling research proposals with English and Chinese texts. An ontology study was conducted for the classification of the concepts of disciplines in terms of the various regions and forming relations among them.

Text mining and optimized technologies were adopted to compile research proposals on the basis of similarities; eventually, balance was achieved in accordance with the characteristics of applicants. Experimental results showed that the proposed method improves the similarity of the proposal sets and promotes efficiency of assembly in the proposal process.

Punitha (2012) compared the performance of strengthening ontological algorithms based on k-means and DBSCAN clustering. Ontology was used to present the concept of weight, which was calculated by determining the correlation coefficient of the word and the

probability of the concept. Different experiments were conducted with performance evaluation. The results showed that the inclusion of ontology increased clustering efficiency; the ontology-based DBSCAN algorithm presented the best performance among the considered k-means algorithms based on ontology.

Many researchers employed graphical text representation, such as conceptual graphs, formal concept analysis, concept frame graphs, and dependency graphs, because of the lack of semantics in previous work on text representation. The following subsections present a review of several graph-based text representation schemes.

2.2.4 GRAPH-BASED TEXT REPRESENTATION

Many researchers have resorted to representing text graphically because earlier work on text representation lacks semantics. Graphical text representation includes a formal analysis of concepts, graphs related to the conceptual frame, and graphs based on ideas. For action graphs, algorithms based on graphs are dependent on graph partitioning.

In this case, clusters are identified by a graph's cutting edges in such a way that the sum of the weights of the edges that are cut (the edge cut) is minimized. Each document is represented by a node.

In the case of similarity between documents in different clusters, the edge appears between two nodes. If the documents that are contained in the cluster are highly related, the edges in the same cluster will weigh more than the edges across clusters. Each graph-based algorithm may produce the ground differently. They may also use graph partitioning algorithms differently (Kaur & Kaur, 2013).

Several researchers employed graphical text representation as a classification system. An email that is represented as a graph instead of merely a bag of words has a better likelihood of achieving high accuracy compared with other approaches.

Chakravarthy, Venkatachalam, and Telang (2010) proposed a graph-based approach that employs two domains -independent graph representations to cover text (webpages and email). The graph representations are selected on the basis of domain knowledge to provide focus on the domains.

A. Conceptual Graphs

Conceptual graphs (CGs) comprise a language for representing knowledge. They are rooted in the field of linguistics psychology and philosophy (Sowa & Way, 1986). Knowledge structure at the semantic level can be represented by CGs. CGs involve two elements: concept and relation. CGs are therefore bipartite, connected, and finite.

A set of edges and nodes for vertices are included in a graph. CGs simplify the relations of any arity to the rest of the network languages that use a labeled arc. CGs are similar to graphs used in normal natural languages. Accurate and highly structured information can be adequately represented by CGs.

Constructed CG is most frequently used for graph matching; it produces results that are reliable for various purposes. The procedure of comparing data in the text is eased by representing text with CG formalism and performing CG matching.

B. Formal Concept Analysis

Formal concept analysis (FCA) is an initial method utilized to derive a concept hierarchy or formal ontology from a set of objects and their characteristics. Each concept in the hierarchy represents a group of objects that share identical values for a certain set of characteristics. Each sub-concept in the hierarchy includes a subset of the objects in the concepts above it. This term was coined by Rudolf Wille in 1984.

The method was derived from applying lattice and order theory developed by Garrett Birkhoff et al. in the 1930s. This analytical method involves analyzing data for clarification to identify conceptual structures in the data set. Studies on the similarity measure for FCA are based on Tversky's model and rough set theory. In FCA, elements of the same type are called "formal objects," and elements of a different type are called "formal attributes".

The adjective "formal" is employed to confirm formal notions. Formal objects need not necessarily be "objects" in any type of the logical meaning of "object." However, using "object" and "attribute" provides an indication because in many applications, it may be useful to select an object-like items as formal objects and their features or characteristics as formal attributes.

Cimiano, Hotho, and Staab (2005) developed an FCA-based approach by analyzing the effect of a smoothing technique on data sparsity. The documents in an information retrieval application can be considered object-like, whereas terms can be considered an attribute-like. Other examples of sets of formal objects and their formal attributes are tokens and types, values and data types, data-driven facts and theories, words and

meanings, and so on (Priss, 2006). FCA has practical applications in the areas of data mining, text mining, knowledge management, machine learning, software development, biology, semantic web, and so on (Wang & Liu, 2008).

Qadi, Aboutajedine, and Ennouary (2010) proposed a mechanism to improve IR in a website based on FCA. The said mechanism creates semantic relations through queries and allows the reorganization of concepts in the shape of a lattice. The replies are then submitted by a search engine.

Furthermore, their group proposed an incremental algorithm for IR based on the Galois lattice. This algorithm allows the formal clustering of data sources, the results of which are classified based on relevance. The control of relevance is exploited during clustering.

C. Concept Frame Graphs

Concept frame graph (CFG) was designed by Rajarmaman and Tan (Rajaraman & Tan, 2002) for knowledge discovery from text. CFG was created using a rule-based system that has the capability to extract CFG from text, and the created CFG was mined to uncover useful knowledge.

An experiment to evaluate the performance of mining with and without graph-based text representation was performed by Rajarmaman and Tan (2003). The algorithm that employed CFG performed better than the algorithm without a graph structure; 18% improvement in precision and 35% improvement in recall were observed. This result indicates that improved mining performance can be achieved by representing text as graph structures.

D. Dependency Graph

A dependency graph is a directed graph that denotes the dependencies of several items. Obtaining an assessment order or the non-attendance of an estimation order that compliments the dependency from the graph is indeed possible (Balmas, 2004). This graph is an appropriate representation of the dependency relationship.

This graph comprises of a group of propositions (nodes), a cover job of assurance (node values), and a constant group of dependency relationships (connecting arcs) constraining the assignment of confidences. The confidences are fully determined (a single value), a specific part (several values), or a nonspecific part (all values).

Zimmermann and Nagappan (2008) defined a dependency graph as a directed graph $G = (V, E)$, where V is a set of nodes (binaries) and $E \subseteq V \times V$ is a set of edges (dependencies).

Dietrich, Yakovlev, McCartin, Jenson, and Duchrow (2008) employed the object dependency exploration model (ODEM) to extract the dependency graph. The dependency graph encoded in ODEM includes nodes that represent classes. These nodes have annotations that determine their classification (class, interface, annotation, etc.), vision, abstraction, and finality.

Each node also includes a list of one-way relations (dependencies) with the full name of the class (package Name .class Name) referred to and a dependency classification annotation (uses, extends, or implements). This technique improves visualization, and thus makes the appearance of the graph much less complicated.

Qu, Qiu, Sun, and Wang (2008) proposed a novel model called feature event dependency graph (FEDG), which is capable of presenting knowledge more efficiently than CGs.

Dietrich, Yakovlev, McCartin, Jenson, and Duchrow (2008) recommended the use of the Girvan–Newman clustering algorithm to calculate the modular structure of programs. This method is considered a new approach to the analysis of dependency graphs of object-oriented programs. This graph is important for software engineers as they redraw the component boundaries in a software to improve the level of re-use and maintenance. The use of this graph has been proven to be effective, but the visualisation of the graph in cases where large numbers of nodes exists requires further improvement.

Patel, Hamou-Lhadj, and Rilling (2009) introduced a new clustering technique that combines dynamic analysis and static dependencies; the dependency graph includes the link representing the structural relations among the classes, and the classes are the nodes of the graphs. The graph is a directed graph with at most two edges between two classes, and all edges have the same weight. The extraction of structural relationship is an automatic task supported by numerous tools. Extraction tools differ in their support of various technologies.

Mitchell and Mancoridis (2010) employed a source code analysis system for the production of a dependency graph that represents the system modules and module-level inter-relations. This graph was then used as input to the bunch tool, which partitions the graph. The results were presented in a clustering graph using graph visualization.

The experimental results obtained by Y. Wang, Ni, Sun, Tong, and Chen (2011) showed that the dependency graph algorithm exhibits the best performance in a specified document clustering among methods that are based on the BOW model. This algorithm can also identify causal relationships and improves the performance of the similarity measure between texts.

The approach proposed by Beck and Diehl (2013) involves the merging of dependency graphs before performing clustering and an applied set of operations, such as union, weighted union, and intersection on the set of edges. The researchers concluded that combining both approaches improves the overall clustering quality.

This section presents the types of schemes used in text representation and their influence on the clustering process. Researchers who employed these schemes and their studies that identified the advantages and disadvantages of each method were also presented. In the current research, the technique of the dependency graph for text representation was used to determine if this technique exhibits higher accuracy in text representation compared with other schemes. Table 2.1 displays the advantages and disadvantages of each type of text representation schemes

Table 2.1

Summary of Text Representation Scheme

Author	The proposed approach	Advantages	Disadvantages
<i>Ramos (2003); Pandit (2008)</i>	<i>TF-IDF</i>	<i>Can be applied to both frequencies of data and presence of data</i>	<i>Ignores important semantic links between words and/or word meanings.</i>
<i>Dietrich,et al.,(2008); Y. Wang et al., (2011)</i>	<i>Dependency Graph</i>	<i>Discover causal relationships and improvement the performance of similarity measure between texts</i>	<i>Required to improve the visualization</i>
<i>Agyemang et al.(2005); Náther,(2005)</i>	<i>N-gram</i>	<i>Faster computation time</i>	<i>Needs large memory allocations</i>
<i>Gardner, (2007); Thi,Thi et al., (2011)</i>	<i>Ontology</i>	<i>Helps to visualize data</i>	<i>1-Needs distance measure to compare ontology. 2-The inability to link different areas of specialty within a discipline. 3-Ignored some words during the construction of the ontology.</i>
<i>Wang &liu (2008)</i>	<i>Conceptual Graphs & Formal Concept Analysis & Concept Frame</i>	<i>Able to capture relation between words</i>	<i>Comparing graphs are computationally complex</i>

Table 2.1 shows that although TF-IDF can be applied to data frequencies and presence, it disregards important semantic links between words and/or word meanings and compares documents based solely on word frequencies. The types of graphs capture the relation between words and the dependency graph in addition to identifying causal relationships that can improve the performance of the similarity measure between texts.

2.3 CLUSTERING METHODS

A clustering method is classified as vector space if it directly processes data in a multi-dimensional feature space, where each data object is represented by a point in the space. Therefore, an ideal cluster is a group of close points that are far away from other groups. Meanwhile, a clustering method is considered graph based if it only makes use of the pairwise relationships between data objects (i.e., a graph with nodes and edges representing data objects and similarities, respectively).

The definition of cluster analysis is general because it does not precisely specify how to measure similarity as well as the quality of a clustering. In fact, one can have many different criteria for measuring and grouping similar data objects.

Thus, determining whether a clustering solution is good is difficult unless the data are easy to cluster according to a particular user's goal. Generally, different solutions may be equally good for the same data set, and different data sets may require different clustering criteria/methods to produce a good result.

Consequently, numerous clustering methods have been developed. Each method has its own assumptions defined explicitly or implicitly in relation to the characteristics of clusters in data. Clustering classifies documents into several classes based on the topics.

Therefore, each class has one topic (Tar & Nyunt, 2011; J. Z. Huang & Ng, 2006). One of the main tasks in text mining is text clustering. The process of text clustering is important in data indexing, management, and mining from text on an Internet website. Its importance comes from its capability to extract the traits of a problem. Clustering text documents into categories or groups is a necessary step in the mining of abundant text data on the Web as well as in indexing and retrieving incorporated information systems and extraction of the proper features (concept) of a problem area.

The challenges in text clustering include large volume, high dimensionality, and complex semantics (Jing, Ng, Xu, & Huang, 2005). The primary aspect of clustering algorithms involves the compactness and isolation of clusters.

These algorithms are supposed to meet such standards based on primary suppositions for a sample of the standards (e.g., initial locations of the cluster centers) or input parameter values (e.g., number of clusters, minimum diameter, or number of points in a cluster) (Halkidi & Vazirgiannis, 2008).

A large number of these algorithms are deliberated upon to discuss document clustering. A condition that should be noted is that a technique that is suitable for one text does not mean it is also suitable for another text. The language of clustering in the text is very sensitive. In the past, partitional clustering, hierarchical clustering, density-based clustering, and grid-based clustering were the preferred clustering methods (Andrews & Fox, 2007).

2.3.1 PARTITIONAL CLUSTERING

Partitional clustering involves making similar objects in the same set very large and dissimilar objects among the various groups very small. Therefore, the function of an algorithmic task can be stated as an optimization problem. A statistical standard that studies variance scatter matrices may be utilized to depict variance within and between scatter matrices as well as to quantify the fitness of the partitions and identify the best one. A clustering algorithm must be clear, simple, and efficient and can handle a large amount of data. Furthermore, it should be strong for equal samples and can discover various cluster forms (Paterlini & Krink, 2006). Partitioning algorithms are generally utilized in studies on databases for the effective creation of clusters of objects (Aggarwal & Zhai, 2012).

These algorithms are further divided into repeated approaches. A large number of partitioning algorithms are repeated. The k-means algorithm is a famous partitional clustering algorithm because of its ease of implementation resulting from its linear time complication. The primary disadvantage of this algorithm is that it spots local optima; hence, it is unable to address clusters that have a non-spherical shape feature, relying on the notion that the heart of the cluster (the centroid) is an outstanding representation of the cluster (M. Kaur, 2013).

The abbreviations of several algorithms are as follows: PAM refers to partitioning around medoids, CLARA refers to clustering large applications, and CLARAN refers to clustering large applications based on randomized search. Lastly, k-prototypes and k-mode (Huang, 1997) depend on the k-means algorithm but focus on the information of clustering decisive data (Halkidi, Batistakis, & Vazirgiannis, 2001).

Several studies have discussed the k-means algorithm. Dhillon and Modha (2001) studied vector space models of large groups of documents. The notion vectors formalize a vigorous sparse and centralize a “basis” for text data sets. These are high-dimensional and sparse and involve the use of the fast, spherical k-means clustering algorithm to generate significant clusters with perfect descriptive class.

Previous studies have shown that partitional clustering algorithms are suitable for clustering documents or datasets with minimum computational demands (Zhao & Karypis, 2002).

Saad, de la Iglesia, and Bell (2006) explained that partitional clustering algorithms are suitable for clustering large documents or data groups because of their low computational demands. However, the quality of clustering by partitional algorithms is actually marginalized and less efficacious than the agglomerative value. This outcome is attributed to the agreement between partitional and agglomerative algorithms.

Tarabalka, Benediktsson, and Chanussot (2009) reported that the approach they recommended allows for pixel-wise SVM categorization and produces a division chart through partitional clustering. This task is accomplished by excluding a large portion of voting in the pixel-wise spectral classification utilizing adaptive neighborhoods that are defined by a split map. The results showed that the recommended method improves classification precision and enhances rating charts with homogeneous areas.

In this study, the k-means algorithm was employed because it is one of the most well-known algorithms utilized by a large number of researchers. In the evaluation stage, the use of this algorithm in previous studies and in the present study was compared.

A. K-means

K-means is one of the oldest clustering algorithms. The centroid for each group is calculated to determine cluster membership. Objects are classified into one of the K groups (where K is pre-selected), and the nearest centroid of the group for each object is identified. The iterative redistribution from members of the cluster reduces the comprehensive within-cluster dispersion (Chen et al., 2002).

This algorithm is simple, straightforward, and based on the strong foundation of analysis of variance. The k-means algorithm groups a set of data vectors into a predetermined number of clusters. The algorithm begins with a random initial cluster center and continuously redistributes the data objects in the dataset to cluster centers based on the similarity between the data object and the cluster center.

The algorithm consists of two separate phases. The first phase determines the k centroids (one for each cluster). The second phase takes each point belonging to a given data set and associates it with the nearest centroid.

Euclidean distances and cosine similarity are generally considered to determine the distance between data points and centroids. The initial step is finished when all points have been included in several clusters and nearly grouped. At this point, the new centroids are recalculated because the inclusion of new points may change the cluster centroids.

Once k -new centroids are found, a new binding is created between the same data points and the nearest new centroid. Consequently, a loop is generated, and the k centroids may change their position in a step-by-step manner.

Lastly, the centroids reach a point when they no longer move. This point indicates the convergence criterion for clustering when all points have been included.

K-means is a clustering algorithm that is based on a simple and clear idea. After a set of initial clusters is selected, each point is then assigned to one of them. Finally, each cluster center is replaced by the mean point of the respective cluster. These simple steps are repeated until convergence is achieved. A point is allocated to the cluster that is close in Euclidean distance to the center.

The initial selection of the cluster centroids determines the main processing of k-means and the partition results of the dataset. The processing of k-means aims to establish the local perfect solution in the vicinity of the initial solution and improve the partition results (Cui & Potok, 2005; Kaur & Kaur, 2013). Figure 2.1 shows the steps for a working k-means algorithm.

1. Begin with a decision on the value of the k = number of clusters.
2. Put any initial partition that classifies the data into k clusters.
 - Take the first k data as single-element clusters
 - Assign each of the remaining (N-k) data to the cluster with the nearest centroid.
After each assignment, recomputed the centroid of the gaining cluster.
3. Take each data in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new sample and the cluster losing the sample. The distance metric used is cosine distance is defined as:

$$sim(C_1, C_2) = CosSim(V_1, V_2) = \frac{\sum_{i=1}^n w_{1i} w_{2i}}{\sqrt{\sum_{i=1}^n w_{1i}^2} \sqrt{\sum_{i=1}^n w_{2i}^2}} \quad 2.1$$

Where C1 and C2, their text descriptions are converted to two vectors of words as V1=(w11, w12, ..., w1n), V2=(w21, w22, ..., w2n).
4. Repeat step 3 until convergence is achieved, that is, until a pass through the training sample causes no new assignments.

Figure 2.1: K- means Algorithm.

The above mentioned steps can be illustrated by drawing. Figures 2.2 A to E clarify the mechanism of a working k-means algorithm step by step. The example includes five groups and randomly determines the number of clustering (each point search which the center is closest to). Then, the points that belong to each center are determined.

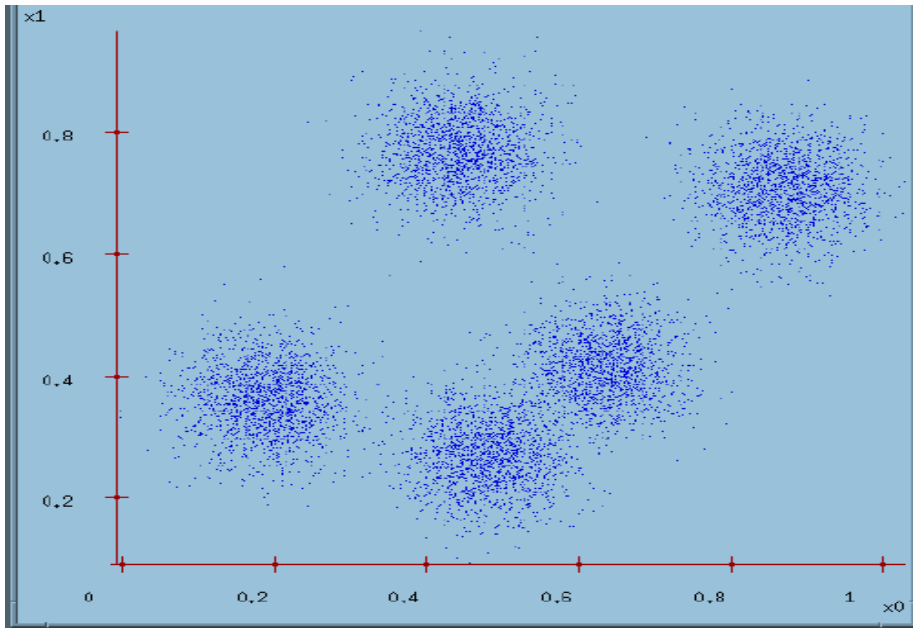


Figure 2.2. A: Determining the Number of Clusters (e.g., $k=5$) (step 1).

