International Journal of Multimedia and Ubiquitous Engineering
Vol.11, No.5 (2016), pp.231-244
http://dx.doi.org/10.14257/ijmue.2016.11.5.21

Efficient Document Similarity Detection Using Weighted Phrase Indexing

Papias Niyigena[1*], Zhang Zuping[1], Mansoor Ahmed Khuhro[1], Damien Hanyurwimfura[2]

[1] *School of Information Science and Engineering, Central South University, Changsha, 410083, PR China*
[2]*College of Science and Technology, University of Rwanda, Kigali, Rwanda*

**papiasni@yahoo.fr, zpzhang@csu.edu.cn, khuhro.mansoor@csu.edu.cn, hadamfr@gmail.com*

Abstract

Document similarity techniques mostly rely on single term analysis of the document in the data set. To improve the efficiency and effectiveness of the process of document similarity detection, more informative feature terms have been developed and presented by many researchers. In this paper, we present phrase weight index, which indexes documents in the data set based on important phrases. Phrasal indexing aims to reduce the ambiguity inherent to the words considered in isolation, and then improve the effectiveness in document similarity computation. The method we are presenting here in this paper inherit the term tf-idf weighting scheme in computing important phrases in the collection. It computes the weight of phrases in the document collection and according to a given threshold; the important phrases are identified and are indexed. The data dimensionality which hinders the performance of document similarity for different methods is solved by an offline index creation of important phrases for every document. The evaluation experiments indicate that the presented method is very effective on document similarity detection and its quality surpasses the traditional phrase-based approach in which the reduction of dimensionality is ignored and other methods which use single-word tf-idf.

Keywords: *Pairwise similarity, Phrase indexing, Efficiency, Document similarity algorithm*

1. Introduction

The number of electronic documents is increasing day by day due to the tremendous growth of Word Wide Web. This growth goes hand in hand with duplication of documents or portions of a document in many different locations. Documents similarity detection has become essential for applications that store or extract information from large corpora. Finding whether two documents are exact duplicates is easy because we can just compare the documents character by character, but in many applications, the documents are not identical, yet they share large portions of their text. Document Similarity Detection researches have been performed for many decades. Generally, application for document similarity detection creates an index that relates documents to the individual words present in each document [1]. These words, known as terms of the documents, are used in matching process which computes a measure of similarity between two text representations [15].

In the past, for many years, individual words have been effective for indexing terms, but intuition and experimental results suggest that phrases provide a better indication of contents of the indexed document than keywords [5]. Documents are represented by a set

ISSN: 1975-0080 IJMUE

of words, but the use of the words has some problems which can hinder the effectiveness of similarity detection. For example, the word may not have any relation to other words in the phrase, a word can be too general, different words can mean the same thing or the same word may have a different meaning. Many natural languages contain homonyms and polysemy, and then using isolated keywords for indexing takes the risk of interpreting words as unintended senses, and using phrases help to alleviate the ambiguity problems with the contextual information provided by the surrounding words [3]. In this paper, we are presenting a method which use efficiently phrases to represent the documents in the collection and then produce a pairwise similarity among these documents. Intuitively, we expect these phrases to express the subjects or main ideas of the document.

Different works using a phrase in document similarity detection have been reported, but most effort have been targeted towards single-word analysis. Zamir et al. [4] proposed a phrase-based document approach based on Suffix Tree Clustering (STC) and Suffix Tree Document (STD). The method basically involves the use of a "trie" (a compact tree) structure to represent shared suffixes between documents. It is a linear time clustering algorithm (linear in the size of the document set), which is based on identifying the phrases that are common to groups of documents. The results they showed were encouraging, but the suffix tree model could be argued to have a high number of redundancies in terms of the suffixes stored in the tree, again the STC algorithm got poor results in clustering the documents in their experimental data sets on RCV1 corpus [7]. Yamamoto and Church [6] presented a method to compute all substrings' (phrases) term frequencies and document frequencies in large document corpora by using suffix array [8]. These methods have achieved goods results, but one can argue on the data produced while computing the similarity. The creation of STD requires more nodes and more comparison which hinder the performance of the whole system.

In our work, we focus on how to extract important phrases to represent every document in the corpus and then efficiently determine the similarity among them. The phrase has been considered as more informative feature term for improving the effectiveness of document similarity detection. Constituents of phrases make other components words in the phrase less ambiguous than when words appear separately and intuitively, we expect the method using a phrase to perform better than the one using words [3].

