TEXT DOCUMENT SIMILARITIES BASED ON WIKIPEDIA CONCEPT RELATEDNESS

by

Xiangru Wang

Submitted in partial fulfillment of the
requirements for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
July 2015

Thanks for my parents, my supervisors, and my friends.

Table of Contents

List of Tables

List of Figures

Abstract

Traditionally, text document similarity is based on lexical overlap between documents. Documents are represented based on bag of words (BOW), which ignores the relatedness among terms. One existing method to address this problem is to use external resources to enhance the BOW representation. Documents are represented by the background knowledge derived from external resources to create bag of concepts (BOC). Then BOC is used along with or instead of BOW to make a new representation. However, this approach assumes concepts to be independent, which is known as the orthogonality assumption.

This work focuses on developing new semantic similarity measures. By employing Wikipedia as the knowledge resource to create a BOC model, we get document similarities by following different concept mapping procedures combined with concept relatedness. We evaluate proposed measures in text clustering. Experimental results show that our BOC based similarity method can improve clustering performance.

List of Abbreviations Used

BOW	Bag of Words
BOC	Bag of Concepts
cf-idf	Concept Frequency-Inverse Document Frequency
GVSM	Generalized Vector Space Model
LCM	Longest Common Subsequence
LDA	Latent Dirichlet Allocation
M3	Multiple to Multiple Mapping
MOM	Multiple to One Mapping
NMI	Normalized Mutual Information
OOM	One to One Mapping
SCM	Single Concept Mapping
SMM	Single Median Mapping
tf-idf	Term Frequency-Inverse Document Frequency
VSM	Vector Space Model
WLM	Wikipedia Link-based Measure
WSD	Word Sense Disambiguation

Acknowledgements

This thesis would not have been possible without the guidance and help of several people. Foremost, I would like to express my gratitude to my adviser Dr. Seyednaser Nourashrafeddin for his patience, guidance, encouragement and advice provided through my thesis. I could not imagine a better adviser and mentor. I would also like to thank my supervisor Dr. Evangelos Milios for his valuable support and advice, and for this great platform he offered for research.

Chapter 1

Introduction

Text document similarity estimation is an important component in many tasks, including document classification, document clustering, information retrieval, and natural language processing.

An appropriate document representation is the basic step leading to accurate semantic similarity measures. Traditional methods usually treat the text corpus as a "bag of words" (BOW) model [1]. In this model, documents are represented using Vector Space Model (*VSM*). The term vectors are assumed to be not related with each other. In this way, the BOW model only covers the lexical information of a document without considering the relatedness among terms [12]. Two documents can be placed into different clusters if they express the same topics but using different terms.

One common approach to address this problem is to extract the topics mentioned in the documents from an external knowledge resource. Previous research work have employed external ontologies such as WordNet [10, 9] and Mesh [36, 37]. However, they all suffer from limited domain coverage. In this paper, we employ Wikipedia as the external knowledge-based resource. Wikipedia as a multilingual, on-line and content free encyclopedia is the result of collaborative engagement of millions of people around the world. According to the statistics in 2015, the English version of Wikipedia has more than 25 millions of registered users to optimize and supplement the contents within it. Wikipedia covers a very large number of named entities, domain specific terms, and new entities [24]. The latest report reveal that the average increase for the English Wikipedia from January to July in 2015 is 1234 new articles per day[1]. Hyperlinks and other relations in Wikipedia are an extraordinary resource to exploit [2]. Wikipedia is claimed to be less noisy when used as knowledge-base thesaurus comparing to WordNet and Open Directory in [7].

[1] `https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia`

It is also quite important to develop an effective way to enrich document representation by incorporating semantic information. The easiest method is to consider a document in "bag of concepts" (BOC) and measure similarity of two documents as the overlap of their concept sets, which is similar to the BOW model. However, this method assumes that the concepts are independent of each other. This assumption is also not accurate in the BOC model since terms like Wikipedia concepts might be related as well.

To relax the orthogonality assumption, this work aims at developing new text document similarity methods enhanced by semantic information extracted from an external knowledge resource.

In this work, we use the English version of Wikipedia as the external resource to represent text documents. Each article in Wikipedia is assumed to capture one topic or concept. The title of each article, which is just a word or a phrase, is referred to as a Wikipedia concept or just concept in the rest of this work. A document is represented by a vector of Wikipedia concepts in the BOC model. Then we present five text document similarity methods based on the BOC model. For a pair a documents, we match concepts of one document to concepts of the other one in different ways. Based on the mapping concept-pairs, we calculate similarity for each pair of documents.

We use document clustering to evaluate the similarity methods. Three clustering algorithms are applied on six datasets. The experimental results show that the BOC model with concept relatedness can enhance the clustering performance significantly without much additional cost. The main contribution of this work is that we propose a text document similarity measure based on semantic information extracted from Wikipedia, which relaxes the orthogonality assumption.

Chapter 2

Related Work

In this chapter, we review previous research in related fields. To the best of our knowledge, there is a limited research on text document similarity at semantic level.

First, we review some existing methods for calculating the text semantic similarity. We summarize the advantages and disadvantages to get a better understanding of the semantic similarity analysis. After that, we present some research related to exploiting semantic information in Wikipedia. Then we introduce some existing Wikification methods and compare them. Last, we introduce the orthogonality assumption in the standard Vector Space Model (VSM).

2.1 Text Semantic Similarity

The text semantic similarity problem is defined as how to derive the similarity score at the semantic level automatically given two text documents as input. The traditional measures just consider the surface form overlap, which is known as lexical similarity between two documents. To fully leverage the information embedded in the document collection, we cannot ignore the semantic information. There have been a lot of word-to-word semantic similarity measures such as a knowledge-based approach was used in [35] and a corpus-based approach was applied to calculate the relatedness among words in [32]. However, not much work has been done to get the text semantic similarity. A measure which depended on both corpus-based and knowledge-based measures of similarity was developed in [21]. This method was applied on short text and the experiments showed that incorporating semantic information into the similarity measure outperformed the simple lexical matching methods. Another measure which modified Longest Common Subsequence (LCS) was presented for text semantic similarity in [17]. This method is actually a corpus-based method. Their method determined the similarity of two texts in terms of both lexical and semantic levels. We investigated the document similarity mainly at a semantic level. By incorporating the

semantic similarity among Wikipedia concepts, we propose five semantic similarity measures.

2.2 Semantic Information in Wikipedia

There have been some research work on employing semantic information in Wikipedia for document clustering. The question of what kind of information in Wikipedia can provide more benefit to the clustering performance was explored in [12]. They used several vector combinations to represent the document collection including: word vector only, concept vector only, category vector only, word and concept vectors, word and category vectors, concept and category vectors, word and concept and category vectors. Finally, their experiments on three datasets showed that category information is more useful than others in document clustering. However, this paper just explored the lexical overlap at a semantic level without considering the relatedness between terms or concepts. Moreover, category information is a higher level information. It is more difficult to measure the relatedness among categories. In our work, we use the concept vectors and concept relatedness.

The semantic information was captured by representing documents with Wikipedia concepts in the BOC model in [14]. Then during the clustering process, they also utilized Wikipedia to facilitate active learning by measuring the semantic relatedness among concepts to analyze the topic distribution within document groups. Their experimental results showed that their approach was effective and comparable to previous work.

A new framework was proposed for partitional clustering by integrating Wikipedia concepts into the bag of words model in [26]. By combining clusters from both BOW and BOC, the documents with the same label in the clusterings were used as a training set to learn a classifier which was used to cluster the remaining documents. Their experiments revealed that the BOC model did help if combined with the BOW model, but could not outperform the BOW model by itself.

These works all showed that Wikipedia is a good resource and can improve the clustering quality to some extent. However, they just pointed out the noisy information problem resulting from simply enriching or replacing original document contents with Wikipedia concepts without addressing that problem. In this work, we explore

Wikipedia information and address the noisy information problem.

2.3 Wikification Methods

Given a text document, the wikification task is defined as identifying the most related Wikipedia concepts associated with the text and linking them to Wikipedia articles [22]. There are two traditional problems in this task: key term extraction and link disambiguation. Some research has been devoted to make the wikification more accurate. By comparing the overlap between a text document and Wikipedia text articles, a list of weighted Wikipedia article titles were extracted in [7]. Another work was proposed to match the text to Wikipedia candidate concepts by constructing a Wikipedia concept candidates text vocabulary and utilizing N-gram method in [7]. They also used a machine learning method to do the sense disambiguation. A dictionary was firstly built in [12], within which each entry includes preferred Wikipedia concepts and redirected concepts. Then two methods are proposed for concept matching. In this paper, we employ an open toolkit [23] for wikification task and for obtaining the concept relatedness to sidestep the laborious effort needed to mine Wikipedia's riches.

2.4 Orthogonality Assumption

In the BOW model, we define the orthogonality assumption as assuming that there is no relatedness among terms. This assumption also exists in the BOC model. To relax this assumption, a method was proposed to measure the pair-wise document similarities by enriching each document with the concepts that have been identified in the other document in [15]. This method did not take the connections among concepts into consideration and still assumed that concepts were mutually perpendicular.

The problem of measuring the term to term relatedness with the use of WordNet was exploited in [31]. They incorporated the semantic information to enrich the document representation as the Generalized Vector Space Model ($GVSM$). Their experiment results revealed that by settling the orthogonal assumption, their measure could improve the text retrieval performance. However, they only utilized WordNet which suffers from Word Sense Disambiguation (WSD) problem [29] in the semantic

relatedness calculation among terms. Though their experimental results revealed that embedding semantic information could improve text retrieval, they still need other semantic network based models to confirm their conclusion. In this work, we investigate Wikipedia and integrate Wikipedia concept relatedness into the BOC model to address the problem resulting from orthogonality assumption.

Chapter 3

Proposed Semantic Similarities

In this chapter, we propose five new semantic similarity measures between two text documents, which are summarized in Table 3.1.

We represent the data in the form of BOC model. Each document is represented as a vector of Wikipedia concepts. We use "similarity" as the terminology used for documents and "relatedness"as the terminology used for concepts in this work.

Table 3.1: Summary of proposed semantic similarity measures

Name of Measure	Procedure of Measure	Time Complexity
Single Concept Mapping (SCM)	Matrix Creation Single Concept Mapping Similarity Calculation	$O(m^2n^2)$
Single Median Mapping (SMM)	Median Searching Single Median Mapping Similarity Calculation	$O(m^2n^2)$
One to One Mapping (OOM)	Matrix Creation Once to One Mapping Similarity Calculation	$O(m^2n^2)$
Multiple to One Mapping (MOM)	Matrix Creation Multiple to One Mapping Similarity Calculation	$O(m^2n^2)$
Multiple to Multiple Mapping ($M3$)	Matrix Creation Multiple to Multiple Mapping Similarity Calculation	$O(m^2n^2)$

where m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus.

3.1 Single Concept Mapping (SCM) Measure

In this part, we explore one simple semantic similarity measure, which searches for the concept pair with the greatest relatedness between two documents to get the document similarity. In this method, each document is represented as a Wikipedia

concept vector. We use *concept frequency-inverse document frequency* (*cf-idf*) [8] as the feature value to measure the concept weight.

Firstly, we create a *concept-concept* matrix for a document pair. By searching for the maximum value in this matrix, we select one concept from each document to make a mapping between two concept sets. This measure is defined as below:

Definition 3.1. Single Concept Mapping (*SCM*): Given two documents and their associate concepts, a single concept is selected from each document and the selected concepts are mapped to get the document similarity.

The main steps of this measure are described below and shown in Fig. 3.1. In the following part, document a is d_a, document b is d_b, the ith concept of d_a is c_{ai}, the jth concept of d_b is c_{bj}, w_{ai} is the *cf-idf* value of c_{ai} of d_a, and w_{bj} is the *cf-idf* value of c_{bj} of d_b.

1. **Matrix Creation:**
   Given a pair of documents d_a and d_b, we assign concepts of d_a to rows and concepts of d_b to columns. In this way, an $I * J$ matrix M is formed, where I is the number of concepts in d_a and J is the number of concepts in d_b. The entry (i, j) of this matrix is represented as $rel(i, j)$, which is the relatedness of c_{ai} to c_{bj}.

2. **Single Concept Mapping:**
   We search for the maximum value $rel(i, j)_{max}$ in M. If there is more than one maximum value in M, we choose the concept pair with the maximum weight product $w_{ai} \cdot w_{bj}$.

3. **Similarity Calculation:**

$$sim(d_a, d_b) = w_{ai} \cdot w_{bj} \cdot rel(i, j)_{max} \tag{3.1}$$

This method utilizes both concept weights and concept relatedness. Given concept vectors of documents, the time complexity of this method to calculate the pair-wise similarity among documents is $O(m^2n^2)$, in which m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus. The main steps of this method are described in Algorithm 1.

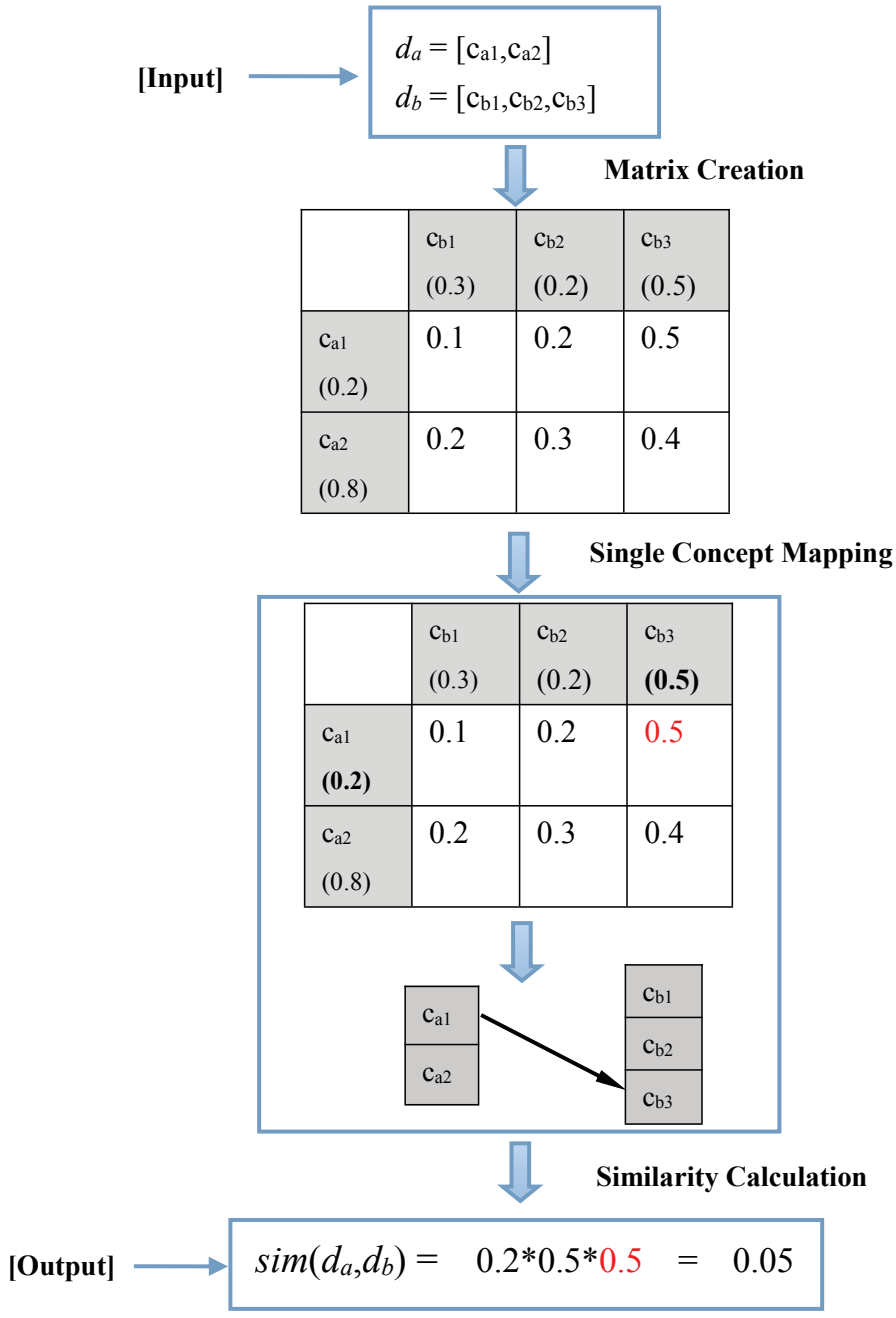


Figure 3.1: Document similarity measure based on *SCM*. The measure includes three steps: Matrix Creation, Single Mapping and Similarity Calculation. By finding the maximum conceptual relatedness in the matrix, we map a single concept of d_a to a single concept of d_b.