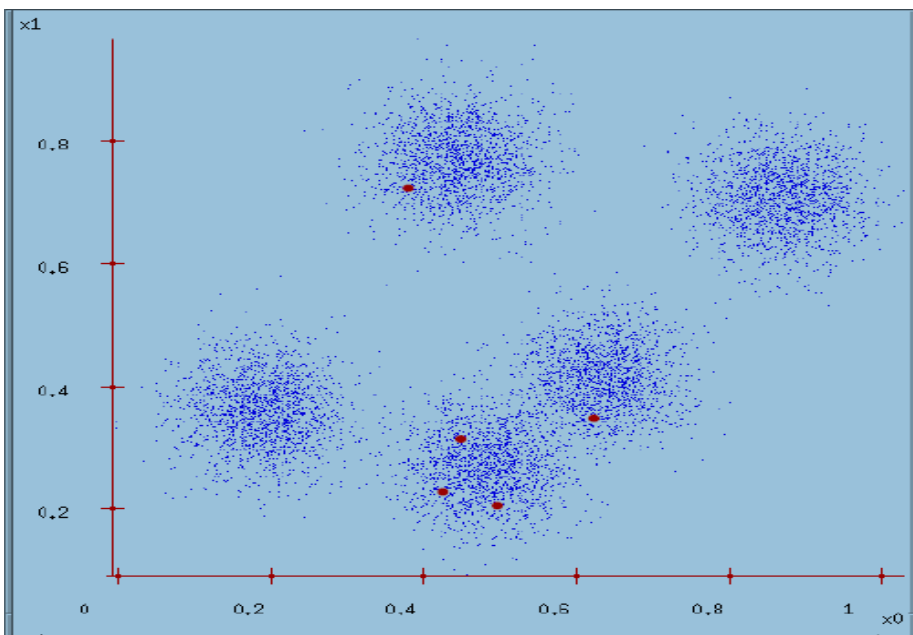


Figure 2.2. B: Randomly Guessing the k Cluster Center Locations (step 2).

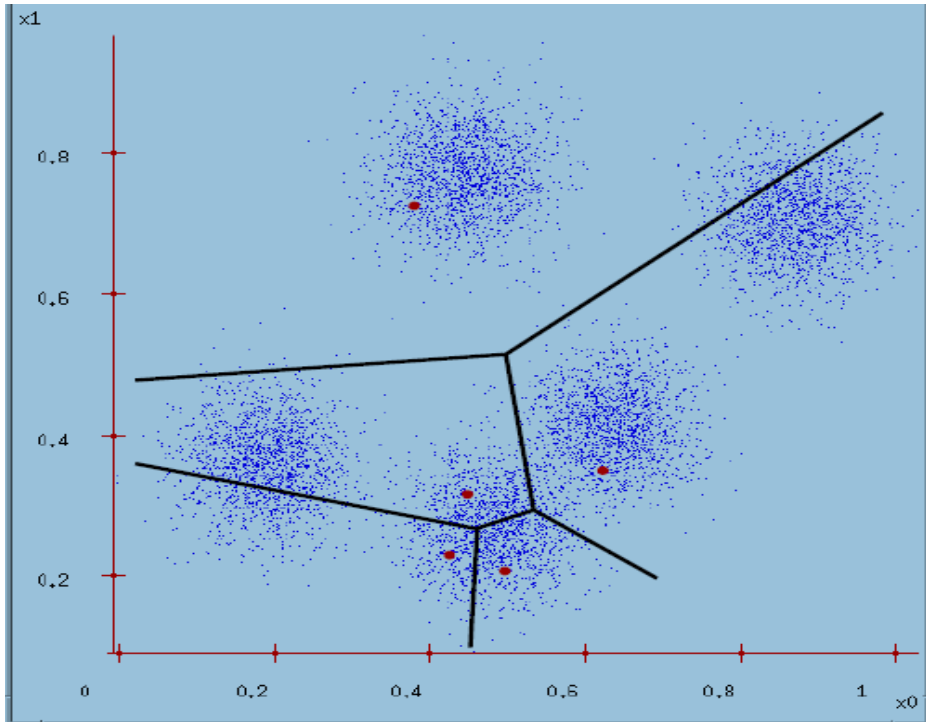


Figure 2.2. C: Determining Which Center Each Data Point is Closest to; each center thus “owns” a set of data points (step 3).

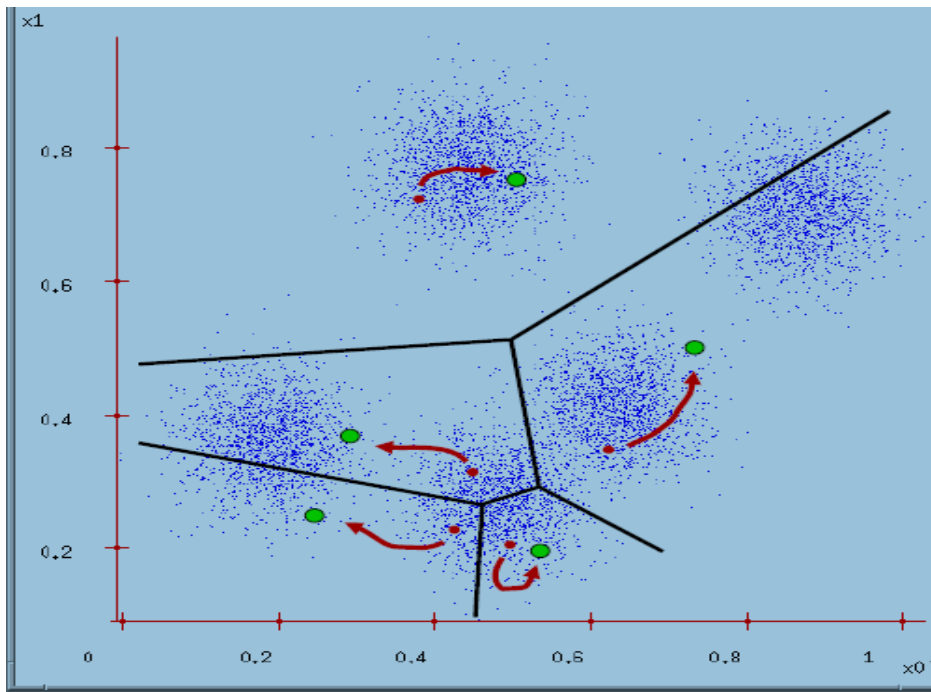


Figure 2.2. D: Finding the Centroid of the Points Owned by a Center (step 4).

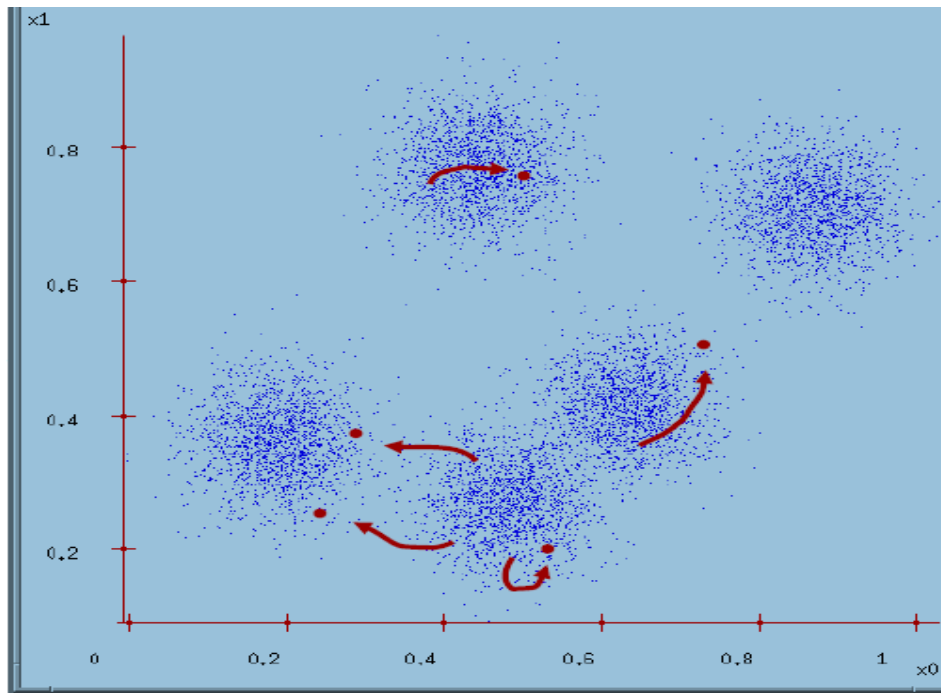


Figure 2.2. E: Repeat Until Terminated (step 5).

B. Setting up K-means Algorithm Parameters

K-means is a very popular vector space clustering method. It partitions a set of points in a feature space into a pre-specified number of clusters to minimize the sum of square distances between the points of a cluster and the cluster centroid. Although the k-means procedure does not guarantee the determination of a globally optimal solution, it usually produces good results and has been utilized widely in clustering applications. K-means is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem.

K-means follows a simple and easy procedure of categorizing incoming data by identifying a specific number of groups (clusters bearing fixed k) a priori. The main idea is to determine k centroids (one for each block). This involves placing centroids in a “cunning” manner because having different positions produces different outcomes. Thus, the best choice is to place centroids as far apart as possible.

The next step is to take every point belonging to a specific set of data and link it to the closest centroid. Suspended points are completed, and the first step is implemented in the early-age group. At this point, one needs to re-account k-new centroids as the barycenter of the blocks produced from the previous step. Following this new k centroids, a new binding has to be performed between the same data set points and the closest new centroid. A loop is then created. Owing to this loop, the k centroids change their positions, step by step so as not to be further changed.

In other words, the centroids do not move anymore. The effectiveness of the clustering depends on the k-means algorithms. Figure 2.3 shows the steps for a working k-means algorithm.

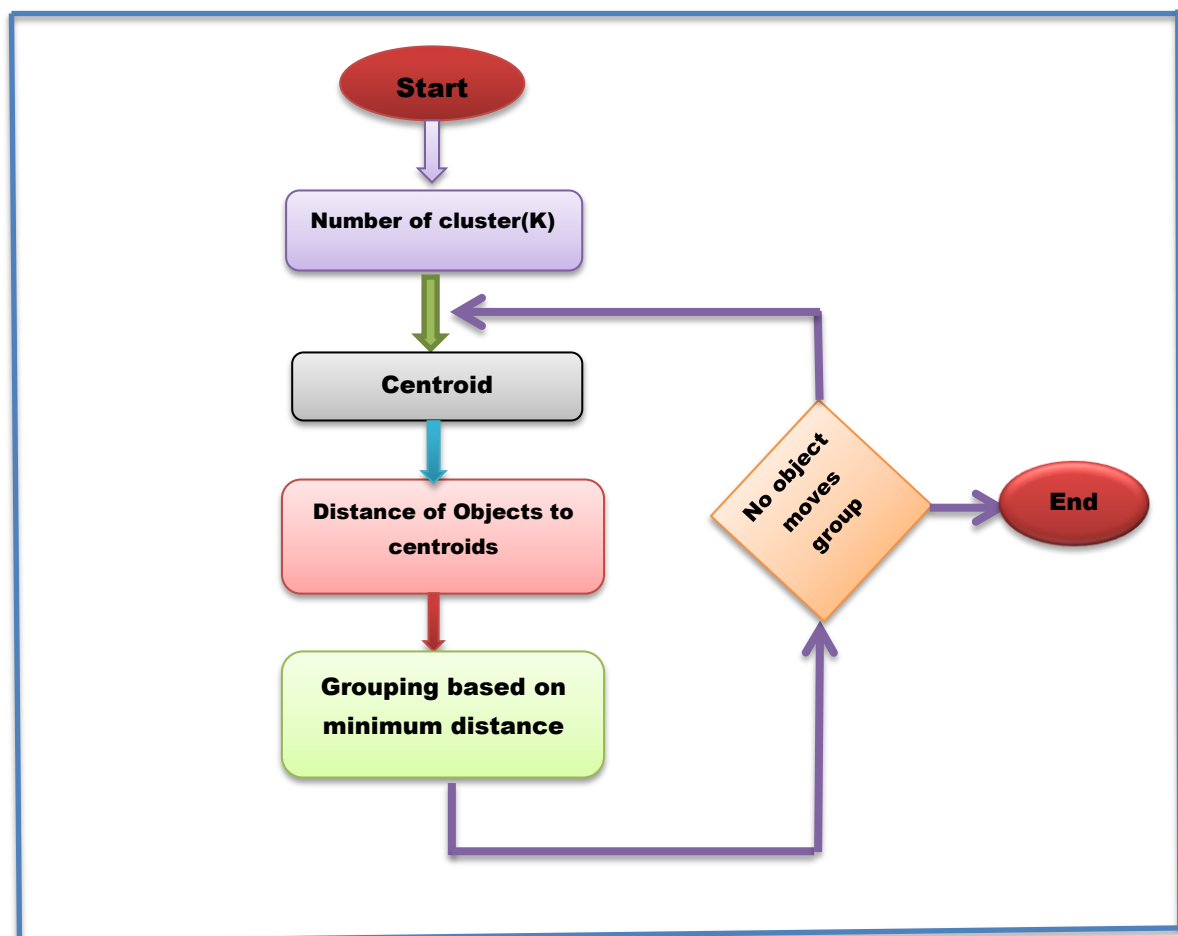


Figure 2.3: Steps of K-means Algorithms

The number of clusters (k) is as an input parameter. An option that is convenient for k can produce unsatisfactory results. Hence, when performance k -media are used, one must run a diagnostic check to specify the number of clusters in the data set. In this research, the number of clusters is 20 because the dataset includes 20 groups.

The similarities were calculated by cosine distance because the use of cosine similarity is very popular in high-dimensional positive areas. For instance, in retrieving information and analyzing texts, each term is theoretically set in different dimensions and characterized by a vector document, where the value of each dimension corresponds to the number of times the term appears in the document.

Cosine similarity then provides a useful benchmark of the extent of potential similarity in terms of subject matter and documents (Giller, 2012). This technique is also utilized to measure the coherence within groups in the field of data mining. One of the reasons cosine similarity is popular is that it demonstrates an efficient evaluation performance, particularly for sparse vectors that have to be regarded only as non-zero dimensions (Ricci, Rokach, & Shapira, 2011). The cosine measure yields high similarity values for documents that share the same set of words with high term frequencies and low values to those that do not (Shaban, 2006).

The first step of this algorithm is to determine the number of clusters. Then, the algorithm takes the first set of data k as single-element clusters and sets the remaining data $N-K$ with the closest centroid of the set. After each assignment, the centroid of the gaining cluster is recomputed. The algorithm then calculates the distance of each cluster from the centroid

by using the distance metric (cosine distance). This process is repeated until convergence is achieved. Grouping of similar data is based on the minimum distance in one group.

2.3.2 HIERARCHICAL CLUSTERING

Data are organized by hierarchical clustering in a hierarchical structure according to the proximity matrix. A dendrogram represents the result of hierarchical clustering (HC). Its root node stands for all the data groups, and each leaf is considered a data item. Thus, the middle nodes depict the number of objects that are close. Dendrogram height is usually based on the space between things, between clusters, or between a thing and a cluster (Xu & Wunsch, 2005).

A dendrogram can extend from the leaves to the root (agglomerative approach) and vice versa (divisive approach) by amalgamation or by separating clusters at each stage (Theodoridis & Koutroubas, 1999).

Gil-García, Badia-Contelles, and Pons-Porrata (2006) presented hierarchical algorithms that include two types: hierarchical compact and hierarchical star algorithms. These algorithms were estimated by utilizing document collections. The result of the experiment reveals that the proposed approaches are faster than conventional hierarchical algorithms and achieve minor hierarchies while exhibiting comparable clustering superiority.

Pons-Porrata, Berlanga-Llavori, and Ruiz-Shulcloper (2007) discussed the subject of finding a system to show that a topic discovery system aims to reveal the underlying knowledge presented through the news streams. This system involves partitional and agglomerative approaches, the main advantages of which are maximized. Lastly, a fresh condensation approach that depends on tester theory was introduced.

Gil-Garcia and Pons-Porrata (2010) also discussed two approaches related to hierarchical algorithms: dynamic hierarchical compact and dynamic hierarchical star. The goal of these methods is to establish a cluster hierarchy that works with a dynamic data group. The former identifies the disconnected hierarchies of clusters, and the latter determines interference hierarchies.

The experimental results of a number of benchmark text groups indicated that these approaches are capable of generating hierarchical clustering solutions in dynamic environments successfully and professionally. They also present hierarchies that are easier to look through than those of traditional algorithms by utilizing a function that demands dynamic clustering, such as information association, determination of document taxonomies, and hierarchical subject recognition.

This approach implies the creation of the subject compendium. The experimental results for topic detection and tracking (TDT) collection reveal the approach's advantages and effectiveness as a TD system and as a sorting and recapitulation instrument. These algorithms generate many hierarchies. Therefore, they consume much time gathering the resemblance that links the new cluster and the residual clusters in each hierarchy's stage.

2.3.3 DENSITY-BASED CLUSTERING

Density-based clustering is one of the key methods for information mining clustering. Identifying the clusters that are built on density is easy because the process is not restricted to the cluster. This study provides a survey of current density-based algorithms, namely, DBSCAN, VDBSCAN, DVBSAN, ST-DBSCAN, and DBCLASD. The survey is based on the vital parameters required for a clustering algorithm.

Analyzing the algorithms through the parameters is essential to provide meaning to the clusters have meaning (Parimala, Lopez, & Senthilkumar, 2011). The type of density determines the region of density detached by the area with low density. Density-based algorithms have two advantages. The first advantage is the possibility of identifying the cluster with a random shape, and the second is the non-importance of the user who inserts the cluster's number (Q. Liu, Deng, Shi, & Wang, 2012).

Many researchers have addressed this type of algorithm. For example, Y. Chen and Tu (2007) proposed D-Stream, a new framework for clustering stream data. This algorithm maps each incoming datum into a grid and then calculates the density of each grid. The grids are clustered by using a density-based algorithm. Unlike earlier algorithms that are dependent on k-means, the proposed algorithm can discover clusters of arbitrary form.

Furthermore, a density decaying plan is generated; this plan can modify the clusters in real time and describe the behavior of the data stream. This method reduces data stream clustering at high speed without demeaning the value of clustering. Kisilevich, Mansmann, and Keim (2010) presented a fresh density-based clustering algorithm called P-DBSCAN, which is based on DBSCAN, for the investigation of location and actions through the use of a group of geo-tagged photos. The researchers introduced two ideas, namely, density threshold and adaptive density.

2.3.4 GRID-BASED CLUSTERING

This clustering type is vital to a hierarchical clustering algorithm. It considers cells instead of the points of data. The entire clustering operations are executed on the data space instead of the basic data objects. Grid-based approaches are more common than other

conventional types because of their computational efficacy. Finding a suitable grid size is the main characteristic of this type of algorithm. Many algorithms achieve good grid sizes, but the real-life data are dense or sparse. Several algorithms achieve optimal grid size, but real-life data are dense or sparse (Parikh & Varma, 2014). Several studies have been conducted on this subject.

For instance, Liao, Liu, and Choudhary (2004) established an algorithm through adaptive mesh refinement (AMR). Dynamic meshes with various granularities and densities can be used in the algorithm. For data with irregular or concentrated distributions, a uniform grid would not result in good clustering quality if irregularly distributed data exist. The adaptability of the AMR system allows the clustering to provide sufficient resolution to the area that requires it through the use of fine grid meshes. The results of the study revealed the proficiency and usefulness of the proposed algorithm when compared with grid methods that employ single uniform meshes.

Parikh and Varma (2014) proposed numerous grid-based algorithms that achieve suitable sizes during grid construction. If the size is very large, more than one cluster may be combined into a single one. When the size is very small, a cluster might be distributed into numerous sub-clusters. Consequently, determining the appropriate size is a serious issue.

The data on clusters with varying densities and illogical shapes are also another problem; a global density value does not result in clusters with small densities. This condition is called as the problem of cluster locality. Another issue is the choice of inclusion condition for the formation of well-organized clusters. Table 2.2 shows the advantages and disadvantages of each type of clustering method.

Table 2.2
Summary of Clustering Methods

Author	Clustering method	Advantages	Disadvantages
<i>Saad de la, et al.(2006); V.Sureka & S.C.Punitha (2012); Kaur , Kaur (2013).</i>	<i>Partitioning method</i>	<i>Reduced time complexity; high clustering quality</i>	<i>Does not consider the semantics of features (terms) selected for clustering; extremely sensitive to the initial centroids because the centroids are randomly selected.</i>
<i>Gil Garcia & et al. (2006) Badia, contelles & et al. (2007)</i>	<i>Hierarchical method</i>	<i>Effective and efficient</i>	<i>Slow in generating a summary of a large document</i>
<i>V.Sureka & S.C.Punitha(2012); Kisilevich, et al.(2010)</i>	<i>Density method</i>	<i>Provides better clustering both in terms of F measure and accuracy</i>	<i>Need to be further investigated</i>
<i>Parikli , Varma (2014)</i>	<i>Grid method</i>	<i>Computational efficiency</i>	<i>Merging condition to form efficient clusters; and clusters with variable densities and arbitrary shapes</i>

This table illustrates of previous studies that dealt with clustering algorithms and the advantages and disadvantages of each type of clustering methods. Where in this study used one type of partitioning method called k-means.