The method presented in this paper consists of an indexing technique that use an important phrase in the document collection to reduce data dimensionality of a big number of documents which will allow our method to efficiently deduct the similarity among different documents. A phrase is an ordered sequence of two or more words in the document. The important phrase is a frequently occurring phrase in the corpus and is more informative than keyword occurrence statistics. The method presented in this paper identify phrases in a large scale corpus, index documents according to these important phrases, search and rank documents in accordance with their phrases, and provide their similarity and descriptive information about the documents.

In our method, important phrase is measured by statistical methods that emphasize a phrase discriminative power in the ad-hoc document collection compared to the global importance, instead of only looking at its local frequency. Interesting phrases are retrieved by counting the occurrence of each phrase in the collection and only the phrases with a count greater or equal to a given threshold should be maintained. We create an Index to store the phrase as the key, the occurrence of the phrase, and an inverted list containing all ID of documents containing the phrase.

The previous method has avoided the use of phrases as document feature to represent documents in the index because they noticed the computational and memory requirements to identify all possible phrases in the corpus. The method presented identifies phrases that have sufficiently frequent usage in the document collection. Once identified, the index is created for top n phrases considered as important phrases in the collection and the computation of document similarity is based on this index. This solves the problem of

indexing all possible phrases of documents in the collection which is impractical considering the size of the corpus. The creation of important phrase index implies that all documents in the collection will not be represented in the index, because some document may not have important phrases and they will be discarded in index creation; therefore, the discarded phrases allow the presented method to efficiently determine the similarity between documents because all documents will not enter in the process of computing the pairwise document similarity for the whole collection.

The index of important phrases created is used by the proposed method to efficiently determine duplicate or near duplicate document in the collection. For a given document, each sentence has a count of its occurrence in the corpus. The sentences of a document are ranked by this count and a number of the top ranking sentences are selected to form a document description. This description is stored in association with the document as a hash of sentences. To determine the similarity between documents, an incoming document is processed the same way to get its description, and then two descriptions are compared to determine the similarity among them according to the ratio of their descriptions.

2. Related Works

The use of linguistically derived phrases as indexing terms has a long history in Information Retrieval (IR). Many different kinds of linguistic phrases have been tried, with at most a modest success [8]. The predominant feeling about the value of NLP to IR, as voiced in [1], is that only 'shallow' linguistic techniques like the use of stop lists and lemmatization are of any use to IR; the rest is a question of using the right statistical techniques.

There are different works reported which use linguistics derived phrases in solving document similarity problem. Two forms of linguistic phrases are distinguished: one is statistical phrases, chunks or collocations: sequences of k non-stop words occurring consecutively, stemmed and ordered bigrams of words [8], even collocations taken from a specially prepared domain terminology. Another linguistic phrases form is syntactic phrases, identified by shallow parsing [9], template matching, and finite state techniques or by "deep" parsing. The methods presented here have achieved goods results, but one can argue on the quantity of data produced while computing the similarity with redundancies in the construction of the data structure used to facilitate the computation of similarity. The uniqueness of the proposed method is the use of similarity detection between two documents based on matching phrases; which is proved to have the significant impact on noisy terms that could hinder the quality of documents detected as similar, and the reduction of dimensionality ignored by other methods using phrases to detect similarity among documents. Phrases are less sensitive to noise when it comes to calculating document similarity; this intuition inspired us to present our method which uses statistical phrases to extract important phrases in the document collection. The statistical phrase is those words combination that co-occur in a certain context in a text corpus more frequently than expected by chance [2]. In text representation, terms are words, phrases, or any other indexing units used to identify the contents of a text. Each term in a document vector must be associated with a value (weight), which measures the importance of this term in a document [10]. The content of the document is represented as a vector in the term space, i.e., $d = (w_1, \ldots, w_k)$, where k is the term (feature) set size. Different terms have different importance in a text, thus an important indicator w_i (usually between 0 and 1) represents how much the term t_i contributes to the semantics of document d. In our method, a model is used that the document is represented as a vector of phrases (or sentences):

$$d_i = \{s_{ij} : j = 1, \ldots, p_i\} \quad (1)$$

Where *di* is document *i* in documents set *D*, s_{ij} is phrase *j* in document *I* and p_i is the number of sentences in document *i*. The phrases are composed of a vector of terms:

$$S_{ij} = \{st_{ijk} : k = 1, \dots, l_{ij}, w_{ij}\} \quad (2)$$

Where t_{ijk}: is term *k* of sentence s_{ij}, l_{ij} is the length of sentence s_{ij}, and w_{ij} is the level of the significance associated with sentence s_{ij}.