Algorithm 1 Semantic similarity based on *SCM*

Input: A *document-concept* matrix

Output: A *document-document* similarity matrix

1: **for** each pair of document vectors (d_a and d_b) from the input **do**
2: Form an $I * J$ *concept-concept* matrix M, while rows correspond to concepts of d_a and columns correspond to concepts of d_b. Each concept has its *cf-idf* weight. Fill in M with $rel(i, j)$.
3: Search for the $rel(i, j)_{max}$ in M
4: **if** There are more than one maximum relatedness **then**
5: Choose the concept pair with the greatest $w_{ai} \cdot w_{bj}$ production
6: **end if**
7: Use Eq. 3.1 to get $sim(d_a, d_b)$
8: **end for**
9: **return** A *document-document* similarity matrix

3.2 Single Median Mapping (*SMM*) Measure

In this part, we explore another semantic similarity measure which employs the idea of *median linkage* method in agglomerative clustering. The document representation is the same as in *SCM* measure.

For each document, we first find its median concept. For a concept set, a median concept is defined as the concept with the maximum average relatedness to other concepts within the same concept set. We use median concept or just median for short in this work. The median of a concept set can be treated as a representative of this set. We map the median of one document to the median of another document to measure the document similarity.

Definition 3.2. Single Median Mapping (*SMM*): Given two documents and their associate concepts, one single concept is selected from each document as the median concept and the two median concepts are mapped to get the document similarity.

The main steps of this measure are described as below and shown in Fig. 3.2. In the following part, document a is d_a, document b is d_b, the ith concept of d_a is c_{ai},

the jth concept of d_b is c_{bj}, c_{am} is the median concept of d_a, c_{bm} is the median concept of d_b, w_{ai} is the *cf-idf* value of c_{ai}, and w_{bj} is the *cf-idf* value of c_{bj}.

1. **Median Searching:**
   For each concept of a concept set, we find its relatedness to the rest concepts of this set and get the average for those relatedness. Concept with the greatest average relatedness is the median for this concept set. If there is more than one median of a concept set, we choose the median with the maximum *cf-idf* weight w.

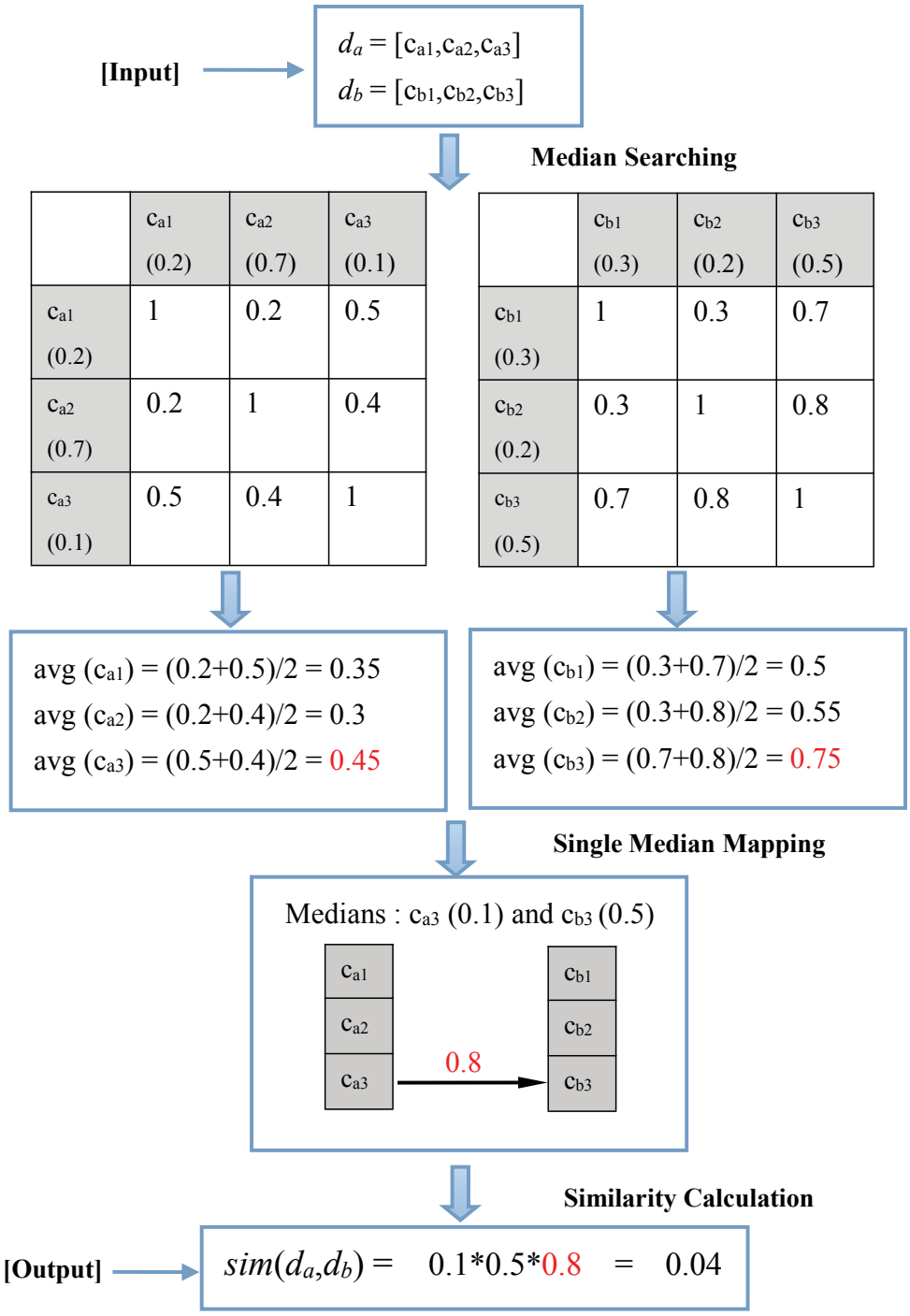
2. **Single Median Mapping:**
   For each pair of documents d_a and d_b, we map c_{am} to c_{bm} and get the relatedness rel between them. At the same time, we keep the *cf-idf* for c_{am} as w_{am} and *cf-idf* for c_{bm} as w_{bm}.

3. **Similarity Calculation:**

$$sim(d_a, d_b) = w_{am} \cdot w_{bm} \cdot rel \tag{3.2}$$

   where rel is the concept relatedness between c_{am} and c_{bm}.

This method also utilizes both concept weights and concept relatedness. Given concept vectors of documents, the time complexity of this method to calculate the pair-wise similarity among documents is $O(m^2n^2)$, where m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus. The main steps of this method are described in Algorithm 2.



	c_{a1} (0.2)	c_{a2} (0.7)	c_{a3} (0.1)
c_{a1} (0.2)	1	0.2	0.5
c_{a2} (0.7)	0.2	1	0.4
c_{a3} (0.1)	0.5	0.4	1

	c_{b1} (0.3)	c_{b2} (0.2)	c_{b3} (0.5)
c_{b1} (0.3)	1	0.3	0.7
c_{b2} (0.2)	0.3	1	0.8
c_{b3} (0.5)	0.7	0.8	1

Figure 3.2: Document similarity measure based on *SMM*. The measure includes three steps: Median Searching, Single Median Mapping, and Similarity Calculation. In each concept set, we find the concept with the maximum average relatedness to other concepts within the same set, which is called median. We map the median concept of d_a to the median concept of d_b to get the document similarity.

Algorithm 2 Semantic similarity based on *SMM*

Input: A *document-concept* matrix

Output: A *document-document* similarity matrix

1: **for** each pair of documents (d_a and d_b) from the input **do**
2: **for** each document **do**
3: Form a *concept-concept matrix* M*, row and column are same concept vector of this document*
4: Find median concept and record the median's *cf-idf* weight w
5: **if** There are more than one concept medians for a document **then**
6: Choose the concept with the greatest *cf-idf* weight.
7: **end if**
8: **end for**
9: **end for**
10: Use Eq. 3.2 to get $sim(d_a, d_b)$
11: **return** A *document-document* similarity matrix

3.3 One to One Mapping (*OOM*) Measure

In this method, each document is represented as a Wikipedia concept vector without any weights. By firstly creating a *concept-concept* matrix M for a pair of documents, we map row concept to column concept following a one to one mapping rule. Each concept of one document can be mapped for only one time. Once a concept pair formed, both of concepts lose chance in the other mappings. The document similarity is calculated by the average score of all concept pairs' relatedness.

Definition 3.3. One to One Mapping (*OOM*): Given two documents and their associate concepts, each concept in the smaller document is mapped to a different concept in the larger document to get the document similarity.

The main steps of this method are shown in Fig. 3.3 and follows the steps below. In the following part, document a is d_a, document b is d_b, the ith concept of d_a is c_{ai}, the jth concept of d_b is c_{bj}, w_{ai} is the *cf-idf* value of c_{ai}, and w_{bj} is the *cf-idf* value of c_{bj}.

1. **Matrix Creation**:

   We make the document with less concepts as the row vectors and the document with more concepts as the column vectors. In this way, an $I * J$ $(I \leq J)$ matrix M is formed, where I is the number of less concepts and J is the number of more concepts. The entry (i, j) of M is represented as $rel(i, j)$, which is the relatedness of concept c_{ai} to concept c_{bj}.

2. **One to One Mapping**:

   For each row in M, we first find the column with the maximum value $rel(i)_{max}$ and save it, and then delete that column. Once a column concept is mapped, it cannot be mapped in the other mappings. In this way, we map each row concept to a column concept.

3. **Similarity Calculation**:

$$sim(d_a, d_b) = \frac{\sum_{i=1}^{I} rel(i)_{max}}{I} \tag{3.3}$$

   where I is the number of row in M, $rel(i)_{max}$ is the maximum value of each row in M.

Due to ignoring the unused concepts in the column vector, this method results in information loss. Given concept vectors of documents, the time complexity of this method to calculate the pair-wise similarity among documents is $O(m^2n^2)$, in which m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus. Besides, we don't use any *cf-idf* weights in this method because we want to see how important the *cf-idf* is in similarity calculation. The main steps of this measure are mentioned in Algorithm 3.

Algorithm 3 Semantic similarity based on *OOM*

Input: A *document-concept* matrix

Output: A *document-document* similarity matrix

1: **for** each pair of document vectors (d_a and d_b) from the input **do**
2: A = number of concepts of d_a;
3: B = number of concepts of d_b;
4: **if** $A \leq B$ **then**
5: $I = A, J = B$
6: **else**
7: $I = B, J = A$
8: **end if**
9: Form an $I * J$ *concept-concept* matrix M while rows correspond to concepts of smaller concept set and columns correspond to concepts of larger concept set.
10: **for** each row in M **do**
11: Find the maximum column value $rel(i)_{max}$ and save it, and then delete that column
12: **end for**
13: Use Eq. 3.3 to get $sim(d_a, d_b)$
14: **end for**
15: **return** A *document-document* similarity matrix

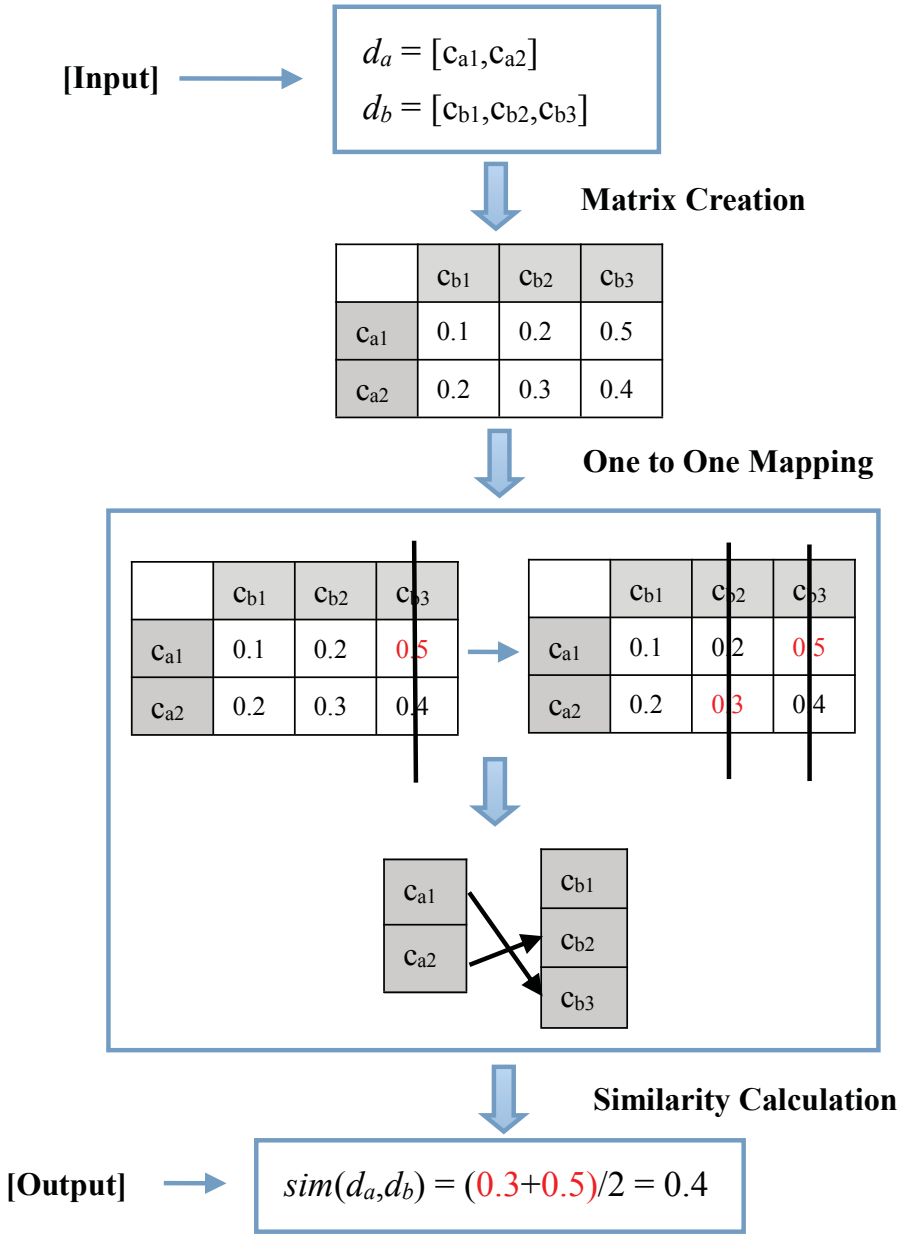


Figure 3.3: Document similarity measure based on *OOM*. The measure includes three steps: Matrix creation, One to One Mapping, and Similarity Calculation. We map each concept in the smaller concept set to the non-duplicate concepts in the larger concept set.

3.4 Multiple to One Mapping (*MOM*) Measure

In this method, each document is represented as a Wikipedia concept vector. The feature value is *cf-idf* here.

Same as in the *OOM* method, we firstly create a *concept-concept* matrix M for each pair of documents. We further realize a multiple to one concept mapping procedure on that matrix. Different concepts of one concept set can be mapped to one same concept of the other concept set. Then we add all the weighted mapped concept pairs' relatedness to get the document similarity.

Definition 3.4. Multiple to One Mapping (*MOM*): Given two documents and their associate concepts, each concept in the larger document is mapped to a concept in the smaller document to get the document similarity.

The main steps are described below and shown in Fig. 3.4. In the following part, document a is d_a, document b is d_b, the ith concept of d_a is c_{ai}, the jth concept of d_b is c_{bj}, w_{ai} is the *cf-idf* value of c_{ai}, and w_{bj} is the *cf-idf* value of c_{bj}.

1. **Matrix Creation**:
   We make the document with more concepts as the row vector and the document with fewer concepts as the column vector. Each concept has its *cf-idf* weight w. In this way, an $I * J$ matrix M is formed, where I is the number of more concepts and J is the number of less concepts. The entry (i, j) of the matrix is represented as $rel(i, j)$, which is the relatedness of c_{ai} to c_{bj}.

2. **Multiple to One Mapping**:
   For each row in M, we find the column j with the maximum value $rel(i)_{max}$ and save them.

3. **Similarity Calculation**:

$$sim(d_a, d_b) = \sum_{i=1}^{I} w_{ai} \cdot w_{bj} \cdot rel(i)_{max} \tag{3.4}$$

   where I is the number of rows in M, $rel(i)_{max}$ is the maximum value of each row in M.