2.4 EVALUATION METHODS

Many different performance measures are used by different research groups to evaluate clustering algorithms and systems. Different measures may result in different comparison results. Generally, most evaluation methods can be divided into internal and external quality measures. In internal quality measures, some external information, such as correct class labels, is unavailable. The goodness of clustering results can be measured based only on the resulting partitions. Steinbach et al. (2000) proposed the use of overall similarity as an internal quality measure.

To assess the quality of results produced by a document clustering process, two classical indices are generally employed: precision and recall. Precision P is the fraction of all correct answers included in a produced set of answers. Recall R is the fraction of the responses that are actually correct from all possible correct answers.

These evaluation indices are borrowed from IR literature. In IR, precision is utilized to indicate the fraction of all relevant documents included in a ranked list of retrieved documents. Recall is the fraction of the top responses that are actually relevant in the entire document set (Chakrabarti, 2003; Shaban, Basir, & Kamel, 2006).

Precision and recall have been adopted to evaluate other mining process results, with a minor alteration in their definitions. For instance, in document clustering, the recall for cluster/class c is the ratio of processed documents manually classified as c with regard to the entire document set. Precision is the ratio of the documents clustered by the process as c that were also manually classified as c with regard to the produced set of results (Chakrabarti et al., 2008).

In IR, recall is interpreted as the ratio of the information that has been correctly extracted and precision as the proportion of the extracted information that is correct (Eikvil, 1999). Combination methods, such as the F-measure, have been also utilized. The F measure value of each class is calculated and then combined to obtain the F measure of the entire set. The F measure combination of precision P and recall R in a single measurement and is computed as (Croft et al., 2010; Özgür et al., 2005) as follows:

$$F_i = \frac{2R_iP_i}{(R_i + P_i)} \quad (2.2)$$

Where: R_i is the recall of class i (percentage of instances correctly predicted as class i out of all true instances in class i) and P_i is the precision of class i (percentage of instances correctly predicted as class i out of all instances predicted as class i). R_i and P_i are obtained as:

$$R_i = \frac{TP_i}{(TP_i + FN_i)} \quad \text{and} \quad P_i = \frac{TP_i}{(TP_i + FP_i)} \quad (2.3)$$

Where: TP_i is a true positive for class i , FN_i is a false negative for class i , and FP_i is a false positive for class i (Özgür et al., 2005). Thus, using the F measure, the relative performance of systems having different values for recall and precision can be easily observed with another technique.

Accuracy is also an evaluation metric from machine learning without considering ordinal class information. Out of all the clustered instances, it calculates how many are correctly clustered. Given confusion matrix M with rows denoting actual values and columns denoting predicted values, accuracy is computed as:

$$accuracy = \frac{\sum_{i=1}^n M_{ii}}{\sum_{i=1}^n \sum_{j=1}^n M_{ij}} \quad (2.4)$$

Where n is the total number of classes. The F- measure and accuracy were used as evaluation criteria for the our experiments in this study. Several studies in the same field of research have employed these criteria to evaluate results (Wang, 2005).

2.5 SUMMARY

This chapter provided a detailed study of the types of clustering methods and a review of previous studies on these algorithms as well as the advantages and disadvantages of each type. This chapter also presented one of the most common partitioning algorithms, k-means algorithm, which has been addressed by many researchers. The types of text representation schemes were discussed, with focus on the dependency graph that was used in this research to solve the sparsity problem.

CHAPTER THREE

RESEARCH METHODOLOGY

3.1 THEORETICAL STUDY

Review of previous studies indicates the inefficiency of some of the techniques used in the text representation and reviewed some of these techniques and discuss their disadvantages. As mentioned in Chapter II, researchers have tried to make improvements in methods of text representation either by enhancing existing method or to propose new mining technique. Most of the problems that exist in the text representation scheme include: high dimensions of representation, ignoring the link and semantic relationship that exists between the terms in a document.

This chapter moves toward the proposals to resolve the referred issues and offers a series of operation that have been implemented in order to investigate objectives of the study. Section 3.2 discusses the research design performed in this methodology and presents the steps to improve the text representation scheme using graph based and attempt to compare the performance of the proposed representation using standard clustering algorithm namely k-means algorithm.

3.2 RESEARCH DESIGN

The proposed document clustering using graph based text representation combines the concept weighting or semantic weighting and clustering algorithm k-means. The proposed methodology includes five phases, namely, document collection, document preprocessing including four steps: split documents, tokenization, stemming and pos-tagging, and third phase in this research is text representation schemes, which contain parsing and

constructing dependency graph, then applying text clustering algorithm using one of partitioning algorithms called k-means and finally evaluation measurement as shown in Figure 3.1.

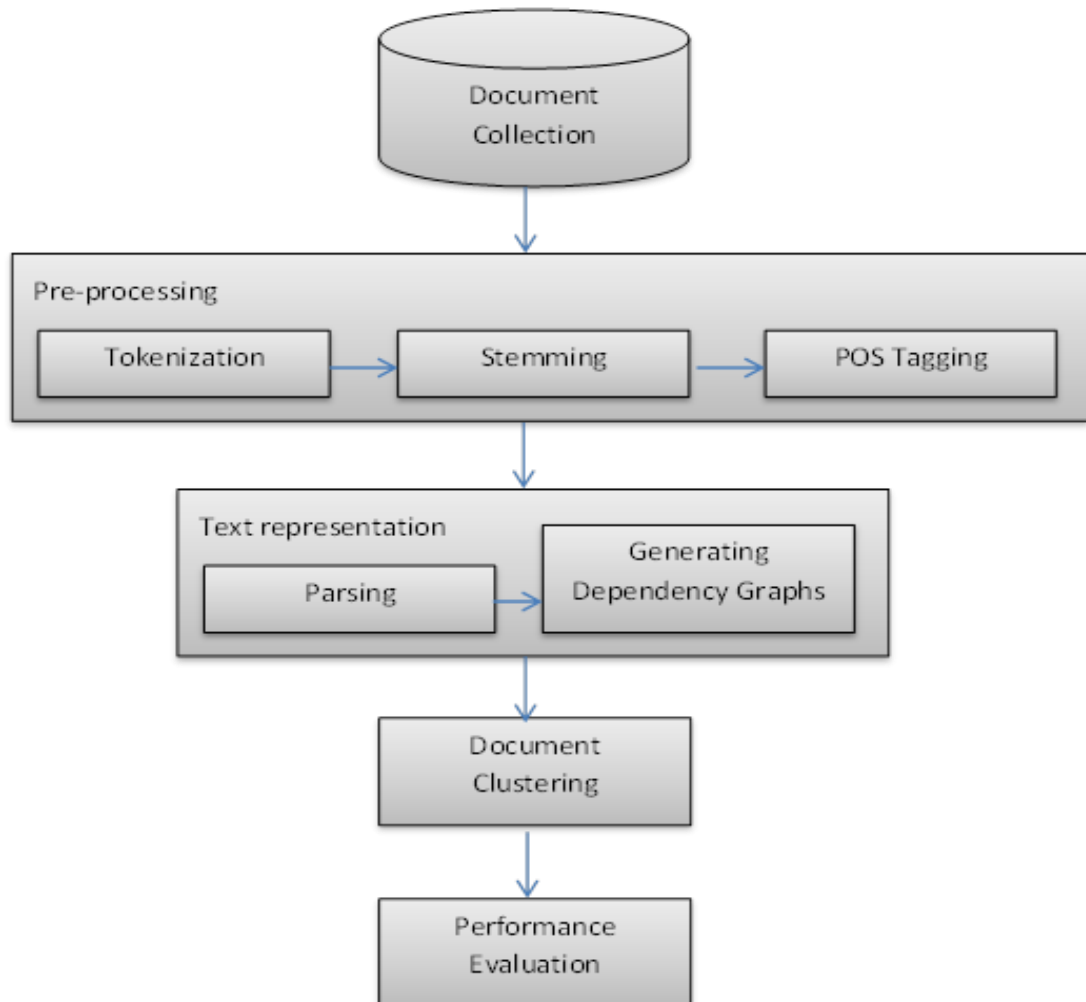


Figure3.1: Research Design

3.2.1 DOCUMENT COLLECTION

The data collection used in this study is the 20 Newsgroups. The 20 Newsgroups data set compiled by Lang (1995) is a set of approximately 20,000 newsgroup documents. The data are organized into 20 various newsgroups, each conformity to a different topic.

Some of the newsgroups are very closely associated to each other (comp.sys.ibm.pc.hardware / comp.sys.mac.hardware), and others are irrelevant (misc.forsale / soc.religion.christian). The Figure 3.2 shows the list of the 20 newsgroups, divided depending on subject matter. Where shows the data that we use in the search classified by the similarity of threads.

"comp.Graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x"	"rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey"	"sci.crypt sci.electronics sci.med sci.space"
"misc.forsale"	"talk.politics.misc talk.politics.guns talk.politics.mideast"	"talk.religion.misc alt.atheism soc.religion.christian"

Figure 3.2: The 20 Newsgroups

The 20 newsgroups group becomes a popular data set for experiments in text applications of machine learning techniques, such as text clustering and text classification (Slonim & Tishby, 2000).

3.2.2 DOCUMENT PRE-PROCESSING

Pre-processing consisting of procedure that transforms the text data in the document to a structure templates for text mining. The main goal of processing is to get basic features or key terms from online news text documents and enhance the relevancy between words and document and the relevancy between words and category.

A document almost contains number of not necessary words that could negatively affect the clusters of the document. In this research using programming language python to execute the work.

There are four steps that must be followed to clean up the text of phrases or words, namely: Split document into sentences, Stemming, Tokenization and Part of Speech Tagging.

A. Split Documents

The first step in pre-processing split each document into sentences using Python programming language.

B. Stemming

Stemming is is the process reduction of inflected words to their stem, root or base form or in general, a written word form. Stemming is used in determining domain vocabulary in domain analysis (Dolamic & Savoy, 2008). For example, stemming algorithms should be determined the string "cats" ("possibly "catlike", "catty" etc") where based on root "cat", "stemmer", "stemming", "stemmed" as depend on "stem".

A stemming reduces the words "fishing", "fished", and "fisher" to the root word, "fish". On the other hand, "argue", "argued", "argues", "arguing", "argus" reduce to the stem "argu" (illustrating the case where the stem is not itself a word or root) but "argument" ,"arguments" reduce to the stem "argument".

In this study use the Porter Stemmer algorithm, used this algorithm because it was simple and works very effectively (Hull,1996; Christopher, et al., 2008; Karaa & Ben, 2013). There is no basis for this linguistic approach made by assuming that do not have a stem dictionary the program is given a clear list of suffixes (Punitha, 2012).

C. Tokenization

Tokenization is the process of breaking a stream of text into words, phrases, symbols, or other meaningful elements called tokens. The list of tokens becomes the input data for further processing such as parsing or text mining. Tokenization is useful in linguistics, where it is a form of text segmentation, and in computer science, where it forms part of lexical analysis (Fares, Oepen, & Zhang, 2013).

Tokenization used to determine sentence borders and to separate the text into a stream of individual tokens (words) by removing extraneous punctuation marks. The text is separated into words with spaces, line breaks, and other forms of word termination in the English language (or any other language).

Document texts must be correctly entered for processing. Most punctuation marks are separated from their adjoining words, and constructions are split into constituent morphemes such that each morpheme is tagged separately. For example, “I’m” will be tokenized to “I” and “m”. This tokenization allows for the additional analysis of each element separately. Therefore, “I” can be in the subject noun phrase, whereas “m” is the head of the main verb phrase. Tokenizations of subtleties for hyphens, ellipsis, dots, and dashes, likewise exist.

The tokenizer can perform parsing operations and other forms of evidence collection on isolated lines of text. Tokenizers distinguish individual lines and the relationships between them. A finite state machine lexical analyzer with heuristic rules for finding the border is used to decide on the best separation of the sentence, paragraph, header, and other text regions. For example, the presence of an English function word such as “the” at the end of a line adds evidence for the existence of prose or a sentential region of lines (Huang, Simon, Hsieh, & Prevot, 2007).

For example:

Input: I shot an elephant in my pajamas

Output: 

D. Part of Speech Tagging

The Part-of-speech (POS) tagger is responsible for assigning the possible syntax classes of words found in the document (e.g., known words, unknown words, and spelling errors). The POS tagger is distributed throughout a number of passes. Syntactically unambiguous words are initially tagged by utilizing a list of English words (a lexical lookup) and their possible syntax classes.

Tagging of ambiguous words is deferred to passes dealing with clausal patterns to utilize the context and enable accurate POS tagging. The differentiation of unknown and misspelled words can be accomplished by setting a threshold of editing distances between the words and the lexicon.

Segond et al. (1997) observed that POS tagging solves semantic ambiguity to a certain extent (40% in one of their tests). The recognition process may be extended to recognize known expressions or phrases. Phrasal recognition can occur at various points to recognize relevant idioms and collocations. This function can be built as a data element in the knowledge base to enable the attachment of sequences of words. The chief parts of speech in English are noun, pronoun, adjective, determiner, adverb, verb, preposition, conjunct, and interjection POS.

3.2.3 TEXT REPRESENTATION SCHEME

Previously clustering methods are represented as "bag of words" in n-grams or (BOW model) without regard to the relations between the words. The text representation model is very fundamental and crucial for the clustering results.

In this study suggest DG (Dependency Graph) to solve this problem, where representation of every document as a dependency graph, the nodes compatible to words which could be seen as meta-descriptions of the document, while the edge representation the relations between pairs of words. This is the process of converting text into graphs and it involves two steps as follows:

A. Parsing

Sentences are parsed to identify their syntactic structure. Thus, the "Link Grammar Parser" (LGP) was implemented. This parser provides the syntactical relation between words in a sentence because it is based on dependency grammar and depends on context-free grammars.

Therefore, LGP is easier to handle than the more advanced parsing techniques but can simultaneously provide a much deeper semantic structure than the standard context-free parsers (Suchanek, Ifrim, & Weikum, 2006).

A dependency parser is needed to obtain word relationships from the original sentences. The goal of a parser is to analyze an input sentence and to produce the corresponding (most preferred) parse tree as its output.

Example sentence: ['I', 'shot', 'an', 'elephant', 'in', 'my', 'pajamas']

After parsing : " (S (NP I) (VP (VP (V shot) (NP (Det an) (N elephant)))) (PP (P in) (NP (Det my) (N pajamas))))".

B. Construct Dependency Graph

In this model, a representation of each document as a dependency graph where each node corresponds to a word that can be seen as a meta-description of the document. The edges between nodes are used to catch the semantic relations between word pair.

Dependency graph is projective because of when writing all the words in linear order. Edges can be extracted without crossing over the words. This is equivalent to saying that word and all descendants it's on (dependents and dependents of her dependents, etc.) constitute a connected series of words in a sentence (Bird, Klein, & Loper, 2009).

After complete parsing the output become as matrix means create an $(n-1) \times (n-1)$ matrix as a list of lists in Python, and then construct dependency graph (Bird, Klein, & Loper, 2009).

A dependency representation is a named directed graph, the arcs represent dependency relations from heads to dependents and the nodes represent the lexical items. As shown in the example in Figure 3.3.

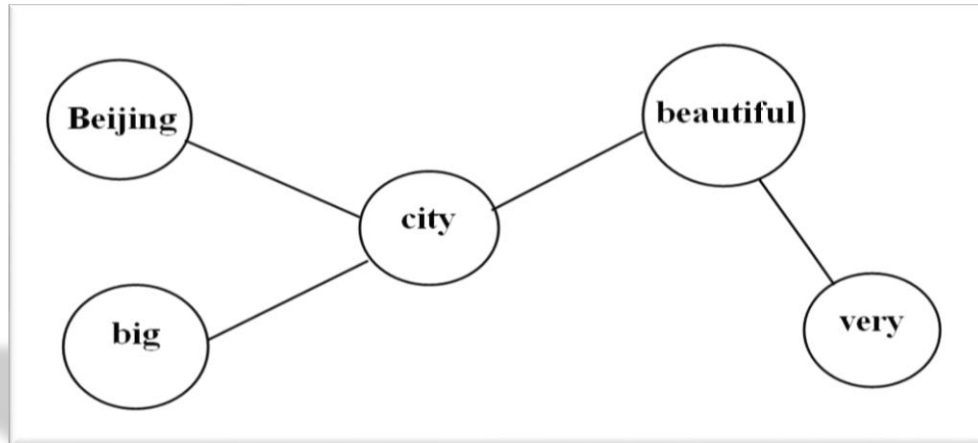


Figure 3.3: The Dependency Graph Generated for “Beijing is a big city. The city is very beautiful

3.2.4 APPLY TEXT CLUSTERING ALGORITHM

Clustering the text represented as a dependency graph using k-means algorithm. In rapid miner many algorithms related to clustering, where including special tools using to work pre-processing for text and completed the process clustering.

3.2.5 EVALUATION AND RESULT ANALYSIS

The evaluation of the quality of the clustering task output is very important, which is measured through the assessment indicators that are used on a large scale for this purpose as clustering quality evaluation measures.

Of the leading indicators used to evaluate the assembly is precision, recall, F-measure that combines them, and accuracy (Punitha, 2012). The meaning of high recall is an algorithm restored many the pertinent consequences, while the meaning of high accuracy is an algorithm restored considerably more pertinent outcome than unrelated.

The definition of recall is the number of related documents that retrieved by a exploration divided by the sum amount of accessible pertinent documents, while we define cluster accuracy to be the total number documents clustered correctly, over the total number of documents within the corpus (Hu, Xiong, Shu, & Zhou, 2009).

Frequently, accuracy and evoke scores are not handled in segregation. Both are shared into a single gauge. Therefore, there are some examples for procedures that are a mixture of accuracy and evoke mean of accuracy and recall (Hjørland, 2010).

$$Precision(P) = \frac{TP}{TP+FP} \quad (3.1)$$

$$Recall(R) = \frac{TP}{TP+FN} \quad (3.2)$$

$$F - Measure (F) = 2 * \frac{P*R}{P+R} \quad (3.3)$$

$$Accuracy (A) = \frac{TP+TN}{(TP+FP+FN+TN)} \quad (3.4)$$

CHAPTER FOUR

DEPENDENCY GRAPH BASED TEXT REPRESENTATION

4.1 INTRODUCTION

This chapter provides the implementation detail of the proposed method and giving details steps taken to build the dependency graph model. The process of creating the DG includes syntactic and semantic analysis. Section 4.2, reviewed the document pre-processing containing several methods, split the document, tokenization, stemming and part of speech tagging. Section 4.3 details the phase for parsing text. Section 4.4 explains how builds the dependency graph. Finally, Section 4.5 summary deals of what was covered in this chapter. Figure 4.1 illustrates the process for building Dependency graph.

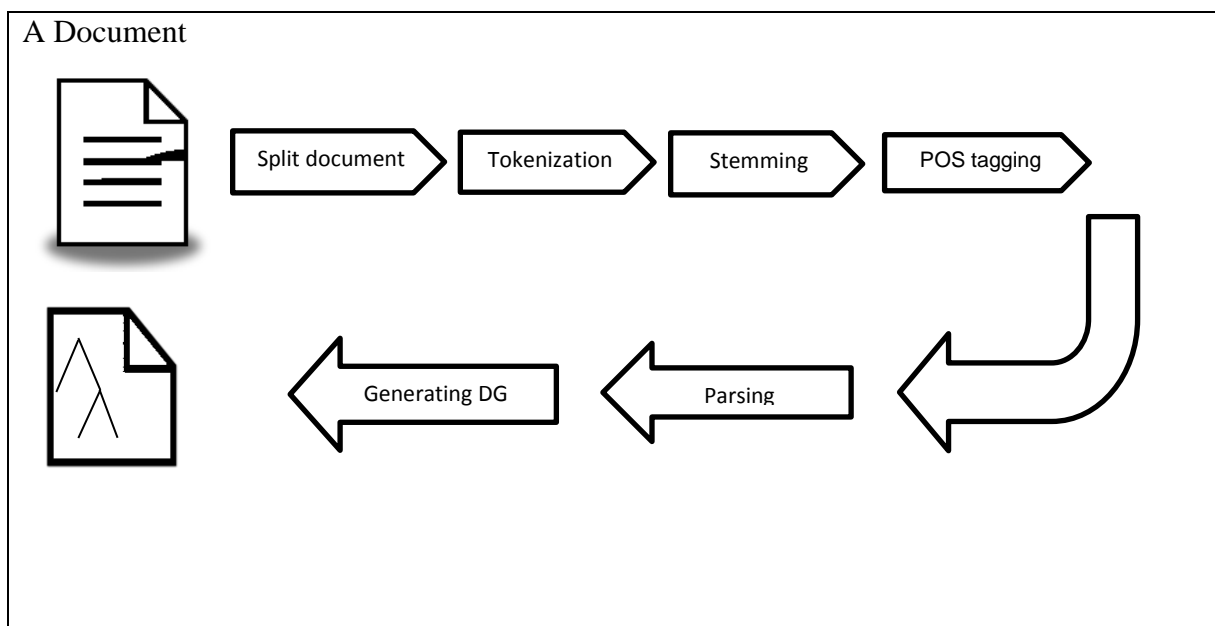


Figure 4.1: Steps Building the Dependency Graph

4.2 DOCUMENT PRE-PROCESSING

The process to create the DG semantic representation includes syntactic and semantic analysis stages. The first step is to assign the input text to a syntactic parser. This step includes processes such as sentence splitting, stemming, tokenization, part of speech tagging. The 200 documents in this phase are in English language.