In this paper, we are interested by the weighted phrases. The collection of document has N number of documents and every document has features to represent it, we use *M* to denote all document features in the collection. To give a weight to the phrases, our method use TF-IDF weighting scheme and every document *d* can be represented as:

$$d = \{w(1,d), w(2,d), \dots, w(M,d)\} \quad (3)$$

$$w(i,d) = 0.1 + \log tf(i,d) \times \log\left(0.1 + \frac{N}{df(i)}\right) \quad (4)$$

Where $w(i,$ is i^{th} feature term weight in document *d*, $tf(i,$ is the frequency of the i^{th} term in the document *d*, and df is the number of documents containing the i^{th} term. To compute the pairwise similarity between two documents d_i and d_j the method uses Jaccard Similarity of sets which is the ration of the size of intersection of d_i and d_j to the size of their union and is given by the formula:

$$Sim(d_i, d_j) = \frac{|d_i \cap d_j|}{|d_i \cup d_j|} \quad (5)$$

Word terms have been shown to be effective for indexing, but the method in this paper present a phrase as the basic unit to identify the content in the document. The intuition and experiment results show that the phrase is more effective than *using* words or stems to identify similarity among documents.

3. Phrase-based Searching for Document Similarity

Finding document similarity now is becoming an important tool for identifying the similarity between different documents in a very growing corpus. Generally, the similarity of two documents is computed using different words present in each document. The similarity is then determined by considering the proportion of the shared words of these documents by applying one of the well-known formulas like Jaccard Similarity, Cosine similarity or any other formula. Although word has been used for long and has shown to be effective indexing terms, a recurring question in document retrieval is: what should be used as the basic unit to identify the content in documents [3]. There has been an intuition which was confirmed by experimental results suggesting that phrases provide a better indication of contents of the indexed document than keywords and offer better chances of a high quality of information retrieval [12]. The idea behind the phrase-based document similarity detection is to explore all documents in the collection and see how often the same phrases co-occur within the same documents. This method may identify in the collection how frequently certain phrase tends to occur in different documents and mark them as important which can be used to compute the similarity between documents.

In this paper, we propose a method which uses a phrase to index, rank and describe documents as described in Fig. 1. The method is designed to consider only important phrases in the document to describe the document and reduce data dimensionality. The interesting phrase is measured by statistical methods that emphasize a phrase discrimination power in ad-hoc document collection compared to the global importance, instead of looking only at its local frequency.

A similar document is likely to share more phrases. The problem of finding duplicate or near duplicate may be conceived as Market-Basket problem. Let us consider sentences to be Basket, and documents to be Items. The question to be solved is to find items which are in the same Basket most frequently. If so, it means they share the same sentence and

then are similar to some extent. If they appear in several baskets it means they are sharing more sentences. The main problem to document similarity computation is the dimensionality of the data, so the good method should have a way to deal with this problem. For the proposed method, not all the phrases in the document collection will be considered, instead every document will be represented by n phrases properly chosen according to their importance in the collection.