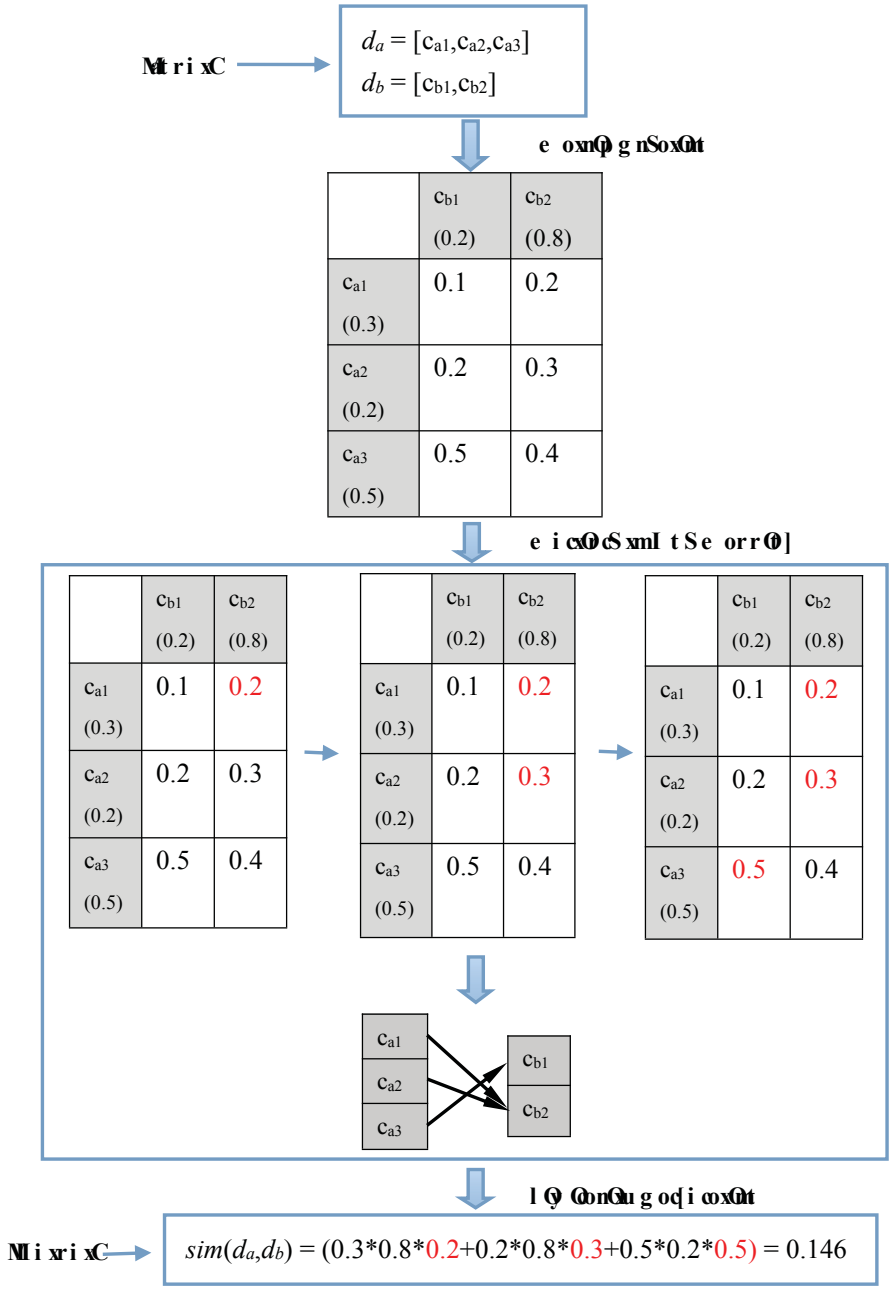


Figure 3.4: Document similarity measure based on *MOM*. The measure includes three steps: Matrix Creation, Multiple to One Mapping, and Similarity Calculation. We map every concept in the larger set to one concept in the smaller concept set.

This method considers all concepts in the larger concept set. Given concept vectors of documents, the time complexity of this method to calculate the pair-wise similarity among documents is $O(m^2n^2)$, in which m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus. The main steps of this measure are described in Algorithm 4.

Algorithm 4 Semantic similarity based on *MOM*

Input: A *document-concept* matrix

Output: A *document-document* similarity matrix

1: **for** each pair of document vectors (d_a and d_b) from the input **do**
2: A = number of concepts of d_a;
3: B = number of concepts of d_b;
4: **if** $A \geq J$ **then**
5: I=A, J=B;
6: **else**
7: I=B, J=A
8: **end if**
9: Form a $I * J$ *concept-concept* matrix M while rows correspond to concepts of larger concept set and columns correspond to concepts of smaller concept set. Each concept has its *cf-idf* weight.
10: **for** each row in the M **do**
11: Find the maximum column value $rel(i)_{max}$ and save it.
12: **end for**
13: Use Eq. 3.4 to get $sim(d_a, d_b)$
14: **end for**
15: **return** A *document-document* similarity matrix

3.5 Multiple to Multiple Mapping (*M3*) Measure

In the *SCM* measure and *SMM* measure, we just take one concept as the representative for one document. In *OOM* measure and *MOM* measure, though we take more concepts into consideration, but only parts of concept relatedness or parts of concept weights are utilized. In this method, we use a multiple to multiple mapping. After forming a *concept-concept* matrix M, we explore more about the semantic information we have extracted from each document by involving all concept vectors and concept relatedness.

Definition 3.5. Multiple to Multiple Mapping (*M3*): Given two documents and their associate concepts, each concept in one document is mapped to all concepts in the other document to get the document similarity.

The main steps are as below and shown in Fig. 3.5. In the following part, document a is d_a, document b is d_b, the ith concept of d_a is c_{ai}, the jth concept of d_b is c_{bj}, w_{ai} is the *cf-idf* value of c_{ai}, and w_{bj} is the *cf-idf* value of c_{bj}.

1. **Matrix Creation**:
   Given a pair of documents d_a and d_b, we assign concepts of d_a as row vectors and concepts of d_b as column vector. Each concept has its *cf-idf* weight w. In this way, an $I * J$ matrix M is formed, where I is the number of concepts from d_a and J is the number of concepts from d_b. The entry (i, j) of M is represented as $rel(i, j)$, which is the relatedness of c_{ai} to c_{bj}.

2. **Multiple to Multiple Mapping**:
   For each concept c_{ai} of d_a, we map it to all concepts of d_b and keep the concept relatedness between each pair of concepts.

3. **Similarity Calculation**:

$$sim(d_a, d_b) = \frac{\sum_{i=1}^{I} \sum_{j=1}^{J} w_{ai} \cdot w_{bj} \cdot rel(i, j)}{\sqrt{\sum_{i=1}^{I} w_{ai}^2 \sum_{j=1}^{J} w_{aj}^2}} \tag{3.5}$$

   where I is the number of concepts in d_a, J is the number of concepts in d_b, $rel(i, j)$ is the relatedness between c_{ai} and c_{bj}.

This method utilizes both the concept relatedness and concept weights. Given concept vectors of documents, the time complexity of this method to calculate the pair-wise similarity among documents is $O(m^2n^2)$, in which m is the number of documents and n is the number of concepts extracted from Wikipedia for the whole corpus. The main steps of this measure are described in Algorithm 5.

Algorithm 5 Semantic similarity based on *M3*

Input: A *document-concept* matrix

Output: A *document-document* similarity matrix

1: **for** each pair of document vectors (d_a and d_b) from the input **do**
2: Form a $I * J$ *concept-concept* matrix M while rows correspond to concepts of d_a and columns correspond to concepts of d_b. Each concept has its *cf-idf* weight.
3: **for** each row in M **do**
4: Save all $rel(i, j)$.
5: **end for**
6: Use Eq. 3.5 to get $sim(d_a, d_b)$
7: **end for**
8: **return** A *document-document* similarity matrix

This method can also be deemed as an extension of cosine similarity based on Vector Space Model (*VSM*). Suppose that document d_a is represented by a term vector (t_1, t_2, t_3, t_4), each term is weighted with its *term frequency-inverse document frequency* (*tf-idf*)[1] as w_i. d_b is represented by (t'_1, t'_2, t'_3, t'_4), each term is weighted by its *tf-idf* as w'_i. The similarity between two documents is expressed as the *Cosine* measure given in the Eq. 3.6.

$$\cos(\overrightarrow{d_a}, \overrightarrow{d_b}) = \frac{[w_1, w_2, w_3, w_4] \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} w'_1 \\ w'_2 \\ w'_3 \\ w'_4 \end{bmatrix}}{\sqrt{\sum_{i=1}^{4} w_i^2 \sum_{i=1}^{4} w_i'^2}} \tag{3.6}$$

However, this is not realistic because this assumption ignores all the relatedness between each pair of terms. For two documents d_a and d_b, if we take term relatedness

[1] `https://en.wikipedia.org/wiki/Tf-idf`

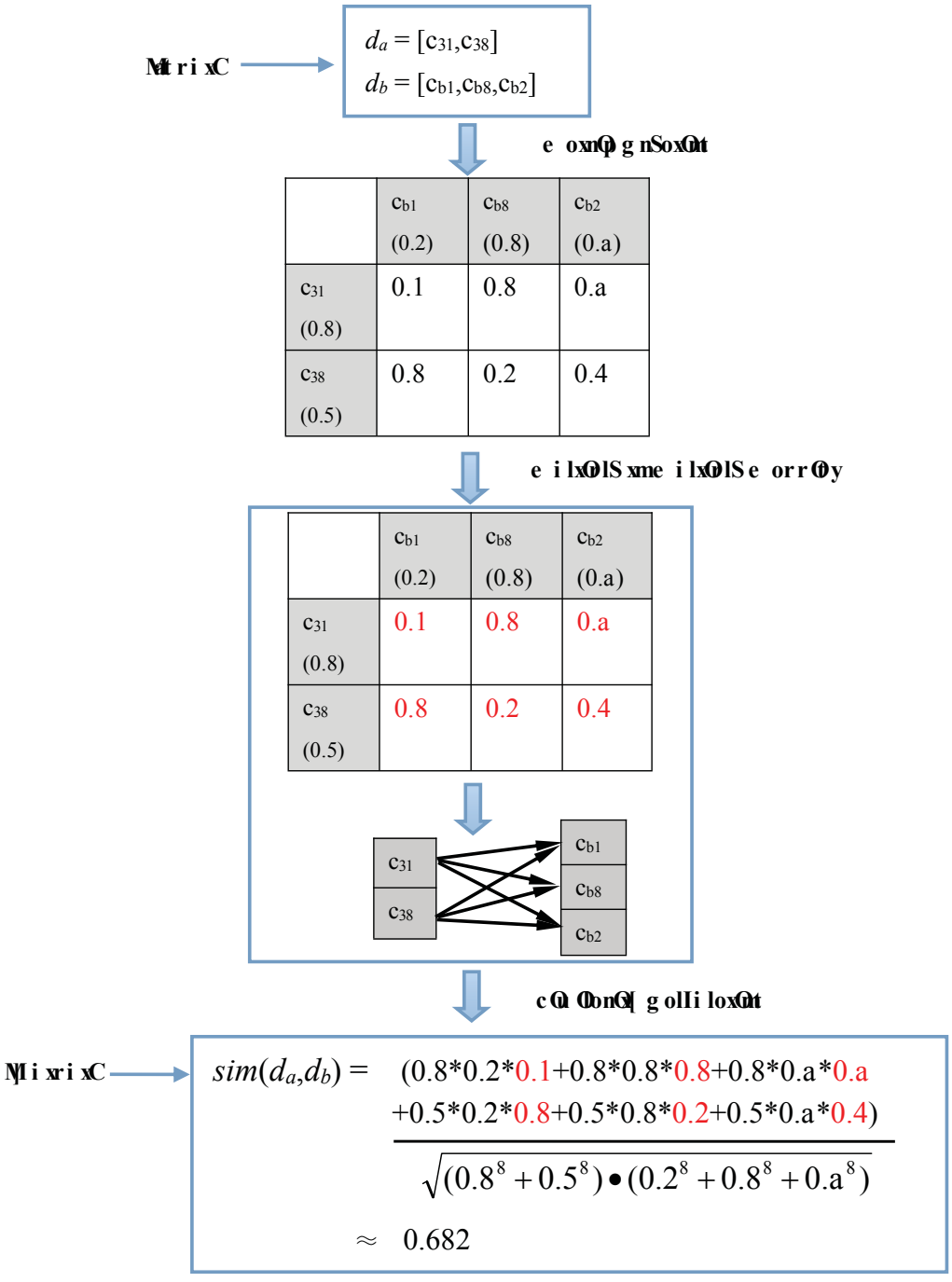


Figure 3.5: Document similarity measure based on *M3*. The measure includes three steps: Matrix Creation, Multiple to Multiple mapping, and Similarity Calculation. We map each concept of d_a to each concept of d_b to make a comprehensive concept mapping.

into consideration, the new *Cosine* similarity measure can be expressed by Eq. 3.7 and Eq. 3.8. Similarly, in the BOC model, we consider all Wikipedia concepts are dependent and have relationship with each other. The similarity is measured by new *Cosine* similarity measure and utilize the concept relatedness to replace the term relatedness in the equations. This new *Cosine* similarity is another form of Algorithm 5.

$$W = \begin{bmatrix} 1 & rel(t_1, t'_2) & rel(t_1, t'_3) & rel(t_1, t'_4) \\ rel(t_2, t'_1) & 1 & rel(t_2, t'_3) & rel(t_2, t'_4) \\ rel(t_3, t'_1) & rel(t_3, t'_2) & 1 & rel(t_3, t'_4) \\ rel(t_4, t'_1) & rel(t_4, t'_2) & rel(t_4, t'_3) & 1 \end{bmatrix} \tag{3.7}$$

$$\cos(\overrightarrow{d_a}, \overrightarrow{d_b}) = \frac{[w_1, w_2, w_3, w_4] \cdot W \cdot \begin{bmatrix} w'_1 \\ w'_2 \\ w'_3 \\ w'_4 \end{bmatrix}}{\sqrt{\sum_{i=1}^{4} w_i^2 \sum_{i=1}^{4} w_i'^2}} \tag{3.8}$$

Chapter 4

Experiments

In this chapter, we first introduce the whole framework of our experiments in Section 4.1. We review characteristics of the datasets used in the experiments in Section 4.3. And we explain how to create and pre-process the datasets using Natural Language Processing methods in Section 4.2. The algorithms and evaluation measures used in this thesis are mentioned in Section 4.4 and Section 4.5. Finally, the experimental results are presented in Section 4.6.

4.1 Framework for Experiments

The framework of our experiments which leverages *SCM*, *SMM*, *OOM*, *MOM*, and *M3* as the semantic document similarity measures for document clustering is presented in Figure 4.1.

We conduct different document similarity measures to different data representations. We apply *Cosine* similarity in BOW and BOC. At the same time, we apply *SCM*, *SMM*, *OOM*, *MOM*, and *M3* in BOC. A *document-document* similarity matrix is generated from each similarity measure. Then, we convert the similarities into distances for document clustering. Documents are clustered by three clustering algorithms including Agglomerative clustering [27], Partitional clustering [4] and *LDA*-based clustering [19] for a comprehensive comparison. Finally, we evaluate and compare the quality of clusterings based on different evaluation measures.

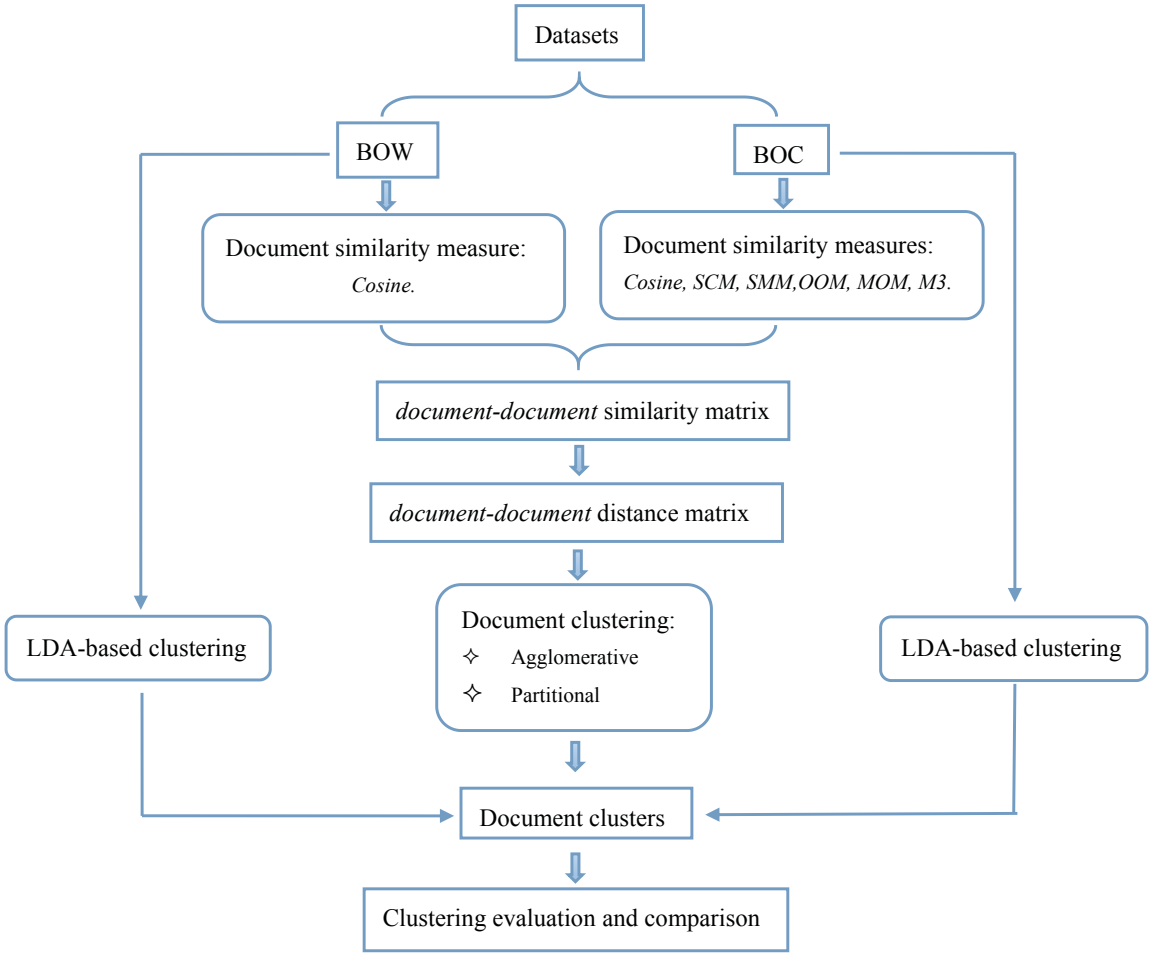


Figure 4.1: The framework of experiments, which leverages different document similarity measures and different clustering algorithms for document clustering.