The first step is to split the single document into the sentences using the Python programming language. Figure 4.2 shows an example of one document from the 20 newsgroup data called alt.atheism.

```
{alt.atheism.51164From: mccullou@snake2.cs.wisc.edu (Mark McCullough) Subject:
Re: Idle questions for fellow atheists In article
<1993Apr5.124216.4374@mac.cc.macalstr.edu>acooper@mac.cc.macalstr.edu
writes: I wonder how many atheists out there care to speculate on the face of the
world > I shot an elephant in my pajamas > A persons religious belief seems more as a
crutch and justification for actions than a guide to determine actions}
```

Figure 4.2: Single Document Called alt.atheism

This document is split into sentences using special code in python for split the document by placing some of the conditions that control the division of documents. Figure 4.3 displays special code in python for splitting single document to the many sentences. After splitting the document can see the output in Figure 4.4.

```
*Python 2.7.6 Shell*
File Edit Shell Debug Options Windows Help
Python 2.7.6 (default, Nov 10 2013, 19:24:18) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> import nltk.data
>>> tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
>>> file = open('c:/text/split.txt','r')
>>> data = file.read()
>>> print '\n-----\n'.join(tokenizer.tokenize(data))
```

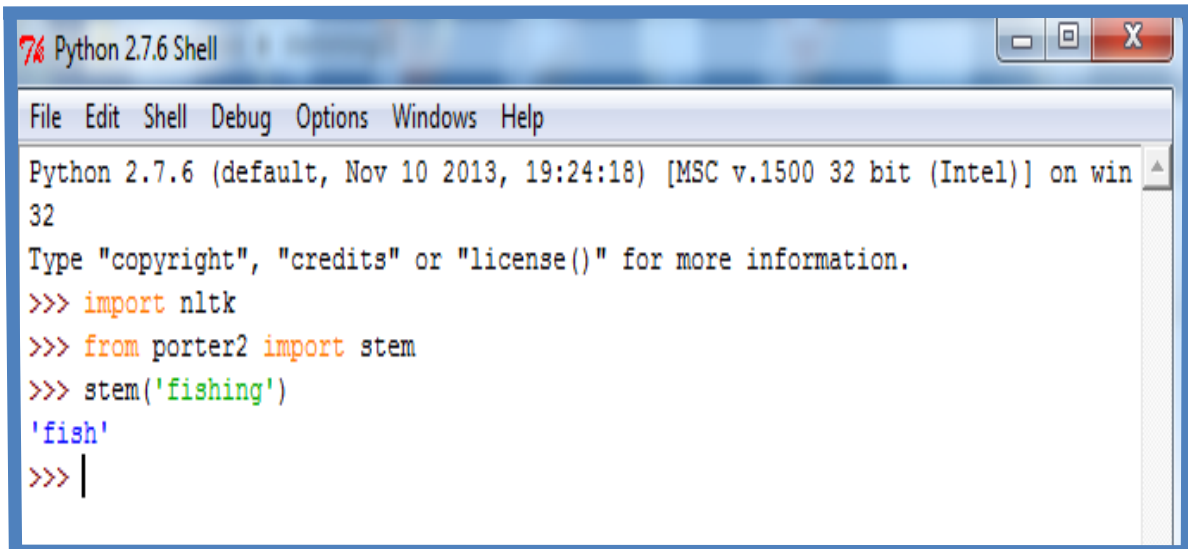
Figure 4.3: Split Single Document in Python

[I wonder how many atheists out there care to speculating on the face of the world]
[I shot an elephant in my pajamas]
[A persons religious belief seems more as a crutch and justification for actions than a
guide to determine actions].

Figure 4.4: The Example Document After Splitting Into Sentences.

The second step is the stemming process. Stemming algorithms are used in information retrieval to reduce different variations of the same word with different endings to a common stem (C. Ramasubramanian & R.Ramya, 2013).

Stemmers can help information retrieval systems by unifying vocabulary, reducing term variants, reducing storage space, and increasing the likelihood of matching documents (Korde & Mahender, 2012). Figure 4.5 displays the porter stemmer algorithm in python for only one word and how this algorithm is effective to return word to the root.



```
Python 2.7.6 Shell
File Edit Shell Debug Options Windows Help
Python 2.7.6 (default, Nov 10 2013, 19:24:18) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> import nltk
>>> from porter2 import stem
>>> stem('fishing')
'fish'
>>> |
```

Figure 4.5: Porter Stemming Algorithms

After completing the process of dividing the text into sentences move on to the next stage, which is re every word to root, in this example, the document produced three sentences.

Figure 4.6 shows the sentences before and after stemming.

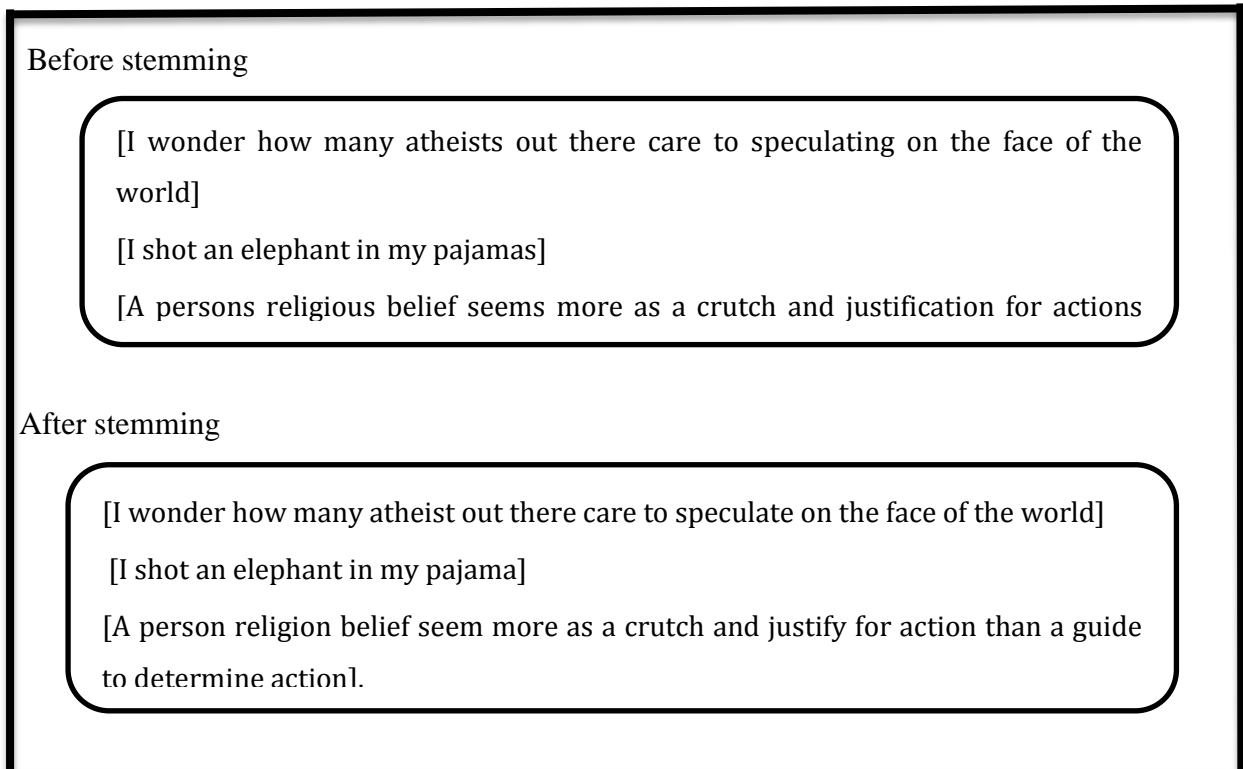
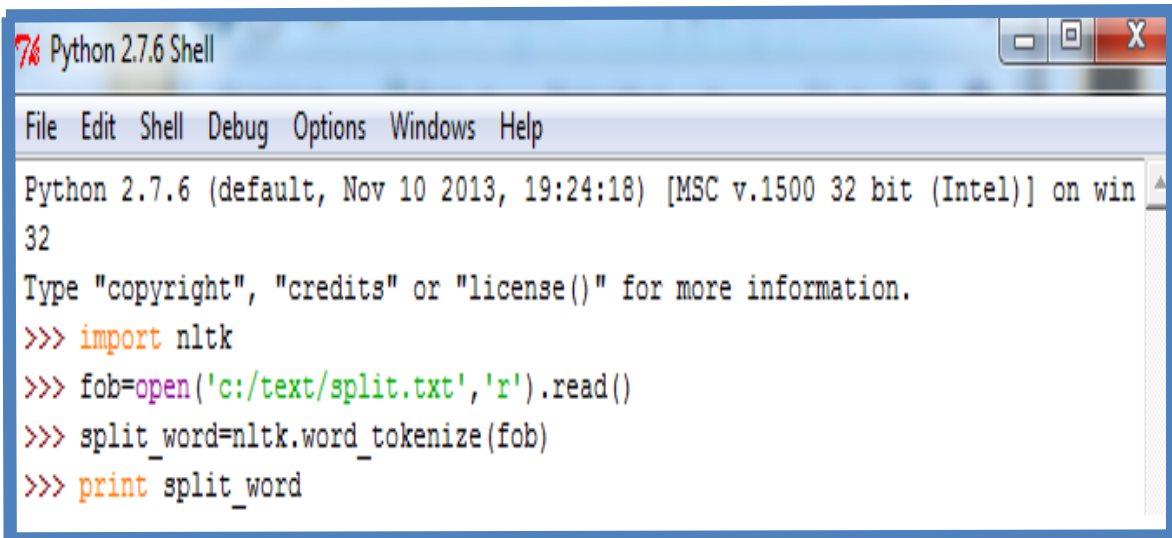


Figure 4.6: Stemming Process

The third step is tokenization. Tokenization is a process of breaking the text into a number of pieces called tokens. In this research the tokenization process is able to call a few sentences or several documents and perform the tokenization. Figure 4.7 shows the code to tokenize one document in programming language python. Figure 4.8 illustrates the output after the process of tokenization for the example document.

A screenshot of a Python 2.7.6 Shell window. The window title is "Python 2.7.6 Shell". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Windows", and "Help". The main text area shows the following code:

```
Python 2.7.6 (default, Nov 10 2013, 19:24:18) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> import nltk
>>> fob=open('c:/text/split.txt','r').read()
>>> split_word=nltk.word_tokenize(fob)
>>> print split_word
```

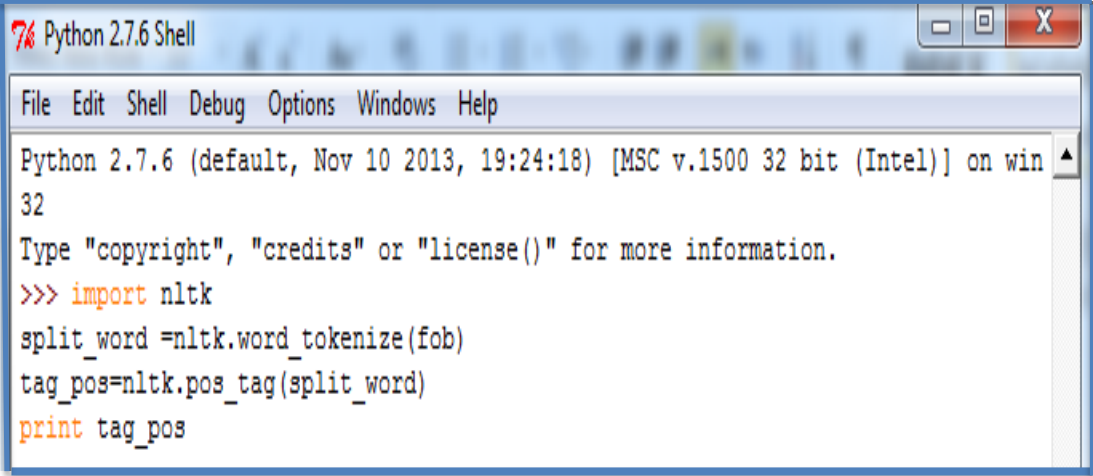
Figure 4.7: Tokenization Code in Python

```
['I', 'wonder', 'how', 'many', 'atheist', 'out', 'there', 'care', 'to', 'speculate', 'on', 'the', 'face',
'of', 'the', 'world'], ['I', 'shot', 'an', 'elephant', 'in', 'my', 'pajama'], ['A', 'person', 'religion',
'belief', 'seem', 'more', 'as', 'a', 'crutch', 'and', 'justification', 'for', 'action', 'than', 'a', 'guid',
'to', 'determine', 'action'].
```

Figure 4.8: Example Document After Tokenization Process.

The final step in the pre-processing phase is Part of Speech Tagging. Part of Speech Tagging is a group of words in a language that may occur in similar position in a sentence.

The main parts of speech in English are noun, pronoun, adjective, determiner, adverb, verb, preposition, conjunct, and interjection POS. Figure 4.9 shows the process of pos-tagging in python, and the result of executing this code can show in Figure 4.10.



```
Python 2.7.6 Shell
File Edit Shell Debug Options Windows Help
Python 2.7.6 (default, Nov 10 2013, 19:24:18) [MSC v.1500 32 bit (Intel)] on win
32
Type "copyright", "credits" or "license()" for more information.
>>> import nltk
split_word = nltk.word_tokenize(fob)
tag_pos = nltk.pos_tag(split_word)
print tag_pos
```

Figure 4.9: Pos- Tagging One Document in Python

```
[(('I', 'PRP'), ('wonder', 'VBP'), ('how', 'WRB'), ('many', 'JJ'), ('atheist', 'NN'), ('out', 'IN'), ('there', 'NN'), ('care', 'NN'), ('to', 'TO'), ('speculate', 'VB'), ('on', 'IN'), ('the', 'DT'), ('face', 'NN'), ('of', 'IN'), ('the', 'DT'), ('world', 'NN')], [(('I', 'PRP'), ('shot', 'VBD'), ('an', 'DT'), ('elephant', 'NN'), ('in', 'IN'), ('my', 'PRP$'), ('pajama', 'NN')], [(('A', 'DT'), ('person', 'NN'), ('religion', 'NN'), ('belief', 'NN'), ('seem', 'VBP'), ('more', 'JJR'), ('as', 'IN'), ('a', 'DT'), ('crutch', 'NN'), ('and', 'CC'), ('justification', 'NN'), ('for', 'IN'), ('action', 'NN'), ('than', 'IN'), ('a', 'DT'), ('guide', 'NN'), ('to', 'TO'), ('determine', 'VB'), ('action', 'NN')].
```

Figure 4.10: Example Document After Pos-tagging.

The previous steps (split document, stemming, tokenization and Part of Speech Tagging) to 200 documents are completed and the text prepares to move to the next phase text representation scheme, which consists of two-steps: The first step parsing and the second step construct the dependency graph.

4.3 PARSING

The main objective of the syntactic parsing region is to recognize a sentence and designation a grammatical structure to it, namely the parse tree. The sentences are parsed to identify syntactic structure. Furthermore, the parser must be having ability to deal efficiently with the problem of ambiguity, where one sentence or words that may have more than one parse. Parsing algorithms dependent on a grammar which is declarative formalities and it can be calculated in several possible ways.

The segment resolution passes are interspersed with chunking and the syntactic parsing passes, so as to use feedback to assign segments and their boundaries with greater confidence. The main and commonly used parsing algorithms are based on the context free grammar parsing algorithm. In this research, using a Standard English Grammar rule (Salton& McGill 1983) as follows:

$S \longrightarrow NP VP$

$NP \longrightarrow [DET] [N-MOD].NOUN [PP]$

$VP \longrightarrow VERB [NP] [PP]$

$PP \longrightarrow PREP NP$

The grammar rule suggests that all sentences (S) should be a combination of Noun Phrase (NP) followed by Verb Phrase (VP). All Verb Phrases are made of verb and followed by an optional Noun Phrase ([NP]) and an optional preposition phrase ([PP]). The noun modifiers can be adjectives to nouns and the preposition can be followed Noun Phrases. Figure 4.11 Clarify the process of parser implemented in the previous example to illustrate the output of this process.

(S I/PRP (VP (V wonder/VBP)) how/WRB (NP many/JJ atheist/NN)
(PP (P out/IN) (NP there/NN care/NN)) to/TO (VP (V speculate/VB)
(PP (P on/IN) (NP the/DT face/NN)) (PP (P of/IN) (NP the/DT world/NN))).

(S (NP I) (VP (VP (V shot) (NP (Det an) (N elephant)))) (PP (P in)
(NP (Det my) (N pajamas))))).

(S (NP A/DT person/NN religion/NN belief/NN) (VP (V seem/VBP))
more/JJR (PP (P as/IN) (NP a/DT crutch/NN)) and/CC (NP justify/NN)
(PP (P for/IN) (NP action/NN))
(PP (P than/IN) (NP a/DT guid/NN)) to/TO (VP (V determine/VB)
(NP action/NN)) ./.)

Figure 4.11: Example Document After Parsing

4.4 GENERATING DEPENDENCY GRAPHS

The semantic dependency graph (DG) model is a representation scheme that is deemed to be suitable for document mining processes. This modeling process focuses on the ability to express distinct readings of sentences as distinct formulas that capture their intuitive structures and meanings.

Each DG represents information in a document sentence by sentence. Each sentence is converted into a directed graph that has concepts as its nodes and relations as its links. The sum of all sentence representations creates the document representation model.

DG represents the third step in the text representation scheme in research design. In this step, the dependency graph was constructed using special code in python. Figure 4.12 shows the output after generating dependency graph (DG) for one document.

```
[('I', 'PRP'), ('wonder', 'VBP'), ('how', 'WRB'), ('many', 'JJ'), ('atheist', 'NN'), ('out', 'IN'), ('ther', 'NN'), ('care', 'NN'), ('to', 'TO'), ('speculat', 'VB'), ('on', 'IN'), ('the', 'DT'), ('face', 'NN'), ('of', 'IN'), ('the', 'DT'), ('world', 'NN'), ('I', 'PRP'), ('shot', 'VBD'), ('an', 'DT'), ('elephant', 'NN'), ('in', 'IN'), ('my', 'PRP$'), ('pajama', 'NN'), ('A', 'DT'), ('person', 'NN'), ('religiou', 'NN'), ('belief', 'NN'), ('seem', 'VBP'), ('more', 'JJR'), ('as', 'IN'), ('a', 'DT'), ('crutch', 'NN'), ('and', 'CC'), ('justification', 'NN'), ('for', 'IN'), ('action', 'NN'), ('than', 'IN'), ('a', 'DT'), ('guid', 'NN'), ('to', 'TO'), ('determin', 'VB'), ('action', 'NN')]
```

Figure 4.12: Dependency Graph for the Example Document

The dependency graph can be constructed with the following steps:

- 1- The graph does not have vertices and edges.
- 2- The vertices and edges are added by processing each sentence in the document.
- 3- Each sentence is parsed by the dependency parser, which produces a set of words and identifies the pairwise relationships between the words.

In this research the dependency graph is constructed for 200 documents and the percentage of reduction for each document is calculated. The percentage reduction of the documents used in this research, i.e. 200 documents of 20 newsgroup dataset is found in Appendix a. Where this table represents the size of each document before and after constructing dependency graph and the percentage of reduction.

For example, in figure 4.2 shows an example of one document from the 20 newsgroup data called alt.atheism, the original number of words are 77 words and after generating dependency graph the reduced number of features are 42 words show in figure 4.12 and the percentage of reduction in this case is 46%. The average percentage reduction for all documents used in this research is 20%. Where the minimum percentage of reduction is 3% and the maximum of percentage of reduction is 78%.

From this result can be seen that the process of constructing dependency graph was able to solve the sparsity problem by reducing the number of features. The line graph shown in Figure 4.13 shows clearly the percentage of reduction in the document size for all the 200 documents before and after constructing the dependency graph.

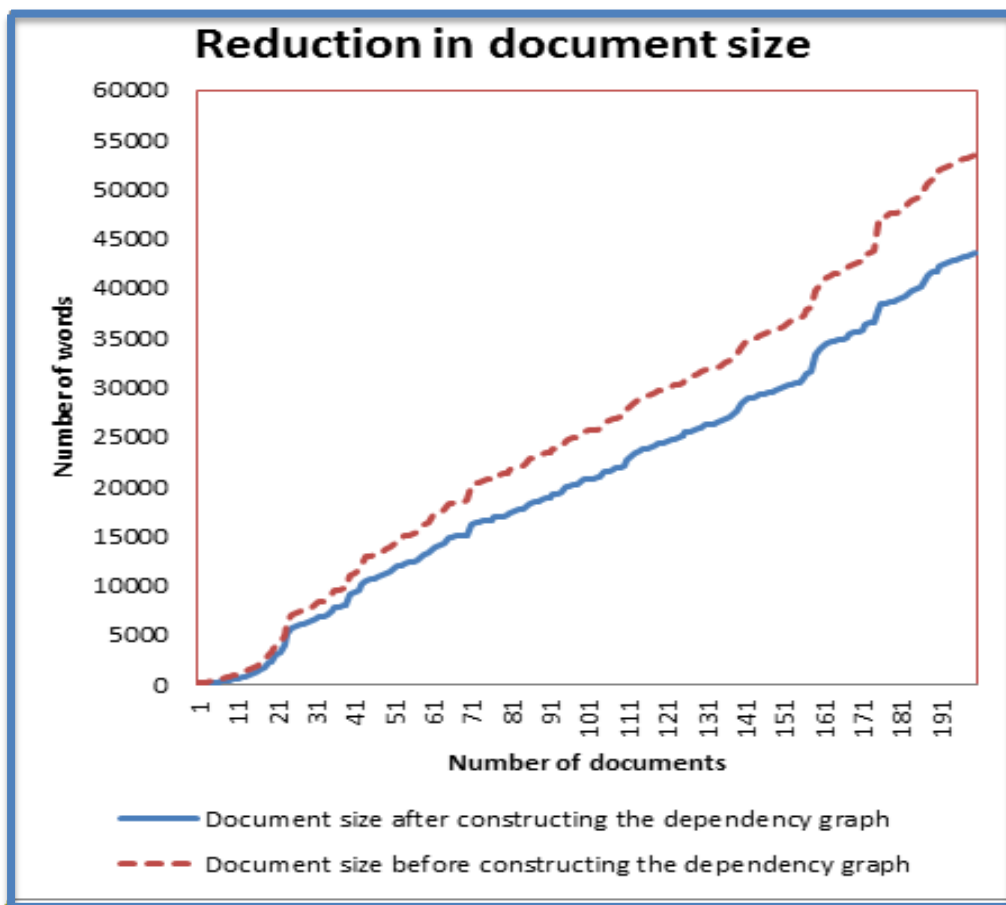


Figure 4.13: The Reduction in Document Size

Constructing the DG can solve the semantic problem as well through extracting only the meaningful sentences. A dependency graph representation is a labeled directed graph, where the nodes are the lexical items and the labeled arcs represent dependency relations from heads to dependents. Figure 4.14 illustrates a dependency graph, where the arrows point from heads to their dependents.

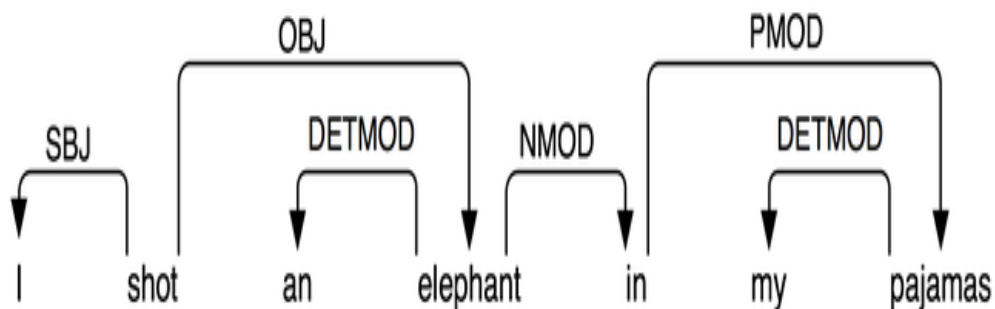


Figure 4.14: Dependency Structure: arrows point from heads to their dependents; labels indicate the grammatical function of the dependent as subject, object or modifier.

The arcs in 4.14 are labeled with the grammatical function that holds between a dependent and its head. For example, *I* is the SBJ (subject) of *shot* (which is the head of the whole sentence), and *in* is an NMOD (noun modifier of *elephant*). In contrast to phrase structure grammar, therefore, dependency grammars can be used to directly express grammatical functions as a type of dependency.

The semantic analysis can be performed using a dependency graph representation of syntax. The three main properties of dependency graphs when used as syntactic representations are thus:

- The graph consists of edges between words, and the presence of an edge between two words denotes a grammatical cooperation between those words.

- The edges are directed, and the direction of an edge between two words denotes which of them determines the grammatical behavior of the complete structure.
- The edges are labeled, and the label associated with an edge between two words denotes the nature of their grammatical relationship, the grammatical function.

Output from (DG) is represented as a tree; where this tree explains in details how the sentences connect to each other. Figure 4.15 Shows the three sentences connected together.

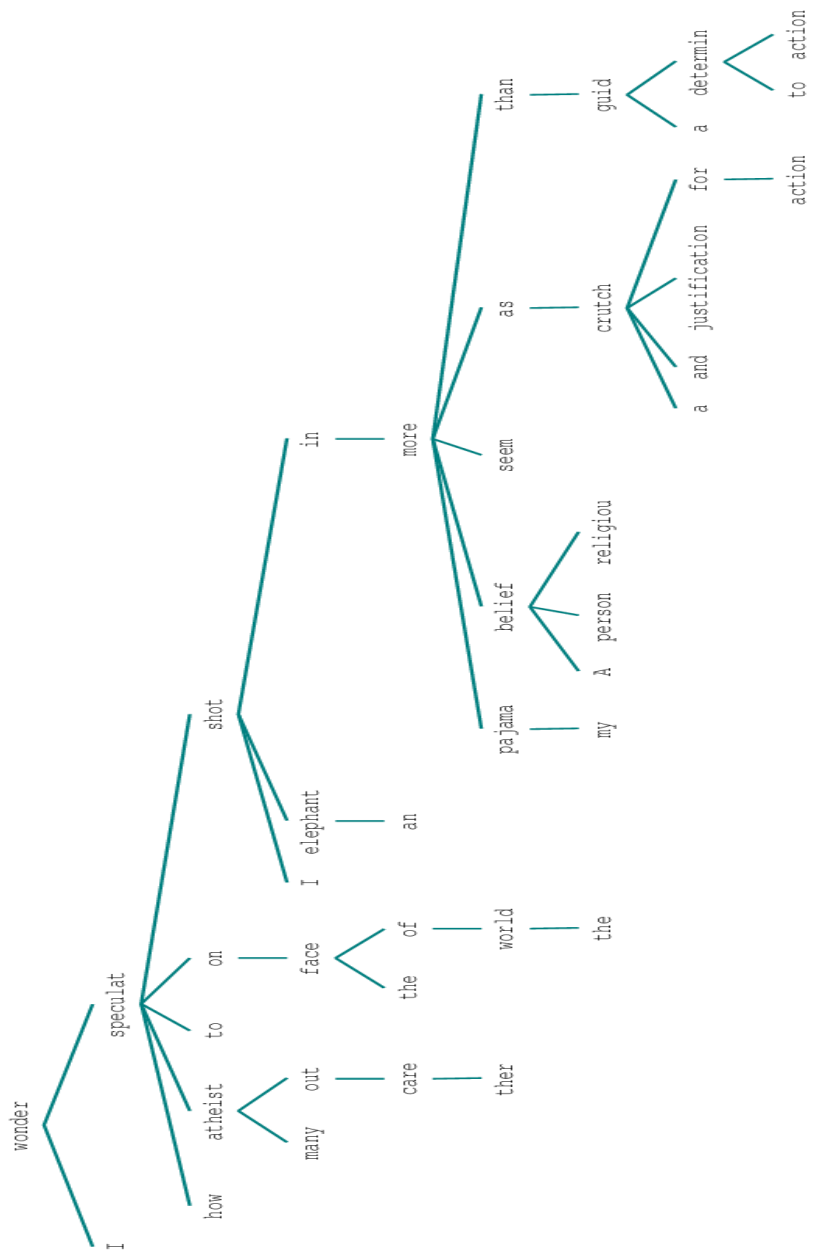


Figure 4.15: Three Sentences Connect to Each Other in The Dependency Graph

4.5 ONTOLOGY BASED APPROACH FOR SEMANTIC ANALYSIS

An ontology-based approach allows the analyst to represent the complex structure of objects, to implement the knowledge about hierarchical structure of categories, as well as to show and use the information about relationships between categories and individual objects.

The first step to represent the text as ontology is to choose the appropriate set of keywords based on the whole corpus of documents, and assigning weights to those keywords for each particular document, thus transforming each document into a vector of keyword weights. This method uses a concept weighting scheme based on ontology.

Ontology is an explicit specification of a conceptualization in a particular domain. Its importance in knowledge management, knowledge sharing and information retrieval has been realized by researchers, especially in biological and biomedical domains, where new discoveries and knowledge emerge at a fast pace. Many different ontologies have been developed in recent years. Whereas each ontology is useful for a particular domain or subdomain, the interoperability between these ontologies has yet to be built up (Tong, 2011).

In this regard, ontology is defined as a set of concepts of the interest domain organized as a hierarchical structure. Sowa (2000) defined ontology, as a catalogue of the types of things that are assumed to exist in a domain. Ontology helps to figure out what a specific term means. Ontologies provide a way to describe the meaning and relationships of terms so that a shared understanding or a consensus can be acquired among people and machines (Guarino 1998). The algorithm used to generate the ontology is as shown in Figure 4.16.

Steps:

1. Select significant terms to form the feature vector of each document, so need to sort all the terms of a given document.
2. Using the ontology weights, can find the weights of all the terms of each document and hence sort them according to their weight.
3. Then can select the top x terms together with their weights to form the feature vector of that document.
4. Calculate the new weight of each term, uses a concept weighting scheme based on ontology. Using the function with weight values as follows (Equ 4.1).

$$W = \text{Len} \times \text{Frequency} \times \text{Correlation Coefficient} + \text{Probability of concept} \quad (4.1)$$

Where W is the weight of keywords, Len is the length of keywords, Frequency is times, which the words appear and if the concept is in the ontology, then correlation coefficient =1, else correlation coefficient=0. Probability is based on the probability of the concept in the document. The probability is estimated using Equ (4.2).

$$P(\text{concept}) = \frac{\text{Number of Occurences of the Concept}}{\text{Number of Occurences of All Concepts in Document}} \quad (4.2)$$

Figure 4.16: Ontology Algorithm

The following assumptions are made during the calculation of weight

- a. The more times the words appear in the document, the more the possibility that it is the characteristic word.
- b. The length of the words will also affect the importance of words. Apparently, one concept in the ontology is related to another concept in that domain ontology. That also means that the association between two concepts can be determined using the length of these two concepts connecting path (topological distance) in the concept lattice. The length of the feature vector of each cluster is determined by the length of the feature vector of each document belonging to it and the total number of documents in that cluster.
- c. If the probabilities of one word are high, then the word will get additional weight. Finally, the system ranks the weights and selects the keywords that have with bigger weight in the pre clustering process.

Example:

A simple test - To show the process of ontology a small piece of text is chosen in Figure 4.17 consisting of 11 sentences containing 80 words which 27 of them are unknown (have no entry in the lexicon). The ontology is limited to the kernel (primitive concepts). The underlined words are unknown and have no corresponding meaning(s) in the ontology. The weights of all relations are simply zero for primitives and one for others.

Fish live in water and breathe by Gill but Humans live on land and breathe by lung. Horses and rabbits breathe by lung too. Blacky is John's horse. Blacky eats fresh grass. John is a human. He has a farm and eats fresh fruits in it. Robby lives in that farm. Robby is the name of a white rabbit who eats carrot. ... All birds except penguin fly. ... Any human who is killed, is not alive. If a human is not alive, he is dead. ...

Figure 4.17: The Simple Text in English.

After processing the text, placing p-concepts, clustering and reorganization of the ontology, the result will be obtained. Part of the resulted ontology is shown in Figure (4.18) and table (4.1). Figure (4.18) shows part of the taxonomic relations (the hierarchy) and table (4.1) shows part of the non-taxonomic knowledge and the axioms extracted from the sample text. There are also new words learned at the lexicon which their syntactic (grammatical category) and semantic (a corresponding concept in the ontology) fields are completed.

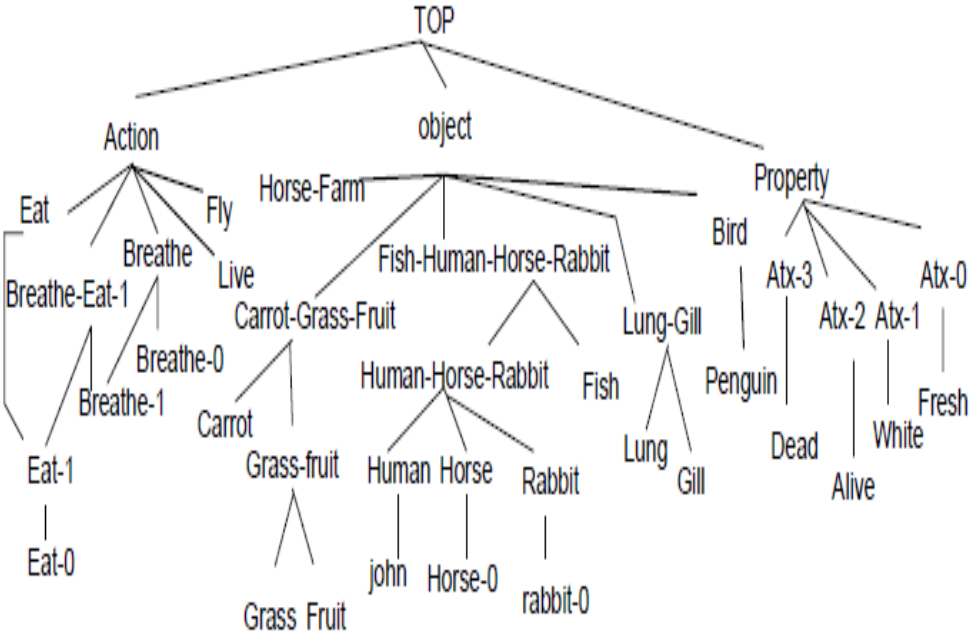


Figure 4.18: The Resulted Hierarchy After Processing the Text .

Table 4.1 *Part of Non-taxonomic Knowledge and Axioms Extracted From the Sample Text.*

(P-Agent Human-Horse-Rabbit Breathe-eat-1)	(Has-Prop Rabbit -0 Atx-0 White)
(P-Agent Fish-Human-Horse-Rabbit Breathe)	(P-Has-Prop Rabbit Atx-0 White)
(P-Agent Fish Breathe-0)	(Name Rabbit-0 Robby)
(P-Agent Bird Fly)	(P-Has Rabbit Name)
(Not (P-Agent Penguin Fly))	(Has John Horse-Farm)
(P-Agent Human-Horse-Rabbit Eat)	(P-Loc Farm Eat-Live)
(P-Instrument Lung-Gill Breathe)	(=> (and (instance-of ?x Human)
(P-Instrument Lung Breathe-1)	(instance-of ?y Kill)
(P-Instrument Gill Breathe-0)	(Patient ?x ?y))
(P-Agent Human-Rabbit Eat-0)	(Not (Has-prop ?x Atx-2 alive))
(P-Patient Carrot-Grass-Fruit Eat-1))
(P-Patient Grass-Fruit Eat-0)	(=> (and (instance-of ?x Human)
(P-Has-Prop Grass-Fruit Atx-1 Fresh)	(Not (Has-prop ?x Atx-2 alive)))
(= Horse-0 Blacky) (Is-horse-of Blacky John)	(Has-prop ?x Atx-3 dead)
(Has john blacky))
(P-Has Human Horse)	

From the previous example, clearly in the ontology there are some words that are not found in the ontology domain where one of the drawbacks associated with existing ontology systems is a general lack of names in the ontology that correspond to all the information or concepts in a document set. These words are ignored during the construction of the ontology.

When the words are represented in ontology as a tree, it can be seen that some words that are not found in ontology domain are ignored. For example killed, Robert, blacky. This will lead to lost in the meaning of some sentences. In addition the existing ontology systems is unable to map ontology terms to infer linkages to related or referenced entities in wider data sources (Gardner, 2007). However, in the dependency graph no words are ignored. All the words are constructed and used in the construction of the dependency graph.

Furthermore, there is no need for domain construction in the dependency graph because DG does not depend on domain while in ontology construction of domains is a must and these will lead to extra effort and consumes time. Figure 4.19 shows the example as dependency graph.

```

[('Fish', 'JJ'), ('live', 'JJ'), ('in', 'IN'), ('water', 'NN'), ('and', 'CC'), ('breath', 'NN'), ('by', 'IN'),
('Gill', 'NNP'), ('but', 'CC'), ('Human', 'NNP'), ('live', 'VBP'), ('on', 'IN'), ('land', 'NN'), ('and',
'CC'), ('breath', 'NN'), ('by', 'IN'), ('lung', 'NN'), ('Horse', 'NNP'), ('and', 'CC'), ('rabbit', 'NN'),
('breath', 'NN'), ('by', 'IN'), ('lung', 'NN'), ('too', 'RB'), ('Blacky', 'NNP'), ('is', 'VBZ'), ('John',
'NNP'), ('s', 'VBZ'), ('hors', 'NNS'), ('Blacky', 'NNP'), ('eats', 'NNS'), ('fresh', 'JJ'), ('gras',
'NNS'), ('John', 'NNP'), ('is', 'VBZ'), ('a', 'DT'), ('human', 'JJ'), ('He', 'PRP'), ('has', 'VBZ'), ('a',
'DT'), ('farm', 'NN'), ('and', 'CC'), ('eats', 'NNS'), ('fresh', 'JJ'), ('fruit', 'NN'), ('in', 'IN'), ('it',
'PRP'), ('Robby', 'NNP'), ('live', 'VBP'), ('in', 'IN'), ('that', 'DT'), ('farm', 'NN'), ('Robby',
'NNP'), ('is', 'VBZ'), ('the', 'DT'), ('name', 'NN'), ('of', 'IN'), ('a', 'DT'), ('whit', 'NN'), ('rabbit',
'NN'), ('who', 'WP'), ('eats', 'NNS'), ('carrot', 'VBP'), ('All', 'DT'), ('bird', 'NN'), ('except',
'IN'), ('penguin', 'NN'), ('fly', 'RB'), ('Any', 'DT'), ('human', 'NN'), ('who', 'WP'), ('is', 'VBZ'),
('killed', 'VBN'), ('is', 'VBZ'), ('not', 'RB'), ('aliv', 'NN'), ('If', 'IN'), ('a', 'DT'), ('human', 'JJ'),
('is', 'VBZ'), ('not', 'RB'), ('aliv', 'NN'), ('he', 'PRP'), ('is', 'VBZ'), ('dead', 'JJ')]

```

Figure 4.1 9: The Example as Dependency Graphs.

After constructing a dependency graph is represented as a tree using programming python as following in Figure 4.20.