4.2 Document Representation

In this section, we describe how we process the original datasets to create two data models: BOW model and BOC model. In the BOC model, we also mention how to get concept relatedness.

4.2.1 BOW Model

We pre-process the document collections in the following steps:

1. Remove all the stop words from the original document contents. As the documents are all in English, we remove all the English stop words[1].

2. We use Porter stemming[2] to stem the vocabulary of the collection to reduce dimensionality of the datasets.

3. Remove all the non-alphabet characters.

After the pre-processing steps, each document is represented as a term vector. Each dataset is then represented as a document-term matrix. Each entry of the matrix is the *term frequency-inverse document frequency* (*tf-dif*) value of the respective term in the respective document:

$$tf(t, doc) = \begin{cases} n & \text{if } t \text{ appears in } doc \text{ for } n \text{ times} \\ 0 & \text{if } t \text{ does not appear in } doc \end{cases} \tag{4.1}$$

$$idf(t, Doc) = \log \frac{N}{|doc \in Doc : t \in doc|} \tag{4.2}$$

$$tfidf(t, doc, Doc) = tf(t, doc) \cdot idf(t, Doc) \tag{4.3}$$

where t is a term, doc is a text document, and Doc is the whole corpus.

We normalize the document vectors by *L2* norm[3] to 1. The output of the model is a *document-term* matrix.

[1] `http://www.ranks.nl/stopwords`
[2] `http://tartarus.org/martin/PorterStemmer/java.txt`
[3] `http://mathworld.wolfram.com/L2-Norm.html`

Figure 4.2: *wikify* service in Wikipedia Miner. The input for *wikify* is original text document, the output is shown as *wikifiedDocument* with all detected keywords and corresponding Wikipedia concepts.

4.2.2 BOC Model

In this part, we firstly introduce how to generate related Wikipedia concepts for text documents and then mention how to get concept relatedness for a pair of Wikipedia concepts.

Wikification for a Document

We keep the original documents for further concept extraction to create the BOC model based on Wikipedia. This is a task which first extracts the most important words or phrases as keywords in the document and then identify the appropriate link to a Wikipedia article for each such keyword. The Wikipedia Miner in [23] offers such wikification service called *wikify*[4].

Each text corpus is represented by a *document-concept* matrix using the following steps:

1. Input the original content of each text document into *wikify* of Wikipedia Miner.

2. Extract all related Wikipedia concepts from the *wikify* output.

[4] `http://wikipedia-miner.cms.waikato.ac.nz/services/?wikify`

3. Create a *document-concept* matrix, where rows correspond to documents and columns correspond to concepts.

4. Each entry of the matrix is *concept frequency-inverse document frequency* (*cf-idf*) which is described using formulas below:

$$cf(c,d) = \begin{cases} n & \text{if } c \text{ appears in } d \text{ for } n \text{ times} \\ 0 & \text{if } c \text{ does not appear in } d \end{cases} \tag{4.4}$$

$$idf(c,D) = \log \frac{N}{|d \in D : c \in d|} \tag{4.5}$$

$$cfidf(c,d,D) = cf(c,d) \cdot idf(c,D) \tag{4.6}$$

where c is a Wikipedia concept, d is a document of concepts, D is the whole document collection for this corpus.

After we get the BOC model for each dataset, we also use *L2* norm to normalize the length of document vectors to 1 to create the *document-concept* matrix.

Concept Relatedness Measure

We measure relatedness for all possible pairs of concepts appearing in a document collection using the Wikipedia Miner. The *compare* service is provided to get the semantic relatedness between two Wikipedia concepts and also offer details of how ambiguous two concepts have been interpreted. The relatedness is calculated from the in-going and out-going links of Wikipedia article pages, which is called the Wikipedia Link-based Measure (WLM) [34].

WLM has two components: modelling incoming and outgoing hyperlinks, respectively. Given two Wikipedia articles A and B, the hyperlinks found within them are denoted by A_{in} and B_{in}, the hyperlinks are made to them are denoted by A_{out} and B_{out}. WLM first computes the *Cosine* similarity between A_{out} and B_{out} as $WLM_{out}(A,B)$. Then the incoming links are modelled after the *normalized Google distance* [5] as $WLM_{in}(A,B)$. The average of these two components $WLM_{out}(A,B)$ and $WLM_{in}(A,B)$ is the overall relatedness between A and B. Here we have an example in Fig. 4.3.

Since we do not focus on developing a concept relatedness measure, we just mention how to use this service in our experiments.

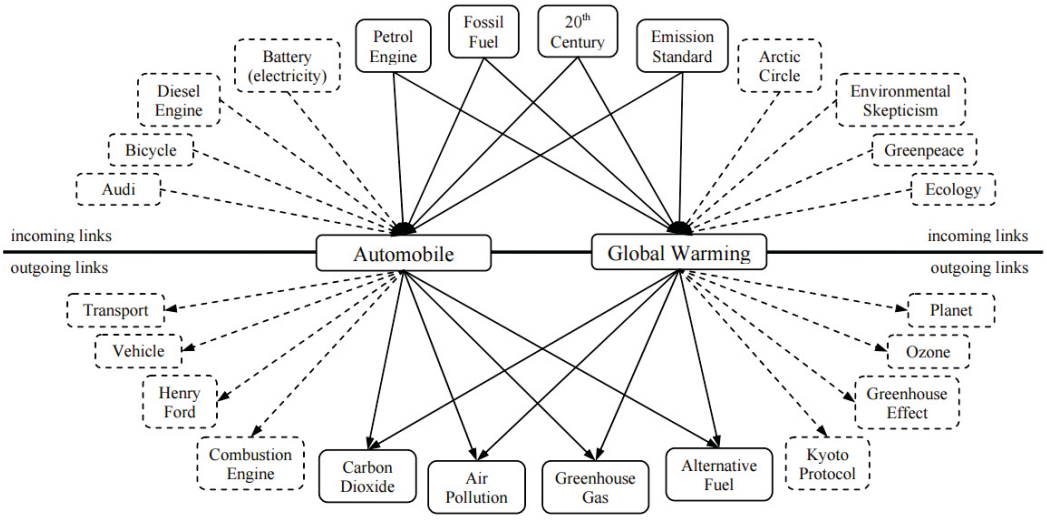


Figure 4.3: Obtaining relatedness between *Automobile* and *Global Warming* from Wikipedia links

The input are Wikipedia concept pairs, and here we use the web service *compare*[5], which uses two Wikipedia concepts as *term1* and *term2*. This service is symmetric, the order of *term1* and *term2* has no influence to the output. Besides, the experiments in [15] have demonstrated its accuracy and consistency with human judgments. We can have different formats (as Json, XML) of output. The output file has one general component named *message*. Under *message*, there are two more sub-components: *request* and *disambiguationDetails*. We extract the *relatedness* value in the *message* and make it as the concept relatedness, which is a value between 0 and 1.