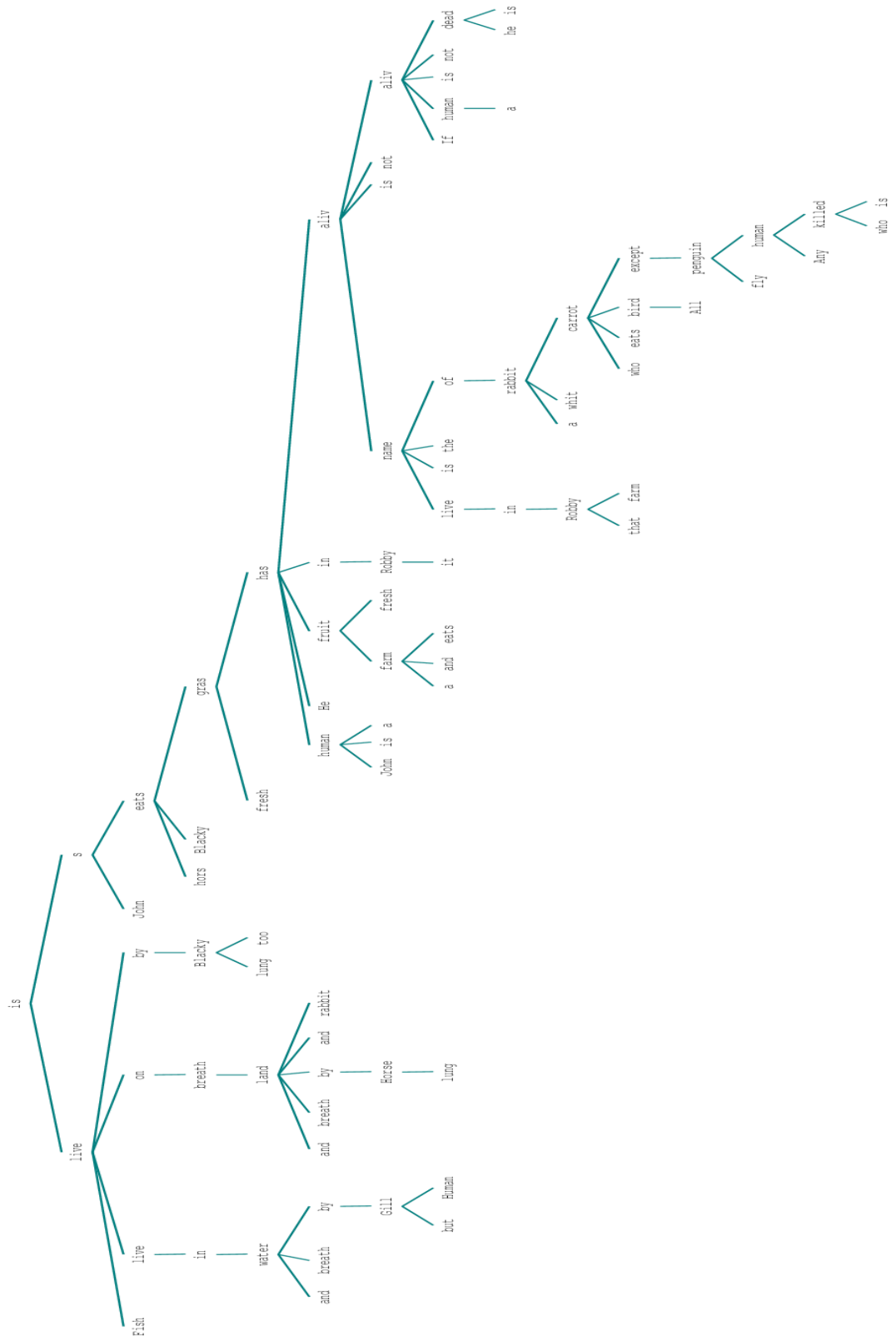


Figure 4.20: The Dependency Graph as a Tree

4.6 SUMMARY

In this chapter, the basic steps of the methodology of this research were discussed beginning with the pre-processing document which includes: splitting each document into sentences, stemming, tokenization and part of speech tagging. Also in this chapter, the process of parsing text to obtain sentence structure was discussed and English grammar rule was also presented. The final step of constructing the dependency graph was discussed and explained with emphasis on how the dependency graph solved the sparsity and semantic problem compared to ontology based text representation. In the next chapter, the detail explanation of results is presented.

CHAPTER FIVE

RESULTS AND DISCUSSION

5.1 INTRODUCTION

In the previous chapter, the construction of a dependency graph was discussed in detail. The manner by which the dependency graph solves sparsity and semantic problems was also investigated. The results of text clustering by TF-IDF, DG, and ontology obtained by using a confusion matrix are presented in Section 5.2. Section 5.3 provides an evaluation of the results of each method according to four evaluation measures, namely, precision, recall, F measure and accuracy. The results are analyzed in Section 5.4.

5.2 CLUSTERING RESULTS

The performance of the algorithms was evaluated based on four measures, namely, precision, recall, F measure, and accuracy. These measures are derived from the confusion matrix table during the performance analysis.

A confusion matrix (Kohavi and Provost, 1998) contains information on the actual and predicted clustering by a classification system. The performance of such systems is commonly evaluated using the data in the matrix.

A confusion matrix is a specific table that allows for the visualization of the performance of an algorithm, typically a supervised learning algorithm (a table for unsupervised learning algorithms is usually called a matching matrix).

Each column of the matrix represents the instances in a predicted class, whereas each row represents the instances in an actual class (Fawcett, 2006). The entries in the table have four possible values, namely, true positive (TP), true negative (TN), false positive (FP), and false negative (FN). In TP, the case is positive and predicted as positive. In TN, the case is negative and predicted as negative. The case in FP is negative, but predicted as positive. In FN, the case is positive, but predicted as negative (Sokolova, Japkowicz, & Szpakowicz, 2006). Figure 5.1 shows the confusion matrix for two-class clustering.

	Same Category	Different Categories
Same cluster	TP	FP
Different clusters	FN	TN

Figure 5.1: Confusion Matrix

The k-means algorithm was used to cluster the selected data set represented by TF-IDF and dependency graph. The number of k was set to 20 because the document contains 20 groups. The similarity measure used in this research was cosine similarity.

5.2.1 TF-IDF

In TF-IDF, the numerical statistics aim to reverse the importance of the word to the document in the collective or corpus and are mostly used as a weighting factor in information retrieval and text mining.

Increasing the value of TF-IDF in proportion to the number of times a word appears in the document (matched by word frequency in the corpus) thus helps control the reality that several words are more common than others.

In this study, selecting the raw text of 20 newsgroups, which selecting 200 documents (10 from each group). Tokenization was implemented, and all stop words were deleted after stemming a word to a root word. Then selected the weighing terms ($TF \times IDF$). A term's document frequency is the number of documents in which the term occurs in the entire collection, and inverse document frequency is a computation of the general significance of the term. This method was presented in Section 2.2.1.

Next, applied the k-means algorithm to cluster the raw text using special tools in a rapid miner. When selecting the algorithm to cluster the text, several characteristics related to this algorithm, such as the number of clusters and measurement of similarity, must also be selected. Cluster analysis works based on the similarity between data objects, that is, similar objects should be in the same cluster while dissimilar objects should be in different clusters.

Therefore, measuring similarity is essential for any clustering method. For vector-based data objects, we usually estimated the similarity of objects based on the distance between objects. A large distance value implies low similarity and vice versa. Distances in a multi-dimensional space are commonly calculated by using the cosine distance.

Set the number of clusters to 20 because there are 20 groups in this research. Figure 5.2 A and B illustrate the process of clustering text using the k-means algorithm in rapid miner.

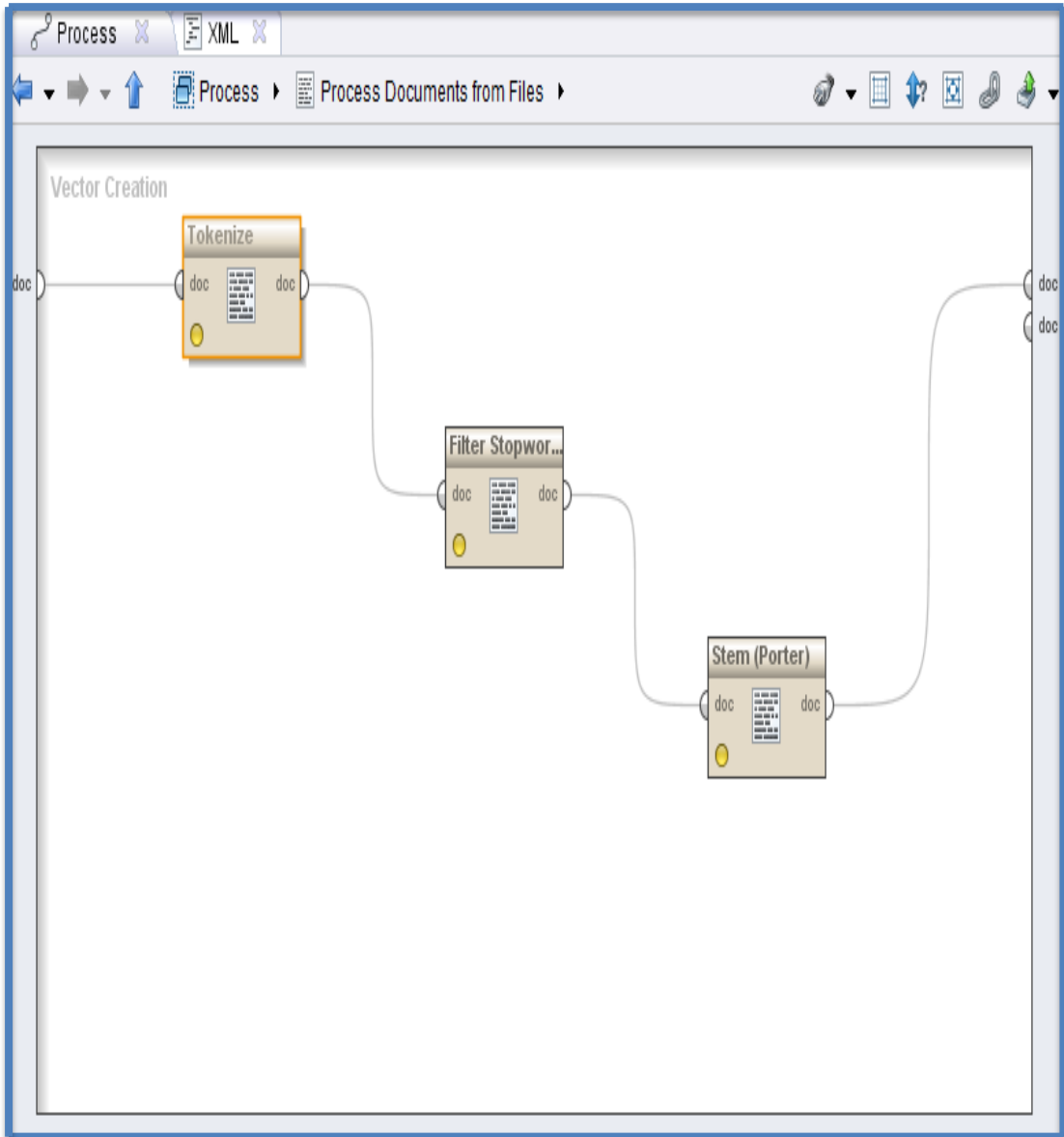


Figure 5.2, A: The Process: Tokenization. Stop word and stemming

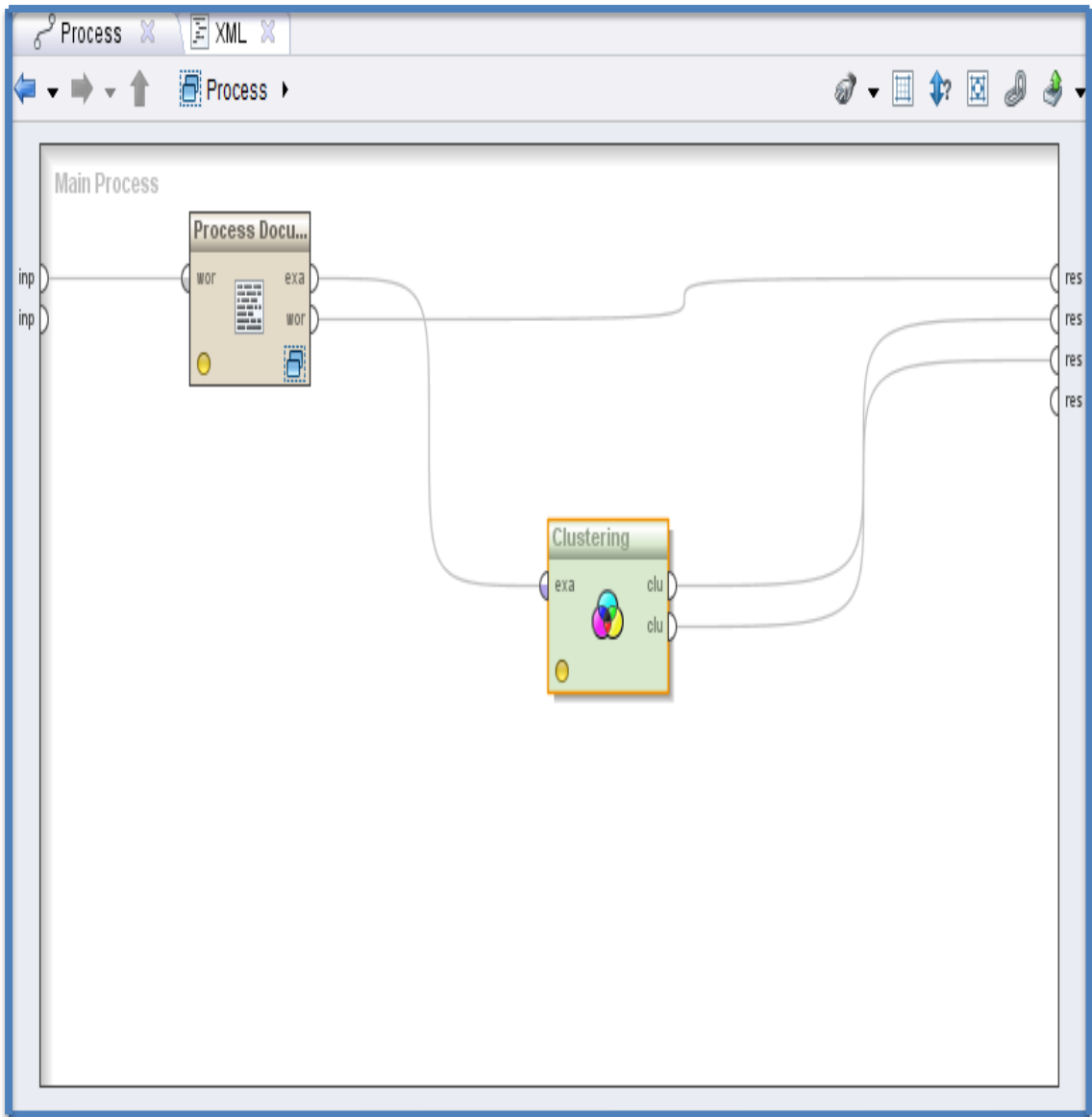


Figure 5.2, B: The Process of Clustering raw Text Using TF-IDF

Text was presented as TF-IDF. Out of the 200 documents, the k-means clustering model obtained 173 TPs and 27 FNs. Table 5.1 shows the values of TPs, FNs, FPs, and TNs.

Table 5.1

The Values of Confusion Matrix of TF-IDF

No.	Category	Documents	RESULT	(TP)	(FP)	(FN)	(TN)
1	<i>alt.atheism</i>	10	19	10	9	0	0
2	<i>comp.graphics</i>	10	6	6	0	4	0
3	<i>comp.os.ms.windows</i>	10	13	10	3	0	0
4	<i>comp.windows</i>	10	12	10	2	0	0
5	<i>misc.forsale</i>	10	9	9	0	1	0
6	<i>rec.autos</i>	10	7	7	0	3	0
7	<i>rec.motorcycles</i>	10	9	9	0	1	0
8	<i>rec.sport.baseball</i>	10	9	9	0	1	0
9	<i>sci.crypt</i>	10	7	7	0	3	0
10	<i>sci.med</i>	10	20	10	10	0	0
11	<i>sci.electronics</i>	10	12	10	2	0	0
12	<i>sci.space</i>	10	14	10	4	0	0
13	<i>talk.politics.guns</i>	10	4	4	0	6	0
14	<i>talk.religion.misc</i>	10	9	9	0	1	0
15	<i>rec.sport.hockey</i>	10	13	10	3	0	0
16	<i>comp.sys.mac.system</i>	10	12	10	2	0	0
17	<i>talk.politic.misc</i>	10	11	10	1	0	0
18	<i>talk.politics.mideast</i>	10	11	10	1	0	0
19	<i>comp.sys.ibm.pc.hardware</i>	10	6	6	0	4	0
20	<i>soc.religion.christian</i>	10	7	7	0	3	0

Table 5.1 shows that from the first group, namely, *alt.atheism*, 10 documents were entered. The result after clustering is 19 documents. We calculated the number of documents that belong to this group; this value represents the TPs.

Out of the 19 documents in the aforementioned table, only 10 were clustered correctly. Nine documents do not belong to this group; this condition means FPs. In the second group, i.e., comp.graphics, 10 documents were entered. The result after clustering is only six documents. After searching for documents that belong to this group, we found that all six documents belong to comp.graphics; this result means four documents were lost. This condition represents the value of FN.

5.2.2 DEPENDENCY GRAPH

Selected the file that includes 200 documents; each document was represented as a dependency graph. In this model, the documents represented as a dependency graph are directly clustered without implementing pre-processing steps because these processes were performed previously before generating the dependency graph. Next, text clustering is performed with the k-means algorithms and with cosine similarity as a similarity measure. Figure 5.3 illustrates the process of clustering by a dependency graph using the k-means algorithm in rapid miner.

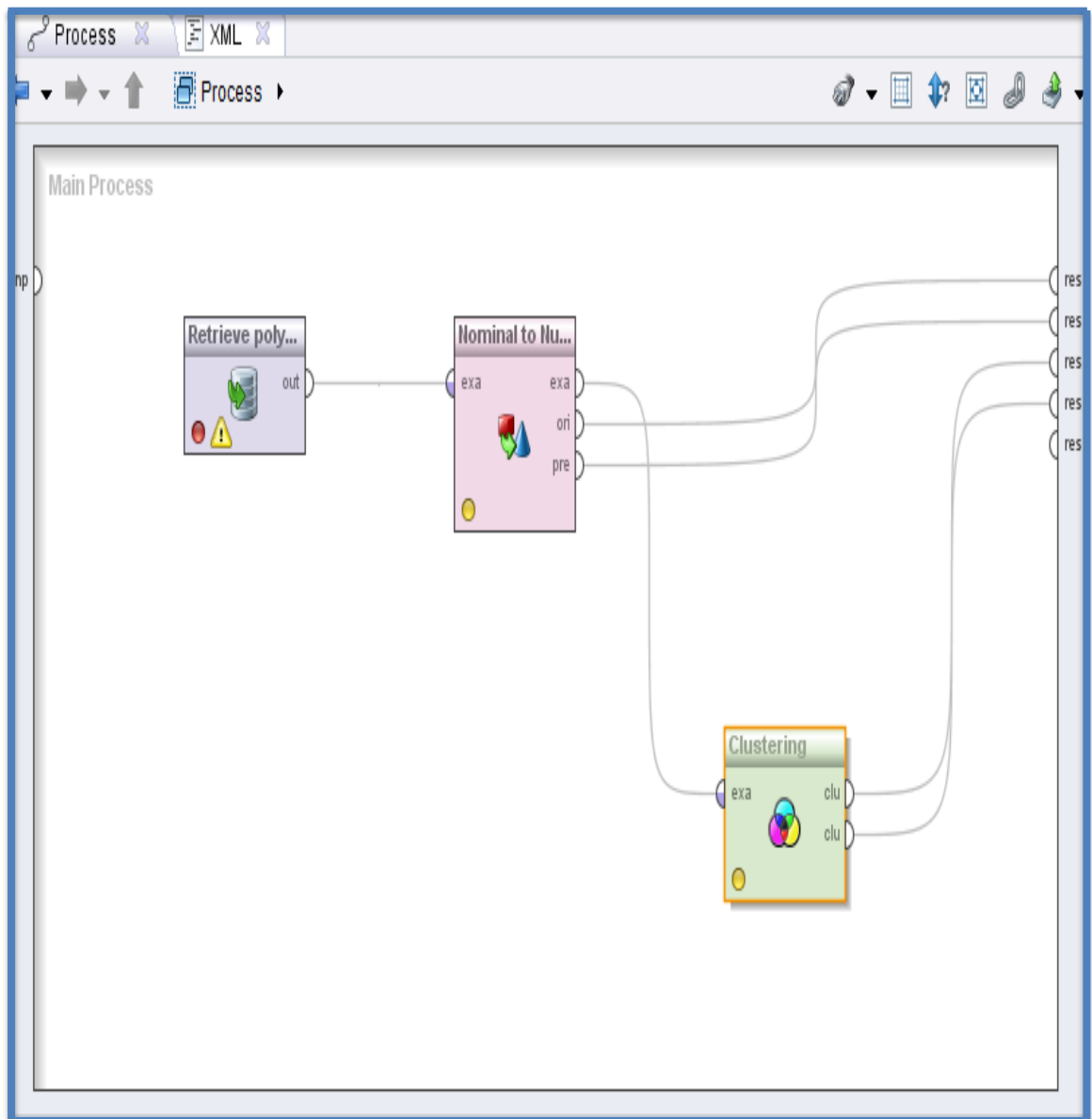


Figure 5.3: Process of Clustering by a Dependency Graph

When a document is inputted to the clustering model, the correct category label of that document should be predicted. The experimental results as average set of trials in six cases where the number of iterations = 5, 20, 35, 50, 80 and 100 shown in the table 5.2.

Table 5.2

Average of Experimental Results of the Dependency Graph

No	Category	No. input	Ite.1 =5	Ite.2 =20	Ite.3 =35	Ite.4 =50	Ite.5 =80	Ite.6 =100	Average
1	<i>alt.atheism</i>	10	13	11	11	11	11	11	11
2	<i>comp.graphics</i>	10	8	9	9	9	9	9	9
3	<i>comp.os.mswindows</i>	10	8	10	10	10	10	10	10
4	<i>comp.windows</i>	10	7	9	9	9	9	9	9
5	<i>misc.forsale</i>	10	6	6	6	6	6	6	6
6	<i>rec.autos</i>	10	9	9	9	9	9	9	9
7	<i>rec.motorcycles</i>	10	11	11	11	11	11	11	11
8	<i>rec.sport.baseball</i>	10	12	11	11	11	11	11	11
9	<i>sci.crypt</i>	10	10	10	10	10	10	10	10
10	<i>sci.med</i>	10	7	7	7	7	7	7	7
11	<i>sci.electronics</i>	10	10	9	9	9	9	9	9
12	<i>sci.space</i>	10	9	10	10	10	10	10	10
13	<i>talk.politics.guns</i>	10	12	12	12	12	12	12	12
14	<i>talk.religion.misc</i>	10	10	10	10	10	10	10	10
15	<i>rec.sport.hockey</i>	10	10	10	10	10	10	10	10
16	<i>comp.sys.mac.system</i>	10	9	9	9	9	9	9	9
17	<i>talk.politic.misc</i>	10	12	12	12	12	12	12	12
18	<i>talk.politics.mideast</i>	10	13	13	13	13	13	13	13
19	<i>comp.sys.ibm.pc.hardw</i>	10	12	10	10	10	10	10	10
20	<i>soc.religion.christian</i>	10	12	12	12	12	12	12	12

This table displays the experimental results when clustering text represented as a dependency graph using k-means algorithm in six cases and the average of these trials. TPs and FNs are very important because they are factors that can measure precision and recall.

Out of the 200 documents inputted to the k-means algorithms, 188 sentences were clustered correctly. The remaining 12 sentences were clustered incorrectly. Hence, the total number of TPs is 188, and the total number of FNs is 12. All TPs, FNs, TNs, and FPs for the individual categories are shown in Table 5.3.

Table 5.3

TP, FP, FN and TN of Dependency Graph

No	Category	Average of a set of trials	(TP)	(FP)	(FN)	(TN)
1	<i>alt.atheism</i>	11	10	1	0	0
2	<i>comp.graphics</i>	9	9	0	1	0
3	<i>comp.os.mswindows</i>	10	10	0	0	0
4	<i>comp.windows</i>	9	9	0	1	0
5	<i>misc.forsale</i>	6	6	0	4	0
6	<i>rec.autos</i>	9	9	0	1	0
7	<i>rec.motorcycles</i>	11	10	1	0	0
8	<i>rec.sport.baseball</i>	11	10	1	0	0
9	<i>sci.crypt</i>	10	10	0	0	0
10	<i>sci.med</i>	7	7	0	3	0
11	<i>sci.electronics</i>	9	9	0	1	0
12	<i>sci.space</i>	10	10	0	0	0
13	<i>talk.politics.guns</i>	12	10	2	0	0
14	<i>talk.religion.misc</i>	10	10	0	0	0
15	<i>rec.sport.hockey</i>	10	0	0	0	0
16	<i>comp.sys.mac.system</i>	9	9	0	1	0
17	<i>talk.politic.misc</i>	12	10	2	0	0
18	<i>talk.politics.mideast</i>	13	10	3	0	0
19	<i>comp.sys.ibm.pc.hardw</i>	10	10	0	0	0
20	<i>soc.religion.christian</i>	12	10	2	0	0

The table above shows the values of the confusion matrix, where the TP value of 10 means that all documents entered from the alt.atheism group were clustered correctly, except for one document that does not belong to this group (value of FP). The value of false negatives is zero because 10 documents were entered, and the output of 10 documents means no document was lost.

In the second row the result after inputting the actual 10 document are 9. In this case one document was not clustered correctly and this is considered false negative. The value of false positive equals zero that means all 9 documents belongs to the same group.

5.2.3 ONTOLOGY

An ontology-based approach allows the analyst to represent the complex structure of objects so as to execute knowledge on the hierarchical structure of categories, as well as reveal and utilize information on the relationships between individual objects and categories.

In this research, the results obtained from representing text as TF-IDF and those obtained when text was represented as a dependency graph (in previous studies) using ontologies to cluster text documents were compared. A total of 20 newsgroup data sets was employed, and k-means algorithms were applied in text clustering. Cosine similarity was employed to calculate the distance between documents (V. Sureka & S.C. Punitha, 2012).

5.3 RESULT EVALUATION

This evaluation task is known as cluster validity. Three types of validation criteria exist: external, internal, and relative criteria (Jain & Dubes, 1988; Theodoridis, Pikrakis, Koutroumbas, & Cavouras, 2010; Velmurugan & Santhanam, 2010).

An external criterion compares a clustering result with that of ground truth clustering. An internal criterion determines if a clustering result is intrinsically appropriate for an input data set. A relative criterion compares clustering results by measuring their relative quality based on a pre-specified criterion.

Overall, our dataset contains 200 documents consisting of 20 groups (the size of each group is 10). We relied on an internal criterion. Therefore, clustering methods were applied to this dataset, and the results were validated based on the known group's partitioning. The quality of the clustering results was evaluated by comparing the output clusters with the 20 given newsgroups.

In the case of text represented as TF-IDF, the results were evaluated with Equations 3.1, 3.2, 3.3, and 3.4, as discussed in detail in Chapter 3. Table 5.4 shows the result of the evaluation.

Table 5.4

Precision, Recall, F-measure and Accuracy of TFIDF representation

Method	Dataset	Precision	Recall	F-measure	Accuracy
TF-IDF	20Newsgroup	0.879	0.865	0.871	0.865

The table shows the precision, recall, F measure, and accuracy of representing text as TF-IDF. The value of precision is clearly higher than that of recall (precision is 0.879 and recall is 0.865). The value of F measure is equal to 0.871, and the value of accuracy is equal to the value of recall. Overall, the value of precision when text is represented as TF-IDF is larger than the values of recall, F measure, and accuracy.

In case of text represented as a dependency graph, after determining all the TPs, FPs, TNs, and FNs of each cluster, precision, recall, F measure, and accuracy values were calculated based on the formulae discussed earlier. Table 5.5 shows the result of clustering text as a dependency graph.

Table 5.5

Precision, Recall, F-measure and Accuracy of Dependency Graph representation

Method	Data set	Precision	Recall	F-measure	Accuracy
DG	20 News group	0.914	0.939	0.926	0.94

The table above provides information on the performance of representing text as a dependency graph. The values of precision, recall, F measure, and accuracy are shown. Clearly, the value of recall 0.939 when text is represented as a dependency graph exhibits an increase from that in the previous method (text represented as TF-IDF). The value of precision is less than that of recall in this method, whereas in the previous method, the value of precision is greater than that of recall.

This result means the performance of representing text as a dependency graph is better because the recall value is the proportion of positive cases that were correctly identified. The F measure value is equal to that of recall.

In this research was selected the same documents and method of measuring distance because we wanted to compare text represented as a dependency graph with raw text represented as TF-IDF and cluster them using the same algorithm (k-means) to determine which methods perform best in text representation. The produced results were also compared with those of a previous study using ontology as the text representation scheme for 20 newsgroup datasets (Punitha, 2012).

Punitha's study was selected for comparison because of the graph-based representation method he presented, namely, ontology. The same clustering algorithm was implemented (k-means). The similarity measure used in this work to calculate the similarity between documents was cosine similarity. Table 5.6 shows the result when an ontology is used as a text representation scheme.

Table 5.6

Precision, Recall, F-Measure and Accuracy of Ontology

Method	Data set	Precision	Recall	F-measure	Accuracy
ONTOLOGY	20newsgroup	0.613	0.726	0.665	92.43

The table shows that the value of recall when text is represented as an ontology is larger than that of precision. The value of the F measure is 0.665. The value of accuracy is the largest of the three values.

The values of precision, recall, F measure, and accuracy of each type used in text representation are presented in a bar graph to allow for the analysis and comparison of the results. Figures 5.4, 5.5, 5.6, and 5.7 present the precision, recall, F measure, and accuracy values of tf-idf, dependency graph, and ontology through a bar graph. In the bar graph, the x-axis represents the types of text representation (TF-IDF, DG, and ONTOLOGY). The Y-axis represents the precision, recall, F measure, and accuracy values of each type.

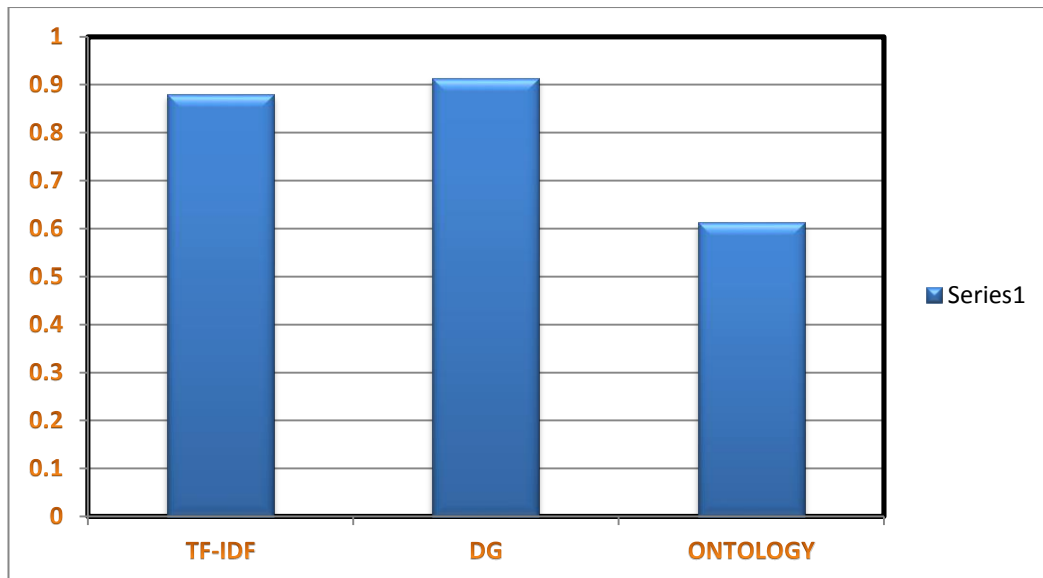


Figure 5.4: Precision Score of TF-IDF, DG and ONTOLOGY

The bar graph shows the precision values of the tf-idf, dependency graph, and ontology. The line graph shows that the value of precision when text is represented as a dependency graph is higher than the values obtained when text is represented as TF-IDF and ontology.

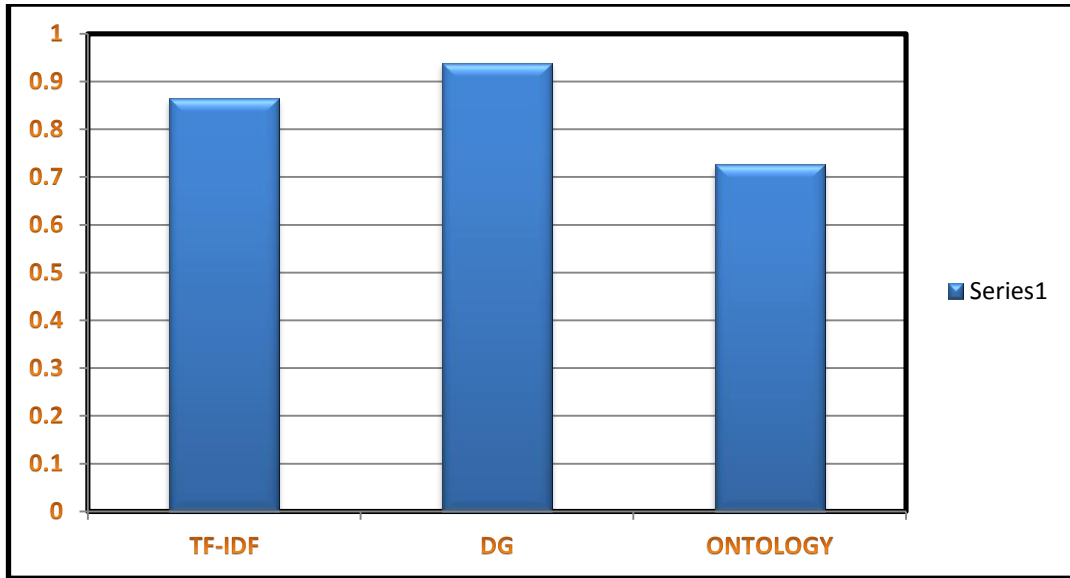


Figure 5.5: Recall Score of TF-IDF, DG and ONTOLOGY

The chart above provides information on the recall values of each text representation scheme. The percentage of recall increased dramatically from 0.726 when the text was represented as ontology to 0.865 when the text was represented as TF-IDF; the highest value was reached when text was represented as a dependency graph.

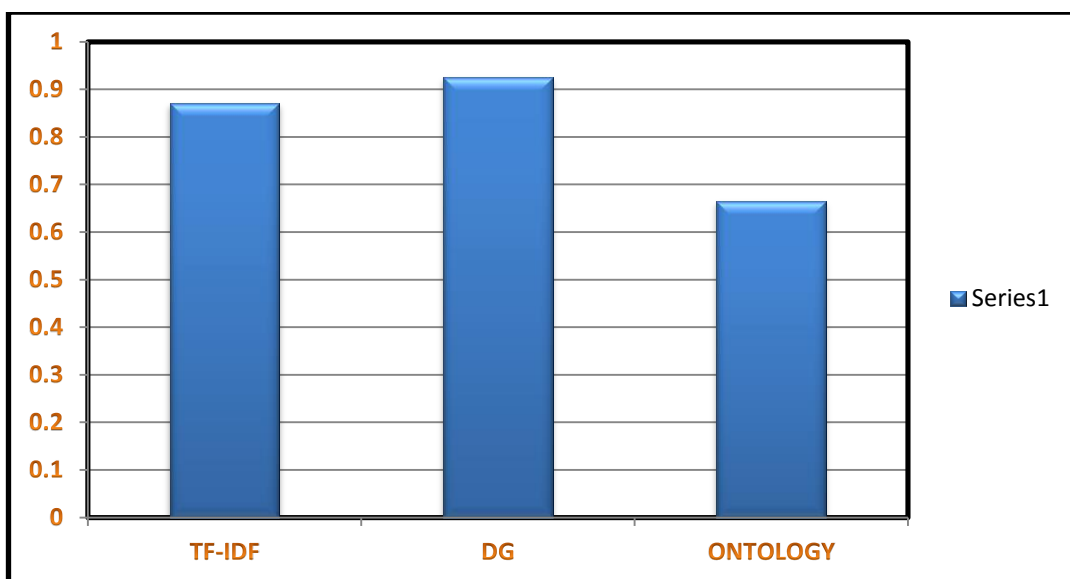


Figure 5.6: F-measure Score of TF-IDF, DG and ONTOLOGY

The bar graph above shows the F measure values of the TF-IDF, dependency graph, and ontology. When text is represented as ontology, the value of recall is smaller than the values of TF-IDF and dependency graph; the values are 0.665, 0.871, and 0.926 respectively.

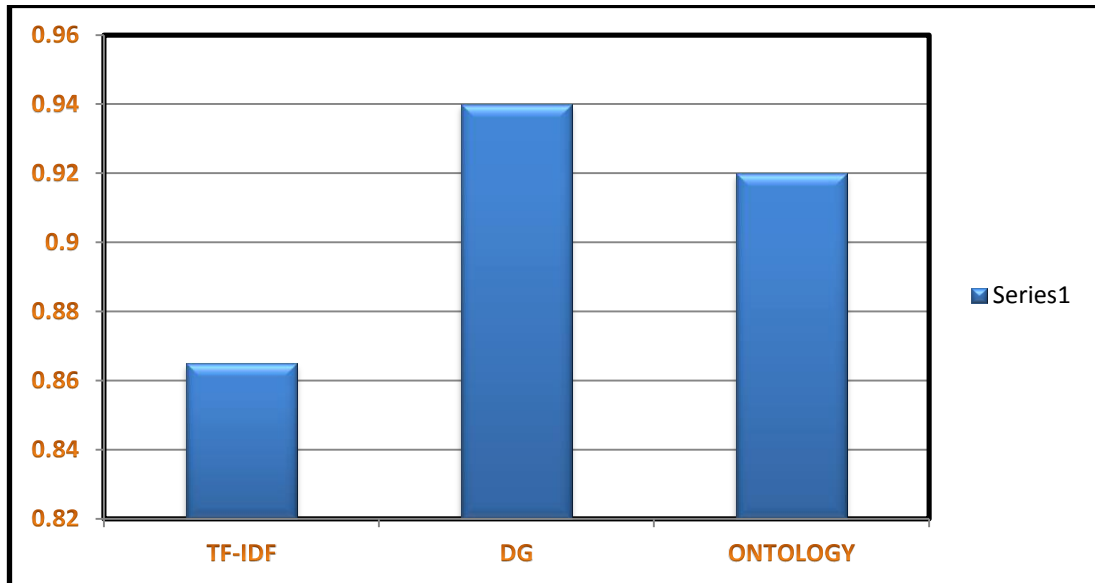


Figure 5.7: Accuracy Score of TF-IDF, DG, and ONTOLOGY

The line graph above shows the accuracy values of each type of text representation scheme. The value of accuracy when text is represented as TF-IDF (0.865) increases gradually and reaches 0.92 when the text is represented as ontology; the highest value is reached when text is represented as a dependency graph.

5.4 RESULT DISCUSSION

Overall, the most important measure of performance is recall, which is a measure of completeness and coverage. A high recall value is produced in specifying the number of sentences that have been clustered correctly.

The precision score is a measure of accuracy. The values of recall for the three methods, namely, TF-IDF, DG, and ontology, are 0.865, 0.939, and 0.726 respectively. The highest value is obtained for text representation as a dependency graph. This result indicates that the number of documents clustered correctly when text is represented as a dependency graph is more than the number of documents clustered incorrectly.

The F- measure combines the precision and recall scores and provides a clear-cut measurement. Text represented as a dependency graph obtained a higher F measure and accuracy value, which indicates better clustering performance. The results shown in Tables 6.3, 6.4, and 6.5 for each method of text represented as TF-IDF, ontology, and DG indicate that the F measure for dependency graph is 0.926; the value for TF-IDF and ontology are 0.871 and 0.665, respectively. Hence, the best result is obtained for dependency graph.

Its larger F- measure value indicates better clustering results (Steinbach et al., 2000). Lastly, the accuracy for DG is 0.94, whereas those for TF-IDF and ontology (previous study) are 0.865 and 0.665, respectively.

These results indicate that dependency graph is a better representation method compared with TF-IDF and ontology. Hence, the DG model can be utilized to reduce sparsity and semantic problems, obtain meaningful sentences, and reduce document features.

5.5 SUMMARY

In this chapter, raw text clustering was implemented by representing text as TF-IDF and as a dependency graph in a fast miner. The results of these types were then compared with those in a previous study, where the text was represented as ontologies. Performance was measured by calculating precision, recall, F measure, and accuracy, and the results from each method were recorded for comparison. The findings in this chapter show that excellent clustering results are achieved when text is represented as a dependency graph. The document analysis components that were carried out were able to determine the meaning of text structures in the documents. Clustering using a dependency graph achieved success and exceeded the performance of conventional means of text representation, such as TF-IDF and ontology. The conclusion and direction for future work are presented in the next chapter.