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
▼<message service="/services/compare" relatedness="0.7269239395969522">
  ▼<request>
     <param name="term1">kiwi</param>
     <param name="term2">takahe</param>
   </request>
  ▼<disambiguationDetails term1Candidates="3" term2Candidates="1">
     <interpretation id1="17362" id2="5274085" title1="Kiwi" title2="Takahe"
     relatedness="0.7505231773143588" disambiguationConfidence="0.9610324558353055"/>
   </disambiguationDetails>
 </message>
```

Figure 4.4: *compare* service in Wikipedia Miner. The input for *compare* is two Wikipedia concepts, we extract the relatedness coefficient from the output.

[5] http://wikipedia-miner.cms.waikato.ac.nz/services/?compare

4.3 Datasets

We use four standard document collections in our experiments to compare different text document similarity measures. For efficiency, six small datasets are created from the four standard datasets with different dimensionality and different numbers of clusters. For each data collection, we first choose topics. Then in each topic group, we randomly choose 100 documents to represent the group according to the experiments performed in [12]. Documents from different groups of the same dataset may share similar topics or not. We give a brief review for the datasets generated from each collection in below:

1. *20Newsgroups*:
   20Newsgroups[6] is a document collection with about 20,000 newsgroup documents which have been divided into 20 different topics.

   (a) *Similar-4*: We choose four topics including *comp.os.ms-windows.misc*, *comp.sys.ibm.pc.hardware*, *comp.sys.ma-hardware* and *comp.windows.x*. These topics share the assemble themes about hardware of computers.

   (b) *Diff-5*: Five different topics are selected including *alt.atheism*, *misc.forsale*, *rec.sport.baseball*, *sci.electronics*, and *talk.politics.mideast*.

   (c) *Multi-7*: This dataset has articles with seven topics including *alt.atheism*, *comp.sys.ibm.pc.hardware*, *rec.sport.baseball*, *sci.electronics*, *sci.med*, *soc.religion.christian*, and *talk.politics.guns*. This is a mixture dataset with different topics.

2. *SMART*
   SMART[7] data repository contains abstracts of paper about *Medical, Information retrieval, Aerodynamics*, and *Computing algorithm*. We created one dataset from this repository and used in our experiments. *Classic-4* contains all 4 different topics: *CACM, CISI, CRAN, and MED*.

[6] `http://qwone.com/~jason/20Newsgroups/`
[7] `http://www.dataminingresearch.com/index.php/2010/09/classic3-classic4-datasets/`

3. *WebKB*

   WebKB[8] contains web pages collected from computer science departments of four universities. There are 8,282 pages which were manually classified into 7 classes: *student*, *faculty*, *staff*, *departments*, *course*, *project*, and *other*. We selected only 4 categories including *student, faculty, course* and *project*, which have more documents than the other categories according to the experiments in [25]. We name this dataset *Webkb-4* in our experiments.

4. *Reuters-21578*

   Reuters-21578 is the most widely used collection for text categorization research[9]. The data was originally collected and labeled by Carnegie Group, Inc. and Reuters, Ltd. in the course of developing the CONSTRUE text categorization system. We created one datasets by selecting a subset of topics including *acq*, *crude*, *earn*, *grain*, *interest*, *money-fx*, *ship*, and *trade*. We randomly selected 100 documents from each class. There are only 51 documents in the *grain* class, in this way, we choose all of the documents in this class. We name this new dataset *R-751* in our experiments. It contains 8 classes and 751 documents.

All the datasets mentioned above are summarized in Table 4.1.

Table 4.1: Summary of datasets used in our experiments

Dataset	Number of documents	Number of classes	Number of terms	Number of concepts
Classic-4	400	4	4317	1362
Diff-5	500	5	11749	1784
Multi-7	700	7	13236	2583
R-751	751	8	5677	1989
Similar-4	400	4	7980	806
Webkb-4	400	4	9373	1902

[8] `http://www.inf.ed.ac.uk/teaching/courses/dme/html/datasets0405.html`
[9] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

4.4 Document Clustering Methods

We evaluate the semantic similarity measures in the problem of document clustering using the datasets described in Section 4.3.

Among the different classes of clustering algorithms, distance-based methods are the most popular ones in a variety of applications [1]. Distance-based clustering algorithms are divided into two categories: agglomerative clustering and partitional clustering. To evaluate the validity of our semantic similarity methods, we apply them to both agglomerative and partitional clustering algorithms. We also use Latent Dirichlet Allocation (*LDA*) topic model [3] for document clustering as a comparison to evaluate our similarity methods. We next briefly describe how the similarity measures are used in clustering algorithms.

4.4.1 Agglomerative Clustering

The goal of agglomerative clustering is to group documents into clusters based on their pairwise similarities. During the agglomerative clustering process, each element is treated as a cluster of its own. The clusters are sequentially combined into larger clusters based on the shortest distance rule until the settled cluster number is satisfied [33]. There are different agglomerative clustering algorithms. Each algorithm has a linkage method. The linkage method specifies how the pair-wise distance between two clusters should be measured during the cluster combination process. There are seven linkage methods including: *average* linkage, *centroid* linkage, *complete* linkage, *median* linkage, *single* linkage, *ward* linkage, and *weighted* linkage[10].

Notation: cluster r is formed from clusters p and q, n_r is the number of elements in cluster r, x_n is the ith object in cluster r, $\tilde{x}_r$, $\tilde{x}_s$, $\tilde{x}_p$, and $\tilde{x}_q$ are weighted centroids for cluster r, s, p and q, $||\ ||_2$ is *Euclidean* distance, n_r and n_s are the number of elements in clusters r and s.

1. *average*: uses the average distance between all pairs of objects in any two clusters.

$$d(r,s) = \frac{1}{n_r n_s} \sum_{i=1}^{n_r} \sum_{j=1}^{n_s} dist(x_{ri}, x_{sj}) \tag{4.7}$$

[10] `http://www.mathworks.com/help/stats/linkage.html?refresh=true`

2. *centroid*: uses the *Euclidean* distance between the centroids of the two clusters.

$$d(r,s) = ||\bar{x}_r - \bar{x}_s||_2 \tag{4.8}$$

where

$$\bar{x}_r = \frac{1}{n_r}\sum_{i=1}^{n_r} x_{ri} \tag{4.9}$$

3. *complete*: uses the largest distance between objects in the two clusters.

$$d(r,s) = \max(dist(x_{ri}, x_{sj})), i \in (1,...,n_r), j \in (1,...,n_s) \tag{4.10}$$

4. *median*: uses the *Euclidean* distance between weighted centroids of the two clusters.

$$d(r,s) = ||\tilde{x}_r - \tilde{x}_s||_2 \tag{4.11}$$

if cluster r was created by combining clusters p and q, then

$$\tilde{x}_r = \frac{1}{2}(\tilde{x}_p + \tilde{x}_q) \tag{4.12}$$

5. *single*: uses the smallest distance between objects in the two clusters.

$$d(r,s) = \min(dist(x_{ri}, x_{sj})), i \in (i,...,n_r), j \in (1,...,n_s) \tag{4.13}$$

6. *ward*: uses the incremental sum of squares.

$$d(r,s) = \sqrt{\frac{2n_r n_s}{n_r + n_s}} \cdot ||\tilde{x}_r - \tilde{x}_s||_2 \tag{4.14}$$

7. *weighted*: uses a recursive definition for the distance between two clusters.

$$d(r,s) = \frac{d(p,s) + d(q,s)}{2} \tag{4.15}$$

We used the Matlab implementation of agglomerative clustering. The input of these algorithms are *document-document* distances. We use different similarity measures to get similarities among documents and then convert them into *document-document* distances using the following formula, which is a variation from [28]:

$$dist(d_a, d_b) = e^{-sim(d_a, d_b)} \tag{4.16}$$

where d_a is document a, d_b is document b, $sim(d_a, d_b)$ is the similarity between d_a and d_b.

The main steps of agglomerative clustering are mentioned in Algorithm 6.

Algorithm 6 Agglomerative clustering based on similarity measures including *SCM*, *SMM*, *OOM*, *MOM*, and *M3*

Input: A *document-concept* matrix

Output: A document clustering

1: Generate a *document-document* similarity matrix using Algorithm 1, 2, 3, 4, or 5.

2: Create a *document-document* distance matrix using Eq. 4.16.

3: Choose one linkage method to run agglomerative clustering.

4: **return** Document clusters

4.4.2 Partitional Clustering

Partitional clustering algorithms are widely used in the literature [12, 38, 6, 39]. The main two partitional clustering, the *k-medoids* and the *k-means*, are the most widely used. Both algorithms aim to partition n points into k clusters in which each point belongs to the cluster with the smallest mean distance. And both algorithms randomly use a set of k representative points as the initial centres. Each point is assigned to its closest representative. Then in the next iteration, if picking other k points as representatives can improve the clustering quality, the centres will be replaced by the new k representatives selected in this iteration. This approach is applied until convergence. However, the difference between the *k-means* and the *k-medoids* is that the *k-medoids* obtains the representatives from the original data while the *k-means* does not. The *k-means* can define a new virtual representative point as a better central point for this cluster. The advantage of the *k-means* over the *k-medoids* is that it requires smaller number of iterations in order to converge [1]. So we choose the *k-means* as the partitional clustering algorithm. However, the input to standard *k-means* should be object vectors rather than the *document-document* similarities. There is a version of *k-means* named *Relational k-means* [30] which takes a *document-document* distance matrix as input. We employ *Relational k-means* by using Eq. 4.16 to convert the similarities into distances to make valid input. The main pitfall of the *k-means* method is that it is sensitive to the initial set of representatives. To overcome this problem, we run the *k-means* for 100 times and get the average result to minimize the error resulting from random representative

selection. The main steps of clustering algorithm based on *Relational k-means* and similarity measures are described in Algorithm 7.

Algorithm 7 Partitional clustering based on *Relational k-means* and similarity measures including *SCM*, *SMM*, *OOM*, *MOM*, and *M3*

Input: A *document-concept* matrix

Output: A document clustering

1: **for** i = 1 to 100 **do**
2: Generate a *document-document* similarity matrix using Algorithm 1, 2, 3, 4, or 5.
3: Convert the similarity matrix into distance matrix using Eq. 4.16.
4: Run the *Relational k-means* clustering
5: **end for**
6: **return** Document clusters

4.4.3 LDA-Based Clustering