CHAPTER SIX

CONCLUSION AND FUTURE WORK

6.1 INTRODUCTION

This chapter provides a summary of the research and indicates its contributions and findings on the application of the dependency graph. The possible future extensions of this work are also discussed. The primary goal of this study is to reduce the sparsity and semantic problems related to traditional clustering methods, such as TF-IDF. Section 7.2 summarizes the contributions of the research in the area of document mining by generating a model called dependency graph based on semantic understanding of the text; text is then clustered as a dependency graph through clustering method. Section 7.3 presents a number of proposals for the development of this work and presents how new areas can be explored.

6.2 RESEARCH CONTRIBUTION

The major contribution of this study lies in the constructing of the dependency graph for each document to reduce the sparsity and semantic problem associated with TF-IDF by introducing the concept of semantic as a basis for text clustering. The emphasis is on the semantic representation, and the process of clustering documents for the purposes of evaluating the performance of the representation scheme. The findings of the study show that the dependency graph based text representation better captures the semantics of text compared to ontology based text representation.

The study also demonstrated the effectiveness of the dependency graph based representation method by producing a more accurate clustering result for the sample of 20 newsgroup dataset used in this study.

The DG representation approach to meaning and the steps to implement it were introduced. The approach is dependent on exploiting semantic structures to improve document clustering effectiveness. The representation involves translating readings of sentences into formulas that encapsulate their structures and semantics by using techniques such as parsing and semantic relation through dependency between words in sentences. The DG of the document is the accumulation of all the provisions of the analysis in the representation of its meaning.

In this work, a dependency graph representation model was constructed to represent meaning. The representation is based on semantic structures that translate readings of sentences into formulas that encapsulate their structures and semantics. It recognizes relationships between concepts and expresses their attributes.

Dependency graph representation allows for comparisons between modeled documents and results in a rich information system, which can be extended to analyze different production processes. Furthermore, it exhibits manipulation efficiency and can thus be used for other mining operations efficiently and effectively.

The DG generation process reduces the problem of sparsity and loss of semantic meanings between sentences by reducing the features of documents in addition to selecting sentences with meanings only; these are the principles required to reduce the sparsity problem. This method is effective in representing text when the result is clustered by the k-means algorithm compared with other methods of text representation.

6.3 FUTURE WORK

This work, in its different development stages, can be extended in many ways. First, for data representation, only 200 documents were selected. Therefore, in the future, the number of documents can be increased or all the 20 newsgroup datasets can be utilized. Second, other algorithms for clustering processes (aside from the k-means algorithm), can be applied; for example, DBSCAN, particle swarm optimization, and support vector machine, can be utilized. Third, clustering evaluation can be improved by developing the evaluation criteria, namely, precision, recall, F measure, and accuracy. By addressing these areas, the contributions of this research could be improved and refined.

6.4 SUMMARY

This chapter presented the contributions of this work, including the representation of text as a dependency graph. Several techniques that can increase the efficiency of the model, such as the use of other techniques to cluster text, were presented. The study also presented ways to enrich this type of text representation scheme for future studies. For example, the number of documents can be increased, and alternative methods of text clustering can be employed. The evaluation criteria, such as precision, recall, F-measure and accuracy can also be improve.

REFERENCES

- Aggarwal, C. C., & Zhai, C. (2012). A survey of text clustering algorithms Mining text data (pp. 77-128): Springer.
- Andrews, N. O., & Fox, E. A. (2007). Recent developments in document clustering. Computer Science, Virginia Tech, Tech Rep.
- Bloehdorn, S., Cimiano, P., Hotho, A., & Staab, S .(2005) .An Ontology-based Framework for Text Mining. Paper presented at the LDV Forum.
- Balmas, F. (2004). Displaying dependence graphs: a hierarchical approach. Journal of Software Maintenance and Evolution: Research and Practice, 16(3), 151-185.
- Balmas, Francoise (2001) Displaying dependence graphs: a hierarchical approach, [1] wcre, p. 261, Eighth Working Conference on Reverse Engineering (WCRE'01).
- Beck, F., & Diehl, S. (2013). On the impact of software evolution on software clustering. Empirical Software Engineering, 18(5), 970-1004.
- Bird, S., Klein, E., & Loper, E. (2009). Natural language processing with Python: O'Reilly Media, Inc.
- Brown, Peter F., Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. Class based n-gram models of natural language. Comput. Linguist., 18(4):467{479, 1992.
- Christopher, D. M., Prabhakar, R., & Hinrich, S. (2008). Introduction to information retrieval. An Introduction To Information Retrieval, 151-177.

- Chen, Y., & Tu, L. (2007). Density-based clustering for real-time stream data. Paper presented at the Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Cui, X., & Potok, T. E. (2005). Document clustering analysis based on hybrid PSO+ K-means algorithm. *Journal of Computer Sciences (special issue)*, 27, 33.
- Cimiaon,p.,Hotho,A.,& Staab,S.(2005).Learning Concept Hierarchies from text corpora Using Formal Concept Analysis. *J.Artif. Intell.Res(JAIR)*,24,305-339.
- Cavnar, W. B., & Trenkle, J. M. (1994). N-gram-based text categorization. *Ann Arbor MI*, 48113(2), 161-175.
- Chakravarthy, S., Venkatachalam, A., & Telang, A. (2010). (A graph-based approach for multi-folder email classification. Paper presented at the Data Mining (ICDM), 2010 IEEE 10th International Conference on.
- Chakrabarti, S. (2003). *Mining the Web: Discovering knowledge from hypertext data*: Morgan Kaufmann.
- Chen, G., Jaradat, S. A., Banerjee, N., Tanaka, T. S., Ko, M. S., & Zhang, M. Q. (2002). Evaluation and comparison of clustering algorithms in analyzing ES cell gene expression data. *Statistica Sinica*, 12(1), 241-262.
- Dolamic, L., & Savoy, J. (2008). Stemming approaches for East European languages *Advances in Multilingual and Multimodal Information Retrieval* (pp. 37-44): Springer.
- Dhillon, I. S., & Modha, D. S. (2001). Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1-2), 143-175.

- Dietrich, J., Yakovlev, V., McCartin, C., Jenson, G., & Duchrow, M. (2008). Cluster analysis of Java dependency graphs. Paper presented at the Proceedings of the 4th ACM symposium on Software visualization.
- Davidov, D., & Rappoport, A. (2008). Classification of Semantic Relationships between Nominals Using Pattern Clusters. Paper presented at the ACL.
- Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. Paper presented at the KDD.
- Eikvil, L. (1999). Information extraction from world wide web-a survey.
- Fawcett, T. (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8), 861-874.
- Fares, M., Oepen, S., & Zhang, Y. (2013). Machine learning for high-quality tokenization replicating variable tokenization schemes *Computational linguistics and intelligent text processing* (pp. 231-244): Springer.
- Gil-García, R., Badia-Contelles, J. M., & Pons-Porrata, A. (2006). A general framework for agglomerative hierarchical clustering algorithms. Paper presented at the Pattern Recognition, 2006. ICPR 2006. 18th International Conference on.
- Gardner, S. (2007). Ontology-based information management system and method: Google Patents.
- Gil-Garcia, R., & Pons-Porrata, A. (2010). Dynamic hierarchical algorithms for document clustering. *Pattern Recognition Letters*, 31(6), 469-477.
- Giller, G. L. (2012). The Statistical Properties of Random Bitstreams and the Sampling Distribution of Cosine Similarity. Available at SSRN 2167044.

- Han, X., Sun, L., & Zhao, J. (2011). Collective entity linking in web text: a graph-based method. Paper presented at the Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval.
- Huang, C., Simon, P., Hsieh, S., & Prevot, L. (2007). Rethinking Chinese Word Segmentation: Tokenization, Character Classification, or Word break Identification.
- Halkidi, M., Batistakis, Y., & Vazirgiannis, M. (2001). On clustering validation techniques. *Journal of Intelligent Information Systems*, 17(2-3), 107-145.
- Huang, X., & Wu, Q. (2013). Micro-blog commercial word extraction based on improved TF-IDF algorithm. Paper presented at the TENCON 2013-2013 IEEE Region 10 Conference (31194).
- Huang, J. Z., & Ng, M. (2006). Text clustering: algorithms, semantics and systems. history, 8, 3.
- Hammouda, K. M., & Kamel, M. S. (2004). Efficient phrase-based document indexing for web document clustering. *Knowledge and Data Engineering, IEEE Transactions on*, 16(10), 1279-1296.
- Hjørland, B. (2010). The foundation of the concept of relevance. *Journal of the American Society for Information Science and Technology*, 61(2), 217-237.
- Halkidi, M., & Vazirgiannis, M. (2008). A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, 29(6), 773-786.
- Harish, B., Guru, D., & Manjunath, S. (2010). Representation and classification of text documents: A brief review. *IJCA, Special Issue on RTIPPR (2)*, 110-119.
- Hofmann, T. (2001). Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42(1-2), 177-196.

- Hotho, A., Staab, S., & Stumme, G. (2003). Ontologies improve text document clustering. Paper presented at the Data Mining, 2003. ICDM 2003. Third IEEE International Conference on IEEE.
- Hu, J., Xiong, C., Shu, J., & Zhou, X. (2009). A novel text clustering method based on TGSOM and fuzzy K-means. Paper presented at the Education Technology and Computer Science, 2009. ETCS'09. First International Workshop on IEEE.
- Huang, J., Sun, H., Song, Q., Deng, H., & Han, J. (2013). Revealing Density-Based Clustering Structure from the Core-Connected Tree of a Network. *Knowledge and Data Engineering* (IEEE Transactions on, 25(8), 1876-1889.
- Hull, D. A. (1996). Stemming algorithms: A case study for detailed evaluation. *JASIS*, 47(1), 70-84.
- Jain, A. K. (2010). Data clustering: 50 years beyond K-means. *Pattern Recognition Letters*, 31(8), 651-666.
- Jing, L., Ng, M. K., Xu, J., & Huang, J. Z. (2005). Subspace clustering of text documents with feature weighting k-means algorithm *Advances in Knowledge Discovery and Data Mining* (pp. 802-812): Springer.
- Jing, L., Ng, M. K., & Huang, J. Z. (2007). An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data. *Knowledge and Data Engineering, IEEE Transactions on*, 19(8), 1026-1041.
- Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*: Prentice-Hall, Inc.
- Kaur, M., & Kaur, N. (2013). Web Document Clustering Approaches Using K-Means Algorithm. *International Journal of Advanced Research in Computer Science and Software Engineering*, (35).
- Kyle, K., Crossley, S., Dai, J., & McNamara, D. S. (2013). Native Language Identification: A Key N-gram Category Approach. *NAACL/HLT 2013*, 242.

- Karaa, A., & Ben, W. (2013). A NEW STEMMER TO IMPROVE INFORMATION RETRIEVAL. *International Journal of Network Security & Its Applications*, 5(4).
- Kisilevich, S., Mansmann, F., & Keim, D. (2010). P-DBSCAN: a density based clustering algorithm for exploration and analysis of attractive areas using collections of geo-tagged photos. Paper presented at the Proceedings of the 1st International Conference and Exhibition on Computing for Geospatial Research & Application.
- Klusch, M., Lodi, S., & Moro, G. (2003). Distributed clustering based on sampling local density estimates. Paper presented at the IJCAI.
- Liao, W.-k., Liu, Y., & Choudhary, A. (2004). A grid-based clustering algorithm using adaptive mesh refinement. Paper presented at the 7th Workshop on Mining Scientific and Engineering Datasets of SIAM International Conference on Data Mining.
- Liu, M., & Yang, J. (2012). An improvement of TFIDF weighting in text categorization. *International Proceedings of Computer Science & Information Technology*, 47.
- Luo, D., Ding, C., & Huang, H. (2010). Towards structural sparsity: An explicit l2/l0 approach. Paper presented at the Data Mining (ICDM), 2010 IEEE 10th International Conference on IEEE.
- Liu, Q., Deng, M., Shi Y., & Wang, J. (2012). A density-based spatial clustering algorithm considering both spatial proximity and attribute similarity. *Computers & Geosciences*, 46, 296-309.
- Mahdabi, P., & Crestani, F. The effect of citation analysis on query expansion for patent retrieval. *Information Retrieval*, 1-18.
- Matteucci, M. (2008). A tutorial on clustering algorithms. See at: <http://home.dei.polimi.it/matteucc/Clustering/tutorial/html/index.html>

- Mitchell, B. S., & Mancoridis, S. Clustering module dependency graphs of software systems using the bunch tool.
- Montes-y-Gómez, M., López-López, A., & Gelbukh, A. (2000). Information retrieval with conceptual graph matching. Paper presented at the Database and Expert Systems Applications.
- Moreira, A., Santos, M. Y., & Carneiro, S. (2005). Density-based clustering algorithms—DBSCAN and SNN. University of Minho—Portugal, Version 1.0, 25.07.
- Náther, P. (2005). N-gram based Text Categorization. Lomonosov Moscow State University.
- Ma, J., Xu, W., Sun, Y.-h., Turban, E., Wang, S., & Liu, O. (2012). An ontology-based text-mining method to cluster proposals for research project selection. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 42(3), 784-790.
- Pandit, S. (2008). On a robust document classification approach using TF-IDF scheme with learned, context-sensitive semantics.
- Parikh, M., & Varma, T. (2014). Survey on Different Grid Based Clustering Algorithms. *International Journal of Advanced Science and Technology*, 2(2) 4
- Parimala, M., Lopez, D., & Senthilkumar, N. (2011). A survey on density based clustering algorithms for mining large spatial databases. *International Journal of Advanced Science and Technology*, 31(1).
- Pons-Porrata, A., Berlanga-Llavori, R., & Ruiz-Shulcloper, J. (2007). Topic discovery based on text mining techniques. *Information processing & management*, 43(3), 752-768.
- punitha, v. s. s. c. (2012). Approaches to Ontology Based Algorithms for Clustering Text Documents. *Int.J.Computer Technology&Applications*, 3 (5), 1813-1817.

- Pandit, S. (2008). On a robust document classification approach using TF-IDF scheme with learned, context-sensitive semantics.
- Priss, U. (2006) "Formal concept analysis in information science." *ARIST*, 40.1: 521-543.
- Patel, C., Hamou-Lhadj, A., & Rilling, J. (2009). Software clustering using dynamic analysis and static dependencies. Paper presented at the Software Maintenance and Reengineering, 2009. CSMR'09. 13th European Conference on IEEE.
- Popat, K. (2013). Word Clustering for Data Sparsity: A Literature Survey.
- Paterlini, S., & Krink, T. (2006). Differential evolution and particle swarm optimisation in partitional clustering. *Computational Statistics & Data Analysis*, 50(5), 1220-1247. Text Documents. *Int.J.Computer Technology & Applications*, 3 (5), 1813-1817.
- Qu, Q., Qiu, J., Sun, C., & Wang, Y. (2008). Graph-based knowledge representation model and pattern retrieval. Paper presented at the Fuzzy Systems and Knowledge Discovery, 2008. FSKD'08. Fifth International Conference on IEEE.
- Qadi, A. E., Aboutajedine, D., & Ennouary, Y. (2010). Formal concept analysis for information retrieval. arXiv preprint arXiv:1003.1494.
- Rajaraman, K., & Tan, A.-H. (2002). Knowledge discovery from texts: a concept frame graph approach. Paper presented at the Proceedings of the eleventh international conference on Information and knowledge management.
- Rajaraman, K., & Tan, A.-H. (2003). Mining semantic networks for knowledge discovery. Paper presented at the Data Mining, 2003. ICDM 2003. Third IEEE International Conference on IEEE.
- Ramos, J. (2003). Using tf-idf to determine word relevance in document queries. Paper presented at the Proceedings of the First Instructional Conference on Machine Learning.

- Roma, V., Bewoor, M., & Patil, S. (2013). Evaluator and Comparator: Document Summary Generation based on Quantitative and Qualitative Metrics for International Journal of Scientific & Engineering Research.
- Rosario, B., & Hearst, M. A. (2004). Classifying semantic relations in bioscience texts. Paper presented at the Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics.
- Ricci, F., Rokach, L., & Shapira, B. (2011). Introduction to recommender systems handbook: Springer.
- Siti S.K.(2011).Frame work for deviation detection in text:Thesis,Universiti Kebangsaan Malaysia,Bangi.
- Shiga, M., & Mamitsuka, H. (2012). A variational bayesian framework for clustering with multiple graphs. Knowledge and Data Engineering, IEEE Transactions on, 24(4), 577-590.
- Salton, G., & McGill, M. J. (1983). Introduction to modern information retrieval.
- Salton, G., Singhal, A., Mitra, M., & Buckley, C. (1997). Automatic text structuring and summarization. Information processing & management, 33(2), 193-207.
- Schedl, M. (2012). # nowplaying Madonna: a large-scale evaluation on estimating similarities between music artists and between movies from microblogs. Information Retrieval, 15(3-4), 183-217.
- Shaban, K., Basir, O., & Kamel, M. (2006). Document mining based on semantic understanding of text Progress in Pattern Recognition, Image Analysis and Applications (pp. 834-843): Springer.

- Suchanek, F. M., Ifrim, G., & Weikum, G. (2006). Combining linguistic and statistical analysis to extract relations from web documents. Paper presented at the Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining.
- Saad, F. H., de la Iglesia, B., & Bell, D. G. (2006). A Comparison of Two Document Clustering Approaches for Clustering Medical Documents. Paper presented at the DMIN.
- Sowa, J. F., & Way, E. C. (1986). Implementing a semantic interpreter using conceptual graphs. *IBM Journal of Research and Development*, 30(1), 57-69.
- Stumme, G. (2002). Formal concept analysis on its way from mathematics to computer science *Conceptual Structures: Integration and Interfaces* (pp. 2-19): Springer.
- Slonim, N., & Tishby, N. (2000). Document clustering using word clusters via the information bottleneck method. Paper presented at the Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval.
- Sowa, J.F. ,(1976) "Conceptual Graphs for a Database Interface", *IBM J. R&D*, pp 336-357 vol.20.
- Shaban, K. (2006). A semantic graph model for text representation and matching in document mining. Citeseer.
- Schenker, A., Last, M., Bunke, H., & Kandel, A. (2003). Classification of web documents using a graph model. Paper presented at the Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on IEEE.
- Sokolova, M., Japkowicz, N., & Szpakowicz, S. (2006). Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation *AI 2006: Advances in Artificial Intelligence* (pp. 1015-1021): Springer.

- Tackstrom, O., Ryan McDonald, and Jakob Uszkoreit (2012). Cross-lingual word clusters for direct transfer of linguistic structure. In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 477-487, Montreal, Canada.
- Tasdemir, K., & Merényi, E. (2011). A validity index for prototype-based clustering of data sets with complex cluster structures. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(4), 1039-1053.
- Tar, H. H., & Nyunt, T. T. S. (2011). Ontology-Based Concept Weighting for Text Documents. Paper presented at the International Conference on Information Communication and Management IPCSIT.
- Tarabalka, Y., Benediktsson, J. A., & Chanussot, J. (2009). Spectral-spatial classification of hyperspectral imagery based on partitional clustering techniques. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(8), 2973-2987.
- Theodoridis, S., & Koutroubas, K. (1999). Feature generation II. *Pattern Recognition*, 233-270.
- Theodoridis, S., Pikrakis, A., Koutroubas, K., & Cavouras, D. (2010). *Introduction to Pattern Recognition: A Matlab Approach: A Matlab Approach: Academic Press.*
- Tong, T. (2011). *Semantic frameworks for document and ontology clustering. University of Missouri--Kansas City.*
- Uszkoreit, J., & Thorsten Brants. Distributed word clustering for large scale class-based language modeling in machine translation. In Proceedings of ACL-08: HLT, Columbus, Ohio, 2008.
- Velmurugan, T., & Santhanam, T. (2010). Computational complexity between k-means and k-medoids clustering algorithms for normal and uniform distributions of data points. *Journal of computer science*, 6(3), 363.

- Wang, Y., Ni, X., Sun, J.-T., Tong, Y., & Chen, Z. (2011). Representing document as dependency graph for document clustering. Paper presented at the Proceedings of the 20th ACM international conference on Information and knowledge management.
- Wang, L., & Liu, X. (2008). A new model of evaluating concept similarity. *Knowledge-Based Systems*, 21(8), 842-846.
- Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3), 645-678.
- Yun-tao, Z., Ling, G., & Yong-cheng, W. (2005). An improved TF-IDF approach for text classification. *Journal of Zhejiang University SCIENCE A*, 6(1), 49-55.
- Zhao, Y., & Karypis, G. (2002). Evaluation of hierarchical clustering algorithms for document datasets. Paper presented at the Proceedings of the eleventh international conference on Information and knowledge management.
- Zimmermann, T., & Nagappan, N. (2008). Predicting defects using network analysis on dependency graphs. Paper presented at the Proceedings of the 30th international conference on Software engineering.