Latent Dirichlet Allocation (*LDA*) is a widely used algorithm for topic modelling and dimension reduction [3]. *LDA* is a three-level hierarchical Bayesian model, in which each item of a collection is modelled as a finite mixture over an underlying set of topics [16]. Each topic is, in turn, modelled as an infinite mixture over an underlying set of word or term probabilities. For using *LDA* as a clustering algorithm, we assume that the number of topics is the same as the number of clusters. In our work, we employ the *JGibbLDA*[11] which is a Java implementation of *LDA*. For one document, *LDA* produces probabilities of each topic. And each document is assigned with the topic with the maximum probability. Documents assigned to the same topic are clustered into one cluster. Here, we use both term-based and concept-based documents. The main steps of clustering algorithm based on *LDA* are described in Algorithm 8.

[11] `http://jgibblda.sourceforge.net/`

Algorithm 8 *LDA*-based clustering

Input: A *bag of terms* or *bag of concepts*

Output: A document cluster

1: Set number of topics to the number of clusters
2: Run *LDA* to generate topic probabilities for documents
3: **for** each document **do**
4: Choose the topic with the largest probability to label the document
5: **end for**
6: Documents with the same label are in the same clusters.
7: **return** Document clusters

4.5 Evaluation Measures

The true labels of documents are used as the gold standard to evaluate the clustering results. Documents are single-labeled. A confusion matrix is created for evaluation of the clusters. We use two measures, *F-score* and *Normalized Mutual Information* (*NMI*). Both measures range from zero to one, with one corresponds to the perfect clustering.

4.5.1 *F-score*

F-score [18] is a popular evaluation measure of for document clustering. *F-score* combines the information of both *precision* and *recall* to evaluate the clustering performance. *precision* is the number of correct positive results divided by the number of all positive results, and *recall* is the number of correct positive results divided by the number of positive results that should have been returned[12]. We use the traditional *F-score* which is the harmonic mean of *precision* and *recall*, which is in Eq. 4.17.

$$F\text{-}score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{4.17}$$

The higher value of F-score indicates the higher accuracy.

[12] `https://en.wikipedia.org/wiki/F1_score`

4.5.2 *Normalized Mutual Information* (*NMI*)

Normalized Mutual Information (*NMI*) [40] is another popular measure of clustering quality. It is defined as the mutual information between the clusters obtained and the ground-truth classes of documents normalized by the arithmetic mean of the maximum entropies of the empirical marginals. In this work, we use the following formulas mentioned in [20]:

$$NMI(W, C) = \frac{I(W, C)}{(H(W) + H(C))/2} \tag{4.18}$$

$$I(W, C) = \sum_k \sum_j \frac{|w_k \cap c_j|}{N} \log \frac{N|w_k \cap c_j|}{|w_k||c_j|} \tag{4.19}$$

$$H(W) = -\sum_k \frac{|w_k|}{N} \log \frac{|w_k|}{N} \tag{4.20}$$

$$H(C) = -\sum_k \frac{|c_k|}{N} \log \frac{|c_k|}{N} \tag{4.21}$$

where $W = w_a, w_2, ..., w_k$ denotes clusters, $C = c_1, c_2, ..., c_k$ denotes classes, $|w_k \cap c_j|$ is the number of common instances between w_k andc_j, and N is the number of documents.

4.6 Experimental Results

In this section, we review and analyze the clusterings obtained by using three clustering algorithms based on the proposed similarity measures and *Cosine* similarity.

4.6.1 Agglomerative Clustering

Different Linkage Methods Comparison

In the first experiment, we run the agglomerative clustering algorithm using different linkage methods and similarity measures. The goal of the experiment is to evaluate the performance of different linkage methods. For each dataset, we use different linkage methods and compare the final clustering results to see which linkage method is the best choice. Here, we briefly describe the experimental results to show *F-score* and *NMI* of each dataset in one figure.

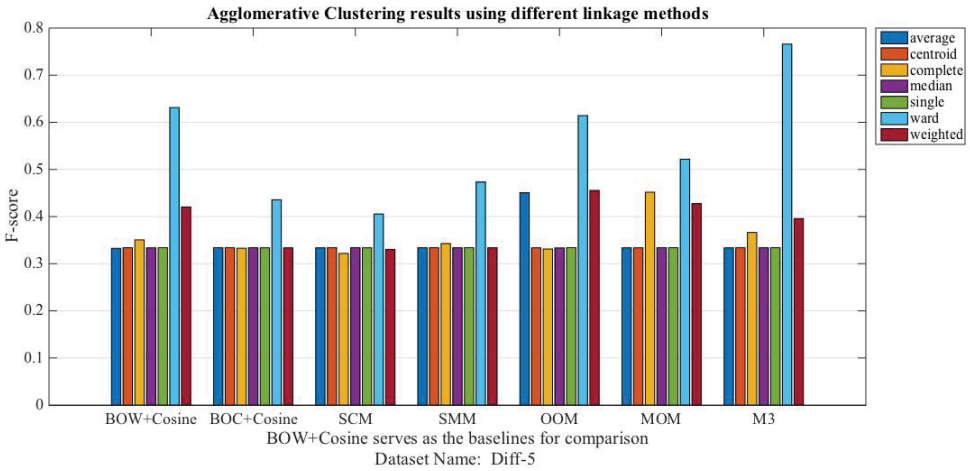


Figure 4.5: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *Diff-5* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

For *Diff5* dataset, we use both *F-score* and *NMI* to evaluate the clustering quality. Different bar groups correspond to different document similarity measures. BOW and BOC employ *Cosine* similarity measure, others use the five similarity measures described in Chapter 3 based on BOC model. Different bars in a group use different linkage methods for clusters. We observe that the *ward* linkage method outperforms other linkage method in each document similarity measure in Fig. 4.5 and Fig. 4.6.

We also conduct experiments on the other five datasets described in Section 4.3 to evaluate the performance of different linkage methods in the agglomerative clustering. The results of those experiments are reported in Appendix A. The main conclusion of this experiment is that the *ward* linkage is a best linkage method for agglomerative clustering in this work.

Agglomerative Clustering Results Based on *ward* Linkage Method

As we can draw the conclusion that *ward* linkage method can always get a better cluster quality in previous experiment, here we extract all agglomerative clustering results based on *ward* linkage method in previous experiment. The goal of this part is

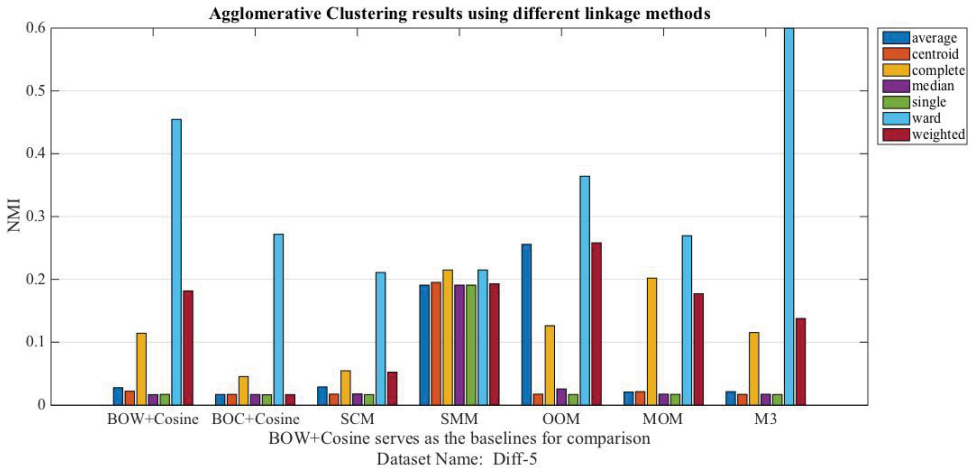


Figure 4.6: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *Diff-5* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

to evaluate the performance of different similarity measures using the same agglomerative clustering method.

We use *Cosine* similarity in both BOW model and BOC model. BOW model with the *Cosine* measure serves as the baseline. Based on BOC model, we used the five new semantic similarity methods to measure the document similarity. The experimental result for the *Diff-5* is shown in Fig. 4.7. The results of the other five datasets are shown in Appendix B.

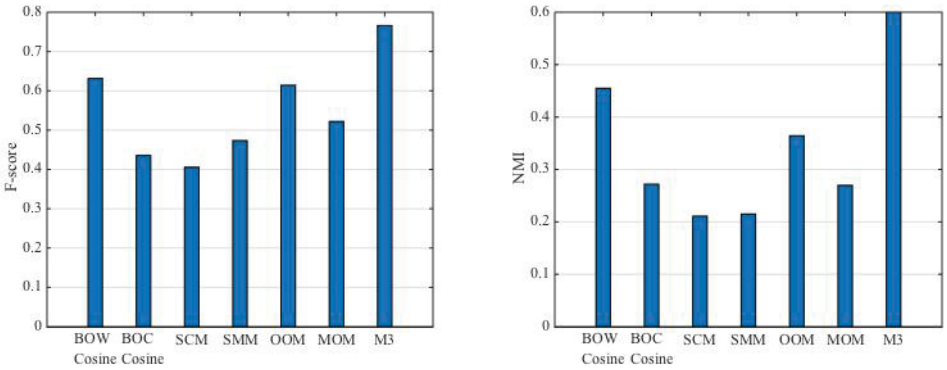


Figure 4.7: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *Diff-5*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* achieves significant improvement in both *F-score* and *NMI* comparing to the baseline.

Comparing to the baseline, we can see that the BOC model with *Cosine* similarity which only relies on the lexical overlap of concepts can not improve clustering performance. This observation is consistent with the previous work in [12] such that clustering based on Wikipedia concepts results in a worse clustering compared to using document terms.

In the BOC model, clustering performances based on *SCM*, *SMM*, *OOM* and *MOM* all resulted in worse clusterings compared to the baseline. It is worth to mention that on *Classic-4*, *Multi-7*, and *Similar-4*, *SCM* generates better cluster quality than *Cosine* measure. In *Classic-4*, *Multi-7*, *Similar-4*, and *Webkb4 SMM* performs better than *Cosine* measure. On *Classic-4*, *Diff-5*, *Multi-7* and *R-751*, *OOM* shows better performance than *Cosine*. And on *Classic-4* and *Multi-7*, *MOM* performs better than *Cosine*. This indicates that employing the concept relatedness into the BOC model can improve the clustering performance to a certain extent. The main observation is that *M3* always results in the best performance by the improvement from 1.99% to 35.39% in *F-score* and from 8.51% to 76.16% in *NMI* value to the baseline on average.

Overall, the agglomerative experiment results show that a comprehensive concept mapping which utilizes all *cf-idf* weights and concept relatedness improves the performance of clustering significantly. This is the reason that *M3* measure always shows a better performance than other measure. The other four proposed semantic

similarity methods though can't beat the baseline, they outperform the BOC model with *Cosine* similarity measure.

4.6.2 Partitional Clustering

Partitional Clustering Results Based on *Relational k-means*

We use a C# implementation[13] of *Relational k-means* clustering algorithm in this experiment. The reason why we use *Relational k-means* has been explained in Section 4.4.2. Since in *k-means* we randomly select the initial representatives as seeds, for each experiment, we run *Relational k-means* for 100 times. Finally we report the average and standard deviation of these 100 runs.

Same as in agglomerative clustering, we use the similarity measures along with *Cosine* similarity. Document terms are only used in BOW model and the other measures are based on concepts extracted from Wikipedia. We convert all the document similarities into document distances using Eq. 4.16. The BOW model with *Cosine* similarity serves as the baseline. The experimental result for *Diff-5* dataset is depicted in Fig. 4.8 and Fig. 4.9. Experimental results for other five datasets are described in Appendix C.

One observation of this experiment is that the BOC model with *Cosine* similarity shows worse performance than other similarity measures in all datasets, which is consistent with the agglomerative clustering results. We cannot replace document terms from the clustering process by using document concepts in our experiments. Just relying on Wikipedia concepts cannot improve the clustering performance.

The clusterings obtained based on the BOC model reveal that the *M3* method outperforms the other similarity measures including the baseline in all six datasets. This is also for the reason that *M3* utilizes all semantic information we get to make a comprehensive mapping.Besides *M3* yields smaller standard deviation. This demonstrates that *M3* can generate relatively stable results in our experiments. *k-means* is sensitive to the initial set of seeds picked during the clustering [1]. *M3* improves this inevitable sensitivity to some extent. It is also worth mentioning that *OOM* produces better average results in *Classic-4*, *Diff-5* and *Multi-7* than the baseline. This indicates that when clustering based on BOC model, considering concept relatedness can improve the clustering performance in our experiments.

[13] `http://arxiv.org/abs/1304.6899`

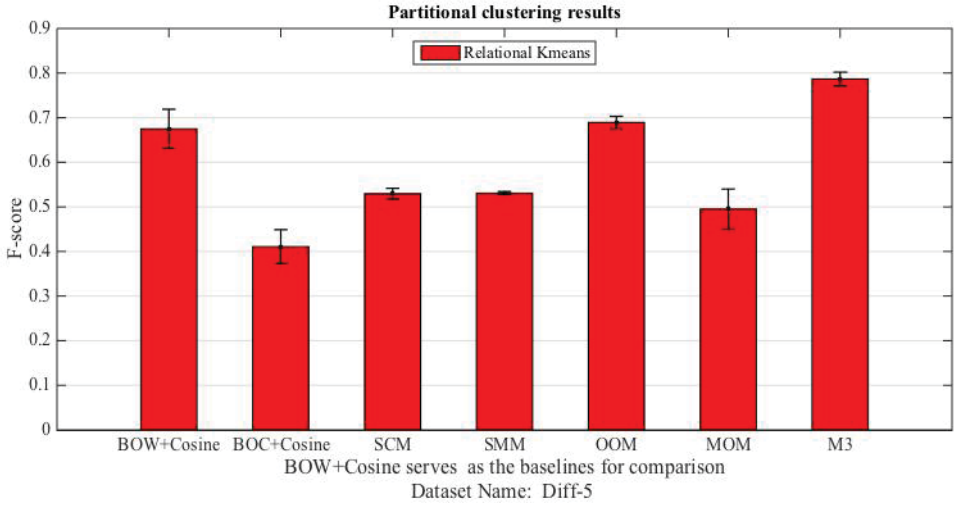


Figure 4.8: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *Diff-5*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures. Compared with the baseline, *M3* and *OOM* generate better results in *F-score*.

4.6.3 Comparison to Clusterings based on *LDA*

The goal of this experiment is to compare clusterings obtained from agglomerative algorithm and partitional algorithm with clusterings obtained from *LDA* topic modelling. *LDA* is a probabilistic model which exploits statistical inference to discover latent pattern of data. *LDA* is often used to discover underlying semantic topics from text data collections [11]. Documents with the same topic can be deemed as a cluster, so we can employ *LDA* for document clustering. First, *LDA* is run on a version of datasets based on terms. The data pre-processing steps for term based *LDA* clustering are just stop-word removal and stemming. Then we made *LDA* run on the BOC model to compare the experimental results.

We use a Java implementation of this model[14] in the experiment. We set the number of topics as the number of classes in the input dataset. There is no objective metric to reveal that the topic modelling is converged. Again in order to make the experimental results more accurate, we let the *LDA* model run for 100 times and each time contains 10,000 iterations. We report the average and standard deviation of these 100 times.

[14] `http://jgibblda.sourceforge.net/`

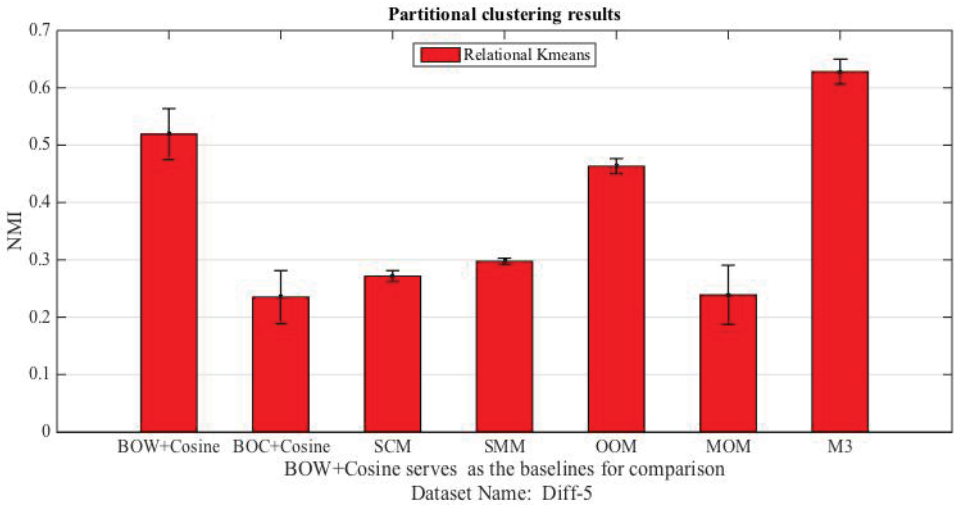


Figure 4.9: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *Diff-5*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *NMI*. Only *M3* outperforms the baseline.

In the following experiment, *LDA* based on terms serves as the state-of-the-art clustering algorithm and the baseline. The experimental results are shown in Table 4.2. The best values obtained in terms of *F-score* and *NMI* are shown in bold font. By comparing *M3*-based agglomerative clustering and *M3*-based partitional clustering with the *LDA*-based clustering, we can see that *M3*-based clustering gets better *F-score* and *NMI* on *Classic-4*. Besides, on *Multi-7* and *Similar-4*, the *M3*-based clustering gets better *NMI* scores.

The experimental results demonstrate that though *LDA* based on terms generally produces better clusterings than our approaches, our *M3* measure can generate comparable results on some datasets. By incorporating concept relatedness into the BOC model and making a complete mapping scheme during document similarity measure, we have made a new way to measure the semantic similarity between documents.

We can also observe that *LDA* based on concepts always generates the worst results in all datasets. This is because BOC model based on Wikipedia concepts suffers from noisy information and sense ambiguity problem, which has been demonstrated in [12]. Though BOC model has a smaller dimensionality, using Wikipedia concepts only loses some detailed information at the same time. And our *M3* based clustering

Table 4.2: Agglomerative and partitional clustering using *M3* as the similarity measure between documents, and *LDA*-based clustering using bag of terms or bag of concepts as the document representation.

Dataset	Algorithm	*F-score*	*NMI*
Classic-4	Agglomerative	0.8774	**0.7762**
	Partitonal	**0.8953±0.0069**	0.7499±0.0105
	LDA (terms)	0.7576±0.8953	0.6696±0.0122
	LDA (concepts)	0.4674±0.023	0.1263±0.0204
Diff-5	Agglomerative	0.7661	0.5999
	Partitonal	0.7868±0.0152	0.6281±0.0218
	LDA (terms)	**0.8756±0.0065**	**0.7166±0.0149**
	LDA (concepts)	0.5660±0.0412	0.3048±0.0410
Multi-7	Agglomerative	0.7345	0.6506
	Partitonal	0.8033±0.0176	**0.7010±0.0109**
	LDA (terms)	**0.8634±0.0151**	0.6565±0.0328
	LDA (concepts)	0.2607±0.0106	0.0517±0.0067
R-751	Agglomerative	0.7250	0.6619
	Partitonal	0.7179±0.0186	0.6470±0.0157
	LDA (terms)	**0.7966±0.0064**	**0.6648±0.0104**
	LDA (concepts)	0.5844±0.0073	0.4247±0.0101
Similar-4	Agglomerative	0.5806	**0.3784**
	Partitonal	0.5540±0.0197	0.2695±0.0284
	LDA (terms)	**0.6616±0.0090**	0.3658±0.0126
	LDA (concepts)	0.3695±0.0193	0.0514±0.0142
Webkb-4	Agglomerative	0.5888	0.3149
	Partitonal	0.6220±0.0106	0.3419±0.0106
	LDA (terms)	**0.6392±0.0096**	**0.3670±0.0163**
	LDA (concepts)	0.3894±0.0115	0.0690±0.0095

outperforms LDA clustering based on concepts, this also demonstrates the effect of considering concept relatedness.

Both our *M3* based clusterings show better performance than *LDA* based clusterings on *Classic-4*, which includes abstracts in *medical*, *information retrieval*, *aerodynamic*, and *computing algorithms*. This is due to the reason that Wikipedia contains more scientific concepts. As for the other datasets, they are contents appearing in newspapers, which contains fewer, if any, scientific terms than *Classic-4*. Therefore, we may infer that our *M3* similarity measure works better on datasets about short scientific papers.

4.6.4 Time Complexity Analysis

In this part, we conducted extensive studies to evaluate the effectiveness and efficiency of proposed semantic similarity measures. All the measures were implemented using Matlab 2014b and tested on a PC with 3.1GHz CPU and 32.0 GB memory running Windows 7.

From the experiment framework in Fig. 4.1, we can see that the differences are formed during the *document-document* similarity creation process. For a better description of the comparison, we take one dataset *Classic-4* as an example to analyze the time costs during matrix creation using different similarity measures. The detailed information of this dataset is in Section 4.3.

The wikification for the original documents in *Classic-4* took 235.584588 seconds via online *wikify* service. We have to measure the relatedness between any two concepts. The concept relatedness measure via online *compare* service took almost 9 hours. We store the concept relatedness in a matrix W for further checking during similarity computation.

Based on different document representations, we can produce the *document-document* similarity matrices using different similarity measures. The real time costs are shown in Table 4.3. We can see that although our new proposed similarity measures have the same theoretical time complexity, the actual time costs vary from each other. *Cosine* similarity based on BOW took less time than our measures. *M3* took more time than the others because of more complex calculation and concept relatedness checking in W.

Table 4.3: Time costs using different similarity measures based on BOW and BOC for *Classic-4*

Similarity Measure	Time Cost for Creating Similarity Matrix (s)
BOW+*Cosine*	0.217
BOC+*Cosine*	0.069
SCM	1153.894
SMM	890.512
OOM	768.333
MOM	710.127
M3	1420.665

Chapter 5

Conclusions and Future Work

In this thesis, we proposed five new semantic similarity measures for text document similarity. The main challenge of the research is how to effectively exploit all semantic information we extracted from Wikipedia. The main contribution of this work is developing a novel similarity measure, which incorporates Wikipedia concept relatedness to relax the orthogonality assumption.

Document similarity measure always plays an important role in many domains, including Biomedical Informatics, GeoInformatics, Linguistics and Natural Language Processing. The measure can reflect the degree of closeness or separation of the target objects [13]. Moreover, choosing an generally appropriate similarity measure is also crucial from cluster analysis. Current concept-based similarity measures mainly suffer two limitations: (1) they do not take semantic information into consideration; (2) they always focus on the content of lexical overlap without considering the inner relatedness between concepts.

Our five new semantic similarity measures are all based on bag of concepts model. Each document is represented by a vector of Wikipedia concepts. For two documents, we use different mapping schemes between two concept sets. By involving the conceptual relatedness into the mapping procedure, we measure the document similarity.

We used document clustering to evaluate these similarity measures. Our tests on six datasets demonstrate that simply replacing the original documents with Wikipedia concepts would result in poor clustering. The experimental results also indicate that though further optimizations could be performed, our *M3* similarity method already outperforms the BOW model and generate comparable clustering results as the *LDA*-based clustering. Compared to *Cosine* similarity, *M3* results in better clustering with smaller standard deviation.

One future work is to use other knowledge ontology to represent documents and

test our semantic measures. Our work uses Wikipedia as the external resource since Wikipedia is a comprehensive resource without suffering the domain coverage limitation. However, as the reason that Wikipedia cover most domains, it loses enough domain specificity like: SNOMED or MESH. We can try different external knowledge ontology with regard to different data so compare whether Wikipedia is appropriate for document clustering task.

Another possible extension of this work is to involve the users' feedback in measuring similarities since our methods are based on Wikipedia concepts, which are more general than terms. In [25], they have considered users' intention into document clustering procedure to make a user-supervised algorithm. Users can easily integrate their minds into the concept selection, which is an advantage over term-based *LDA*.

We can see that our proposed measures take more time than normal *Cosine* similarity for two main reasons: (1) the online services *wikify* for wikification and *compare* for concept relatedness measure impose significant overhead time; (2) more complex calculation in the concept mapping procedure. Further improvement, such as performing *wikify* and *compare* locally as opposed to over the web services, can be pursued to reduce the required time.

Besides, we may extend our work to combine other semantic information like Wikipedia categories and use larger datasets. We may also explore a more comprehensive concept mapping scheme to fully utilize more semantic information hidden in documents.

Bibliography

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

[2] E. Agirre, A. Barrena, and A. Soroa. Studying the Wikipedia hyperlink graph for relatedness and disambiguation. *arXiv preprint arXiv:1503.01655*, 2015.

[3] D.M. Blei, A.Y. Ng, and M.I. Jordan. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[4] M.E. Celebi. *Partitional Clustering Algorithms*. Springer, 2015.

[5] R.L. Cilibrasi and P.M.B. Vitanyi. The Google Similarity Distance. *Knowledge and Data Engineering, IEEE Transactions on*, 19(3):370–383, March 2007.

[6] X. Cui, T.E. Potok, and P. Palathingal. Document clustering using particle swarm optimization. In *Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE*, pages 185–191, June 2005.

[7] E. Gabrilovich and S. Markovitch. Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. In *American Association for Artificial Intelligence*, volume 6, pages 1301–1306, 2006.

[8] F. Goossen, W. IJntema, F. Frasincar, F. Hogenboom, and U. Kaymak. News personalization using the cf-idf semantic recommender. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*, WIMS '11, pages 10:1–10:12, New York, NY, USA, 2011. ACM.

[9] A. Hotho, A. Maedche, and S. Staab. Text clustering based on good aggregations. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 607–608. IEEE, 2001.

[10] A. Hotho, S. Staab, and G. Stumme. Ontologies improve text document clustering. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 541–544. IEEE, Nov 2003.

[11] D. J. Hu. Latent Dirichlet Allocation for text, images, and music. *University of California, San Diego. Retrieved April*, 26:22–33, 2009.

[12] X. Hu, X. Zhang, C. Lu, E.K. Park, and X. Zhou. Exploiting Wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, pages 389–396, New York, NY, USA, 2009. ACM.

[13] A. Huang. Similarity measures for text document clustering. In *Proceedings of the sixth Nsew Zealand Computer Science Research Student Conference (NZCSRSC2008), Christchurch, New Zealand*, pages 49–56, 2008.

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

[15] L. Huang, D. Milne, E. Frank, and I. H. Witten. Learning a concept-based document similarity measure. *Journal of the American Society for Information Science and Technology*, 63(8):1593–1608, 2012.

[16] L. Isaly, E. Trias, and G. Peterson. Improving the Latent Dirichlet Allocation document model with WordNet. In *Proceedings of the 5th International Conference on Information Warfare and Security. London: Academic Conferences Ltd*, pages 163–170, 2010.

[17] A. Islam and D. Inkpen. Semantic text similarity using corpus-based word similarity and string similarity. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2(2):10, 2008.

[18] B. Larsen and C. Aone. Fast and effective text mining using linear-time document clustering. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 16–22, New York, NY, USA, August 1999. ACM.

[19] C.H. Li, B.C. Kuo, and C.T. Lin. LDA-based clustering algorithm and its application to an unsupervised feature extraction. *Fuzzy Systems, IEEE Transactions on*, 19(1):152–163, February 2011.

[20] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge: Cambridge University Press, 2008.

[21] R. Mihalcea, C. Corley, and C. Strapparava. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, American Association for Artificial Intelligence, pages 775–780. AAAI Press, 2006.

[22] R. Mihalcea and A. Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on Information and Knowledge Management*, pages 233–242. ACM, 2007.

[23] D. Milne and I.H. Witten. An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, 194(0):222 – 239, 2013. Artificial Intelligence, Wikipedia and Semi-Structured Resources.

[24] T. Niewiarowski. Tag generalization for facet-based search. Master's thesis, Dalhousie University, Aug 2013. `http://hdl.handle.net/10222/36235`.

[25] S. Nourashrafeddin. *Interactive term supervised text document clustering*. PhD thesis, Dalhousie University, November 2013. `http://dalspace.library.dal.ca/handle/10222/55965`.

[26] S. Nourashrafeddin, E. Milios, and D. V. Arnold. An ensemble approach for text document clustering using Wikipedia concepts. In *Proceedings of the 2014 ACM Symposium on Document Engineering*, DocEng '14, pages 107–116, New York, NY, USA, 2014. ACM.

[27] L. Rokach and O. Maimon. Clustering methods. In *Data Mining and Knowledge Discovery Handbook*, pages 321–352. Springer US, 2005.

[28] R.N. Shepard. Toward a universal law of generalization for psychological science. *Science*, 237(4820):1317–1323, 1987.

[29] C. Stokoe, M.P. Oakes, and J. Tait. Word sense disambiguation in information retrieval revisited. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 159–166, New York, NY, USA, 2003. ACM.

[30] B. Szalkai. An implementation of the relational k-means algorithm. *arXiv preprint arXiv:1304.6899*, 2013.

[31] G. Tsatsaronis and V. Panagiotopoulou. A generalized vector space model for text retrieval based on semantic relatedness. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, EACL '09, pages 70–78, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

[32] P.D. Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In *Proceedings of the 12th European Conference on Machine Learning*, EMCL '01, pages 491–502, London, UK, UK, 2001. Springer-Verlag.

[33] P. Willett. Recent trends in hierarchical document clustering: a critical review. *Information Processing and Management*, 24(5):577–597, 1988.

[34] I. Witten and D. Milne. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, AAAI Press, Chicago, USA*, pages 25–30, 2008.

[35] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting of Association for Computational Linguistics*, pages 133–138. Association for Computational Linguistics, 1994.

[36] I. Yoo, X. Hu, and I. Song. Integration of semantic-based bipartite graph representation and mutual refinement strategy for biomedical literature clustering. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 791–796. ACM, 2006.

[37] X. Zhang, L. Jing, X. Hu, M. Ng, and X. Zhou. A comparative study of ontology based term similarity measures on PubMed document clustering. In *Advances in Databases: Concepts, Systems and Applications*, volume 4443 of *Lecture Notes in Computer Science*, pages 115–126. Springer, 2007.

[38] Y. Zhao and G. Karypis. Evaluation of hierarchical clustering algorithms for document datasets. In *Proceedings of the Eleventh International Conference on Information and Knowledge Management*, CIKM '02, pages 515–524, New York, NY, USA, 2002. ACM.

[39] Y. Zhao, G. Karypis, and U. Fayyad. Hierarchical clustering algorithms for document datasets. *Data Mining and Knowledge Discovery*, 10(2):141–168, 2005.

[40] S. Zhong and J. Ghosh. Generative model-based document clustering: a comparative study. *Knowledge and Information Systems*, 8(3):374–384, 2005.

Appendix A

Clustering Results Using Different Linkage Methods

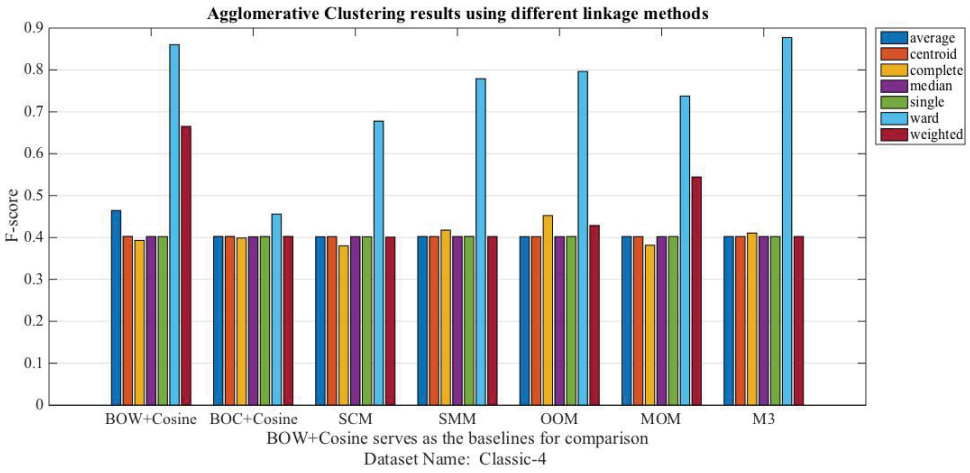


Figure A.1: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *Classic-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

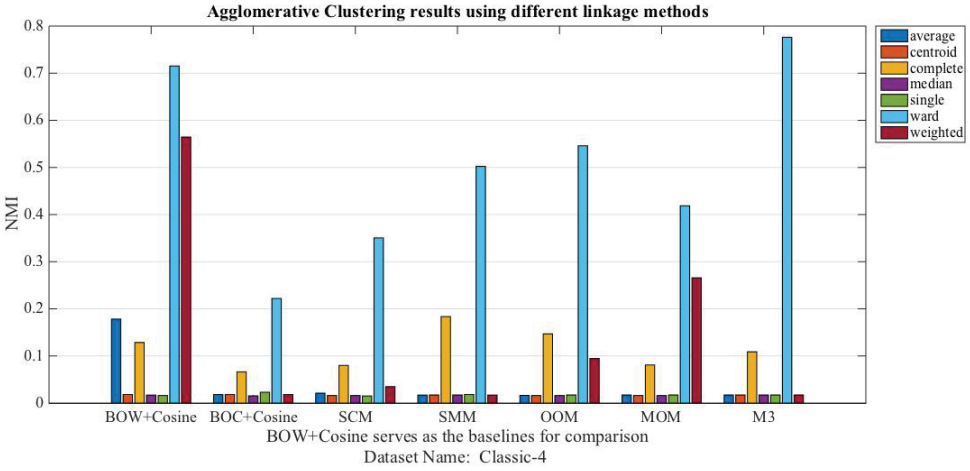


Figure A.2: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *Classic-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

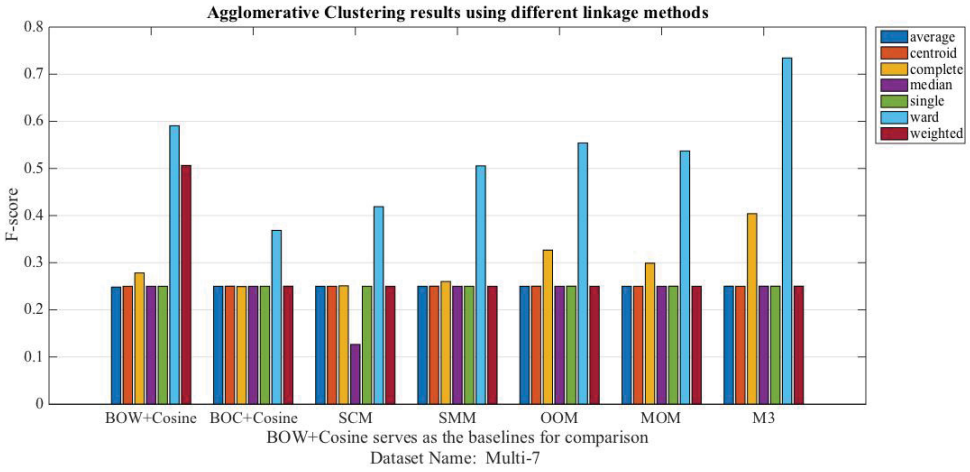


Figure A.3: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *Multi-7* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

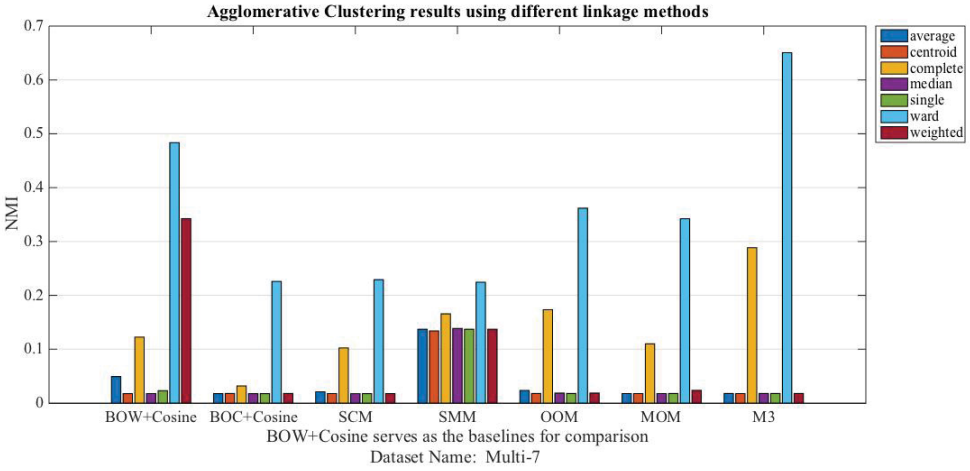


Figure A.4: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *Multi-7* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

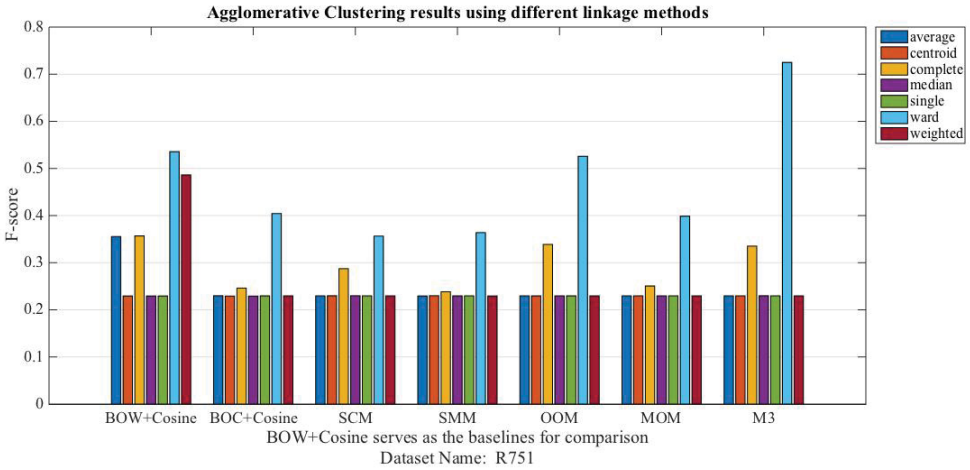


Figure A.5: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *R751* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

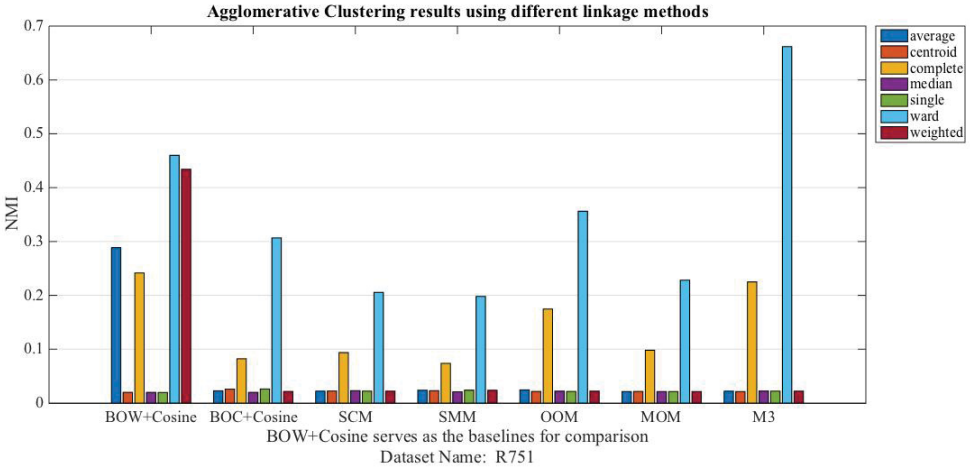


Figure A.6: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *R751* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

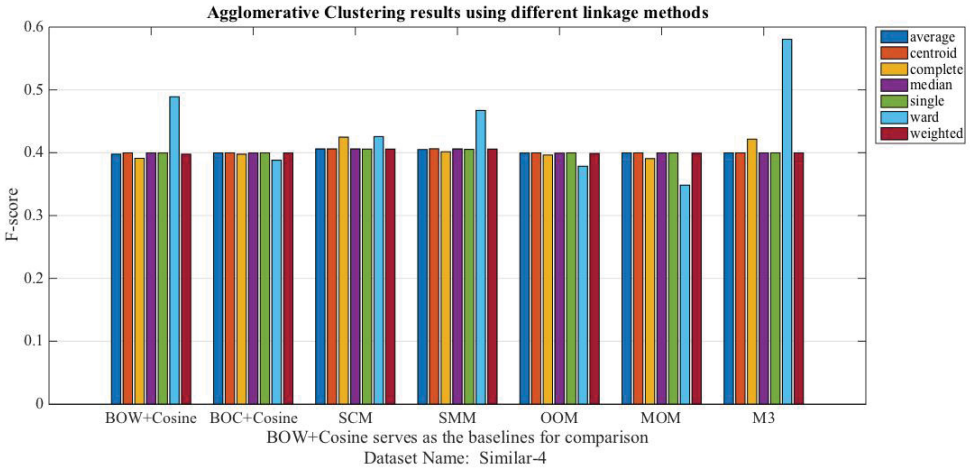


Figure A.7: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *Similar-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

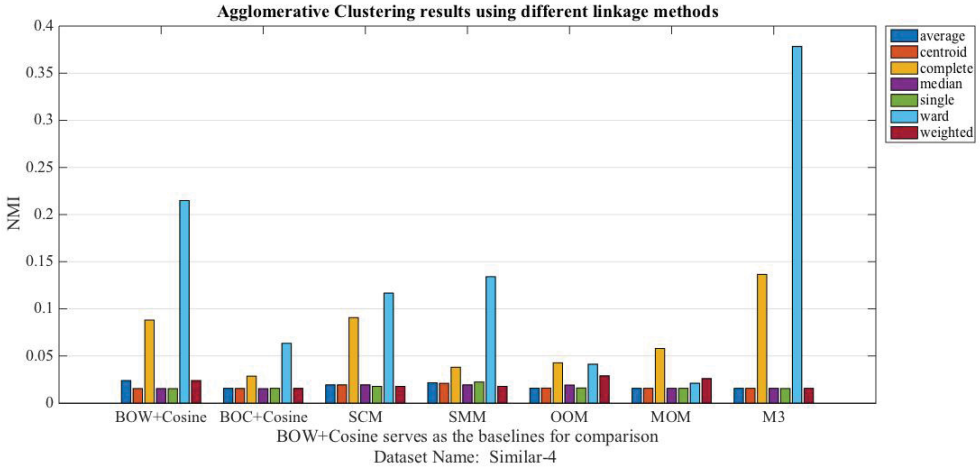


Figure A.8: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *Similar-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

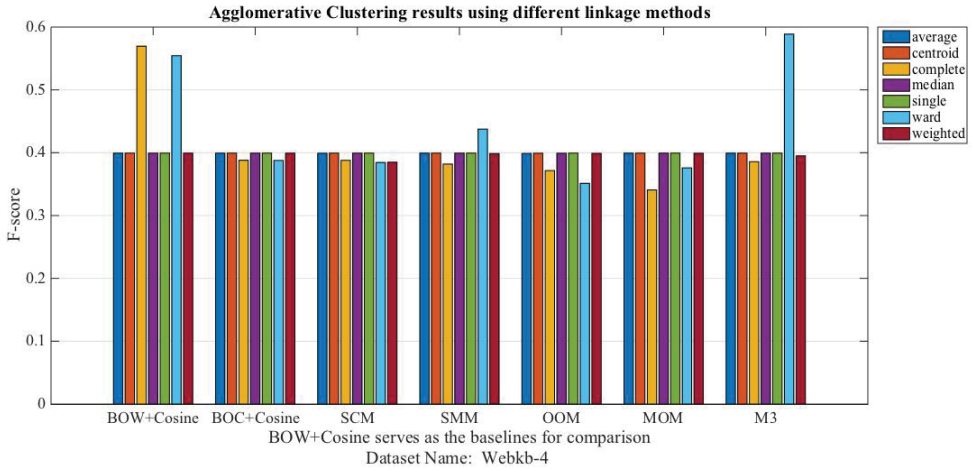


Figure A.9: Agglomerative clustering results using different linkage methods and *F-score* as the measure on *Webkb-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

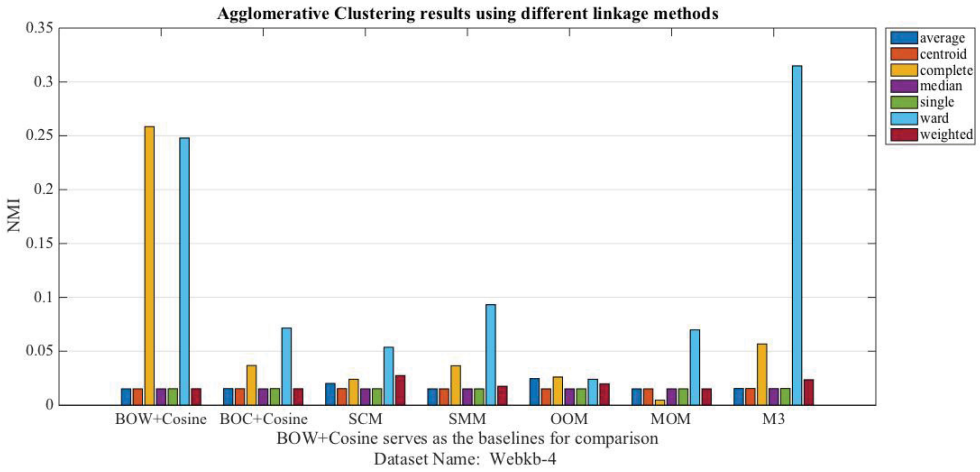


Figure A.10: Agglomerative clustering results using different linkage methods and *NMI* as the measure on *Webkb-4* dataset. Different bar groups use different similarity measures. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. In each bar group, different colours use different linkage methods. The light blue stands for *ward* linkage method, which achieves better performance than other linkage methods.

Appendix B

Agglomerative Clustering Results Using *ward* Linkage Methods

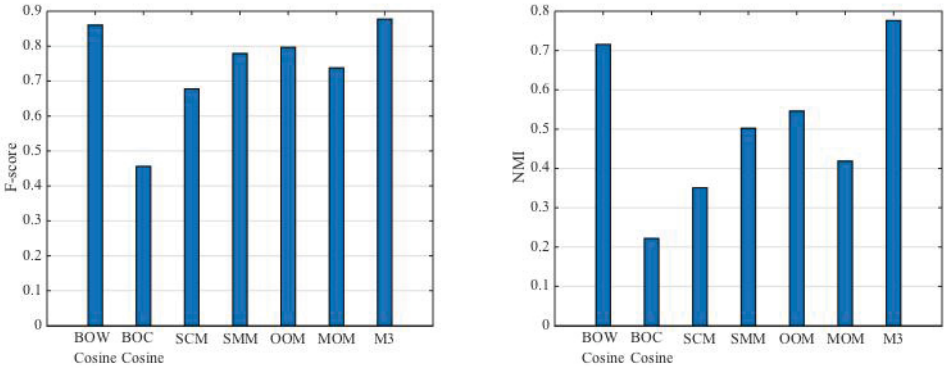


Figure B.1: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *Classic-4*. *M3* outperforms other similarity measure. Compared with the baseline, though not much improvement, M3 still achieves better results.

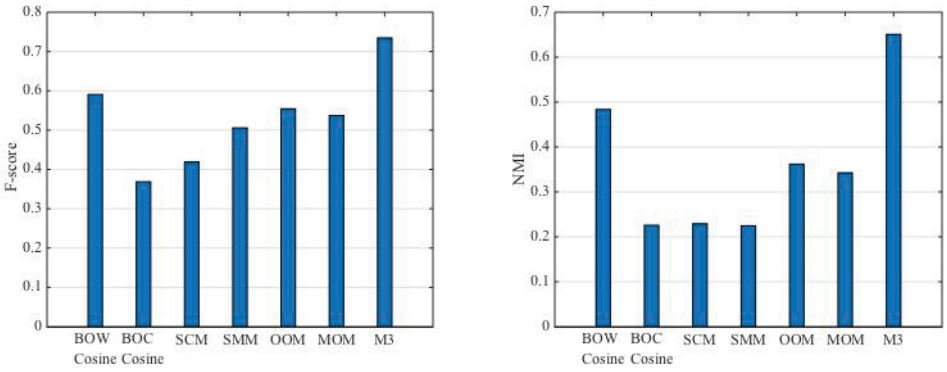


Figure B.2: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *Multi-7*. *M3* again achieves better performance in both *F-score* and *NMI* comparing to other measures.

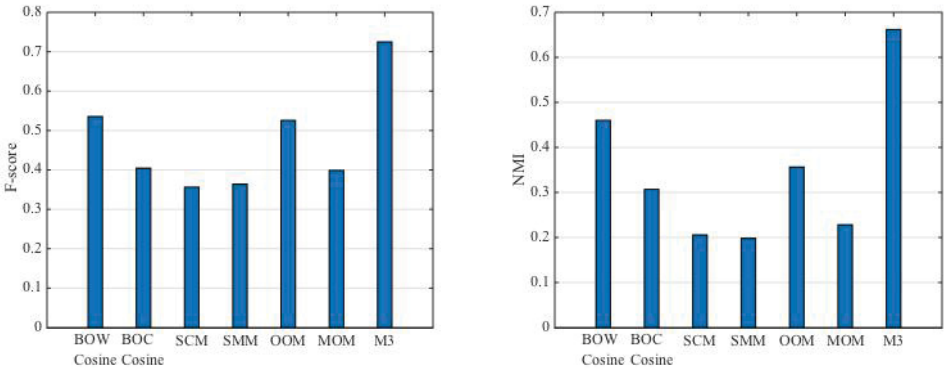


Figure B.3: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *R751*. *M3* outperforms other similarity measures significantly in both *F-score* and *NMI*.

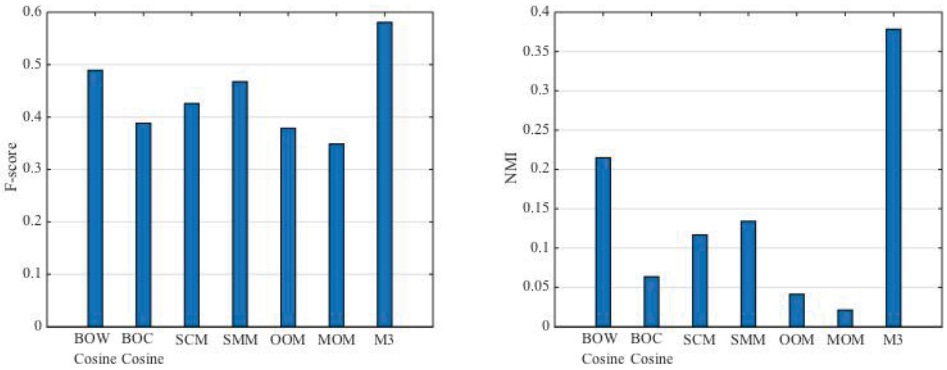


Figure B.4: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *Similar-4*. *M3* outperforms other similarity measures especially in *NMI* evaluation measure.

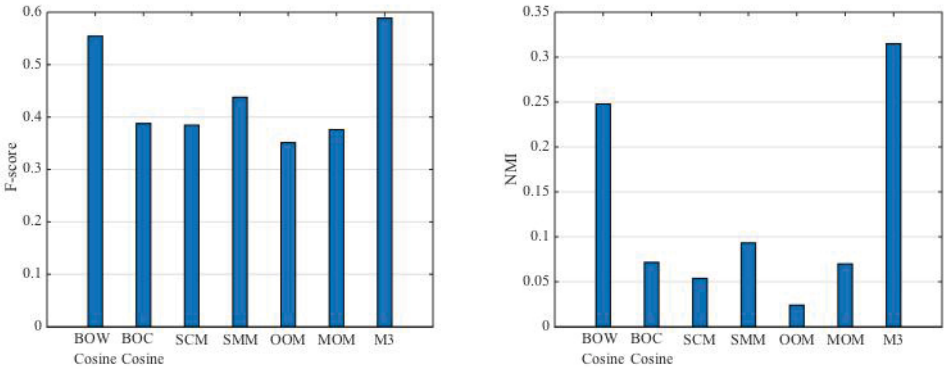


Figure B.5: The quality of clusters in terms of *F-score* and *NMI* obtained from Agglomerative clustering using *ward* linkage on *Webkb4*. *M3* outperforms other similarity measures. Compared with the baseline, though not significant in *F-score*, *M3* generates better result in *NMI*.

Appendix C

Partitional Clustering Results

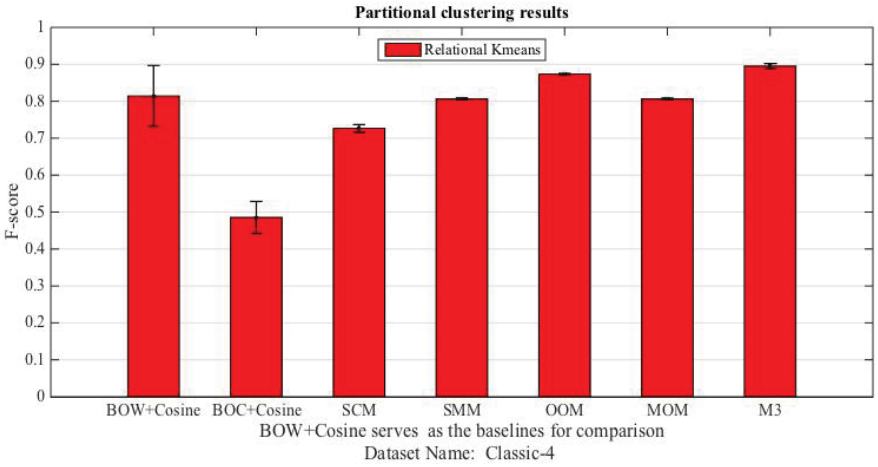


Figure C.1: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *Classic-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures. Compared with the baseline, *M3* achieves better result on *F-score* though not significantly.

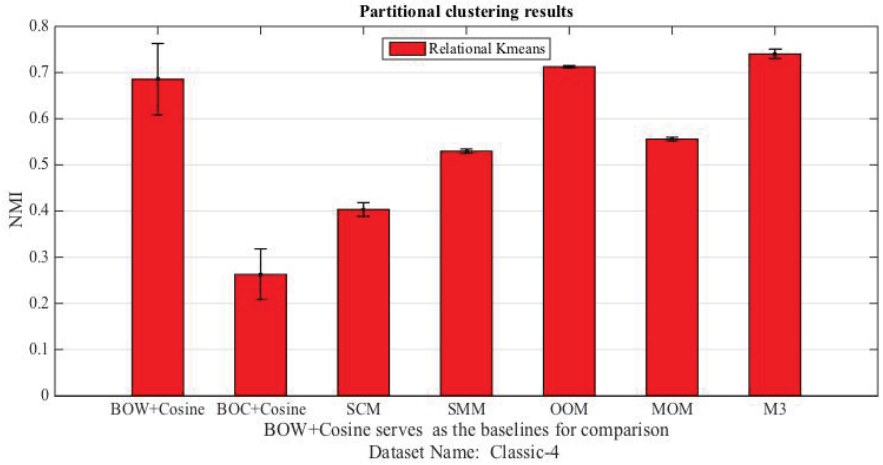


Figure C.2: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *Classic-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. On average, *M3* and *OOM* outperform the baseline.

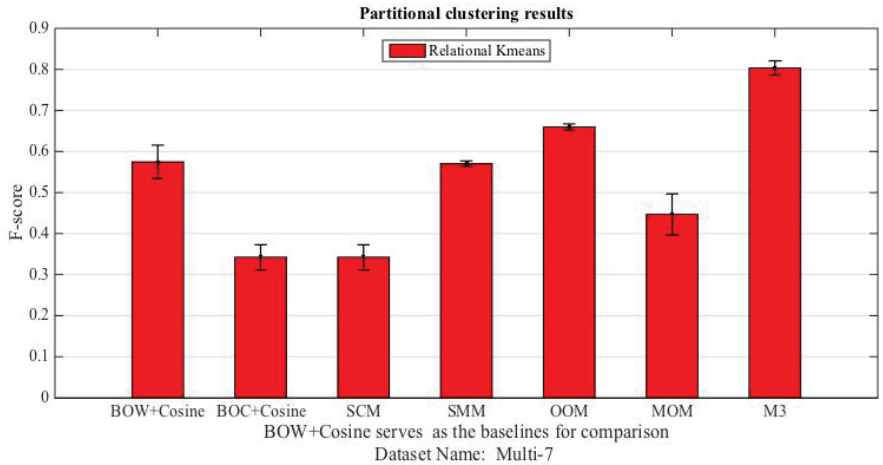


Figure C.3: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *Multi-7*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *F-score*. *M3* and *OOM* outperform the baseline.

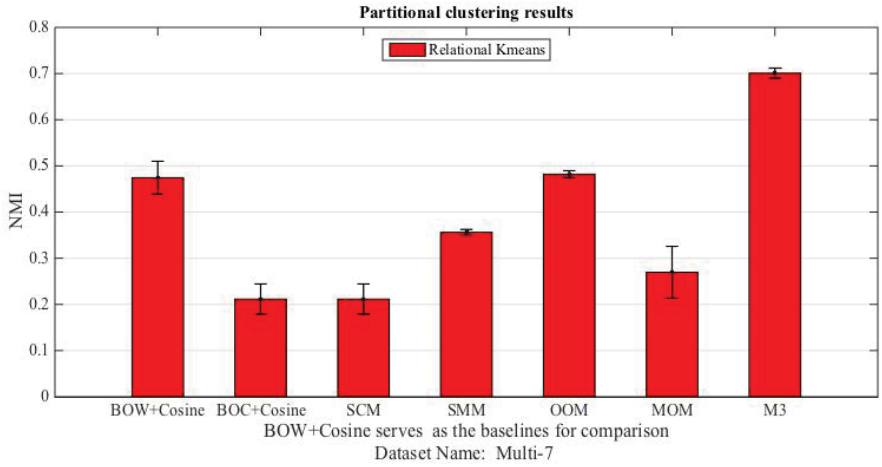


Figure C.4: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *Multi-7*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *NMI*. Only *M3* outperforms the baseline.

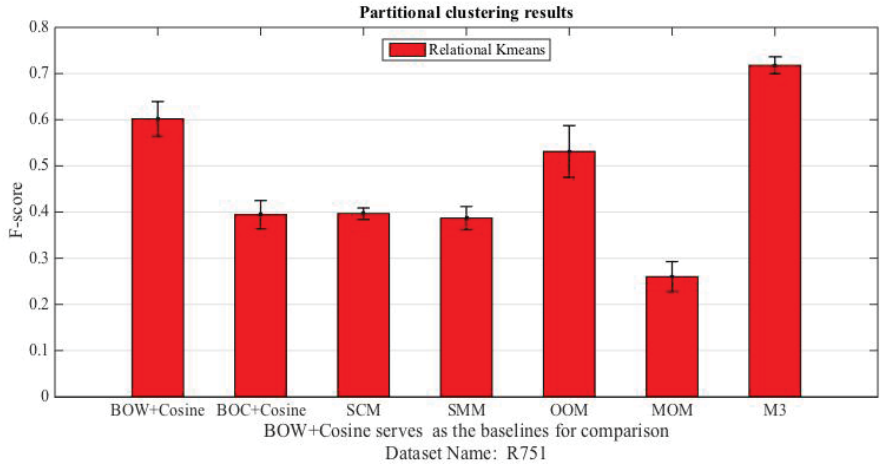


Figure C.5: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *R751*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *F-score*. Only *M3* outperforms the baseline.

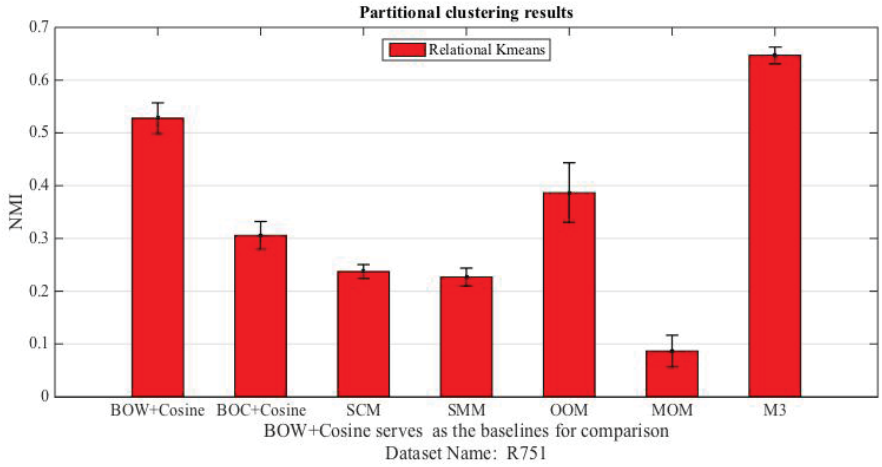


Figure C.6: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *R751*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *NMI*. Only *M3* outperforms the baseline.

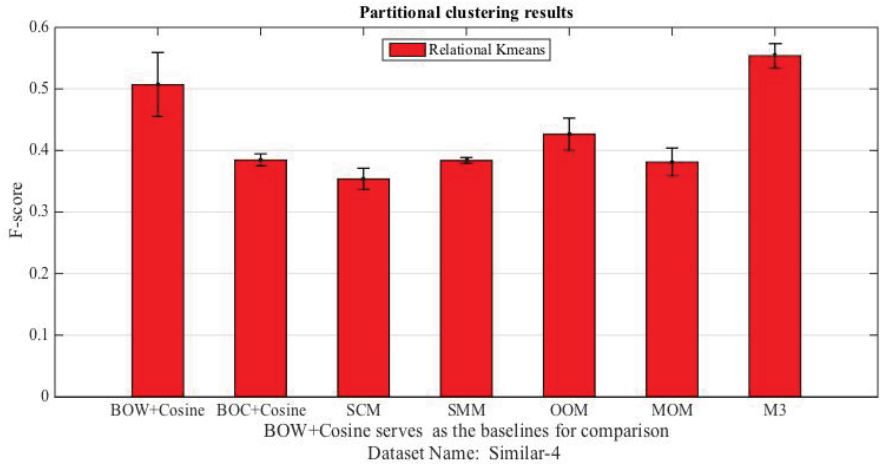


Figure C.7: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *Similar-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *F-score*. Only *M3* outperforms the baseline.

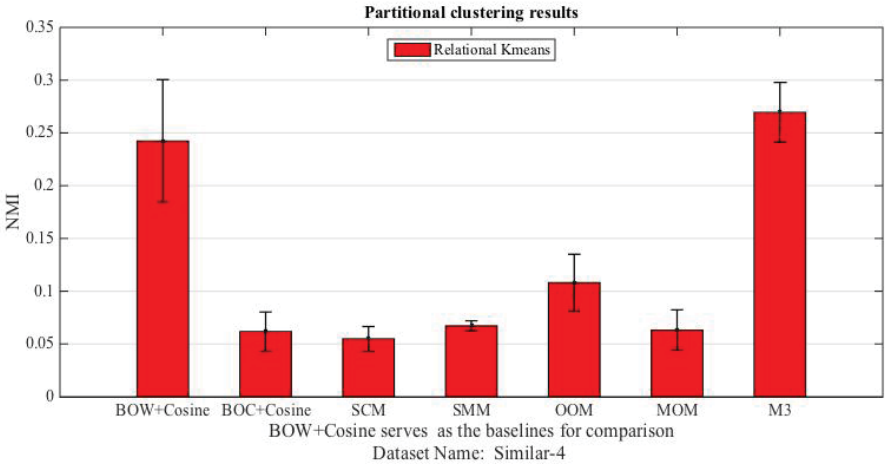


Figure C.8: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *Similar-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. On average, *M3* outperforms other similarity measures in *NMI*. Only *M3* outperforms the baseline.

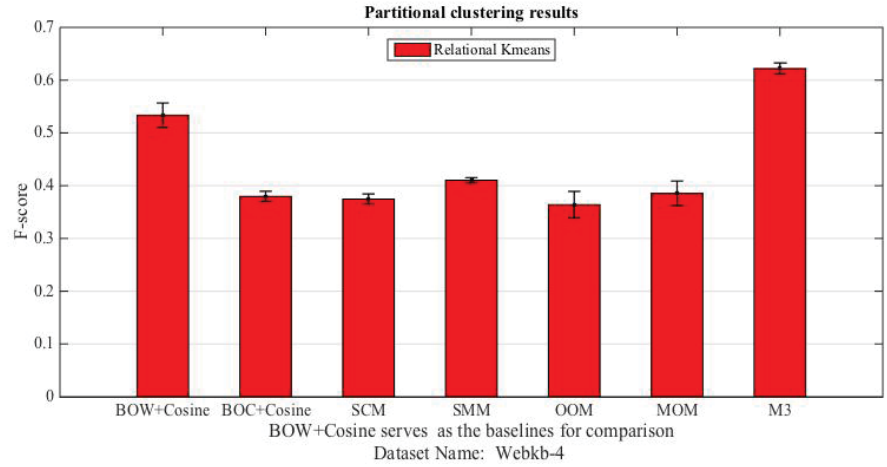


Figure C.9: The quality of clusters in terms of *F-score* obtained from Partitional clustering on *Webkb-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *F-score*. Only *M3* outperforms the baseline.

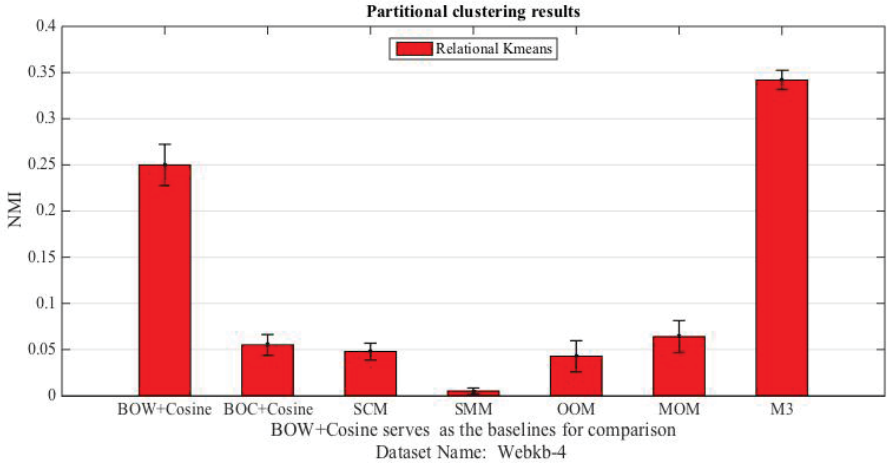


Figure C.10: The quality of clusters in terms of *NMI* obtained from Partitional clustering on *Webkb-4*. BOW with *Cosine* similarity serves as the baseline. *Cosine* and proposed similarity measures are applied on BOC. *M3* outperforms other similarity measures in *NMI*. Only *M3* outperforms the baseline.