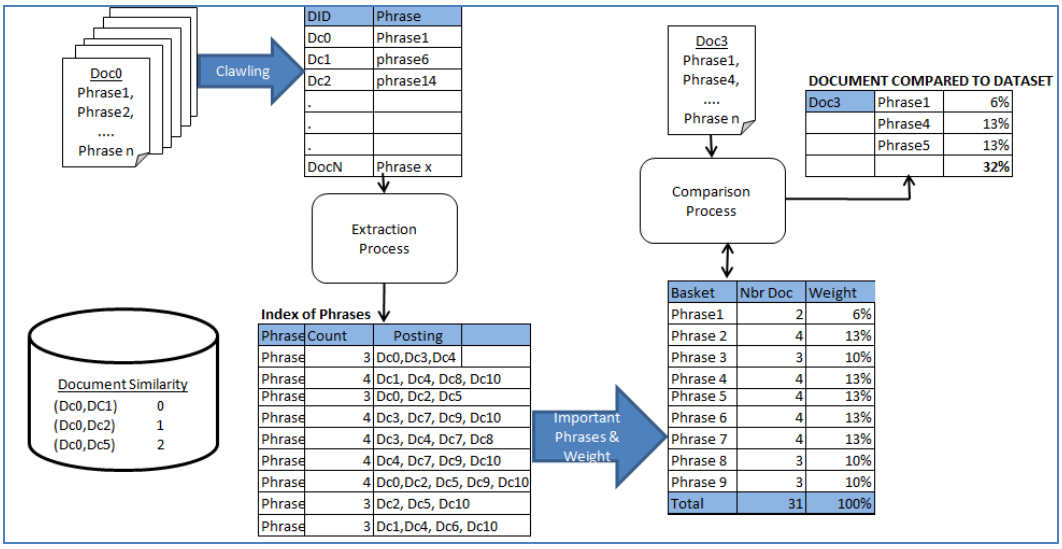

Figure 1. Overview of the Proposed Method

3.1. Efficiently Detection of Good Phrases

This paper proposes an approach that can efficiently identify important phrases for documents in a large corpus. An important phrase will be identified according to its appearance in the corpus if it is greater or equal to a given threshold. The index for important phrases is created which will provide descriptive information of documents and ease the ranking of documents according to theirs phrases. The method identifies phrases that have sufficient frequent number in the data set to be determined as a good or important phrase.

Phrases are a succession of individual terms in the documents. The length of the phrase is a parameter to be determined by the user (this parameter will have an impact on the efficiency of the method). The procedure to look for similar documents in the corpus is described as follows:

1. Preprocessing: clean the corpus by removing different terms which are not important to identify the similarity between two documents like stop words, very short terms. The corpus is then stemmed to reduce all terms to their roots.
2. Identify the good phrases in the corpus. the phrases are identified by taking all the terms in the corpus and apply the window length of the phrase, example, if we have a term T and window size of 4, the different phrases will be *T*, *T* +1, *T* +1+2, *T* +1+2+3 and *T* +1+2+3+4
3. Prune the discovered phrases to obtain only important phrases which can allow identification of document similarity
4. Compute similarity. Each phrase is represented by top *n* phrases ranked according to their appearance in the corpus. Document sharing more of these sentences are similar.

We present the algorithm to perform this action. The algorithm starts by creating the important phrase in the collections. The occurrence of a phrase in the collection is used as a measure to the weight of the phrase in the corpus. The index created for this step is

pruned to remain with an index containing important phrases which will be used to compute document similarity. Another algorithm computes pairwise document similarity in the document collection. The algorithm starts by extracting important phrase in the document to be compared. For efficiency, the algorithm discards the phrases considered as not significant for the process. The similarity is obtained by counting the shared phrases of a different document. As seen before, the index stores the document having the phrase in the posting list. The algorithm retrieves these posting lists and counts the document appearing in these lists. This count represents the similarity between two documents therefore the algorithm keeps it in the similarity index. The algorithm is described below.

Algorithm: Building Phrase Index

```
Input:    C  The corpus to be processed
          W: The window size
Output : I The index of all possible phrases in the corpus
L = EmptyList ()
I = Index()
For Each d document in C
          D1 = CleanTerms (d)
          For Each T term in D1
                    //Create the phrase according to the window size
                    Phrase = EmptyString ()
                    For I 0…W //Window size
                              Phrase = Concatenate (Phrase, T+1)
                              I.AddOrUpdate (Phrase, D1.Name, Count+1)
                    End
          End
End
//Produce the index of Important phrases to be used in similarity computing

For each K Key in the HashTable
          If K.PhraseCount< T
                    Remove(K)
          End if
End
RETURN (I): The Index of possible phrases
```

3.2. Duplicate or Near-Duplicate Document Detection

In large corpus, it is possible to have multiple instances of the same document, or several documents may be near duplicate where the documents are not identical but share large portions of their text. The proposed method provides a way to efficiently identify those portions in a different document and present the percentage of similarity between documents in the corpus. The similarity of two different documents will be expressed by a real number between 0 and 1. For this representation, 0 means that two documents in consideration are totally different while 1 or a number near to 1 means those two documents are similar or near similar. To perform this task, the proposed method rank the phrases of the document in the corpus and then select the top *N* ranked sentences to keep for every document so that they may be used for similarity detection. As we said before, the phrases are ranked according to the frequency of their appearance in the corpus.

A hash table is used to store the phrases and the related documents are kept as posting lists. The creation of this hash table is an offline operation and is stored for consultation, as it will not be necessary to recreate it in every arrival of a document. Two options are possible in computing document similarity: one is to search how documents in the collection are related to each other and secondly comparing every document to the rest of the collection. Another option is the case where we have one document and we want to know its novelty compared to the collection. If the document has a high rate of phrase similarity to the collection it implies that the idea in the document is old because it is

shared by many documents, but if the rate of similarity to the corpus is low then the document is presenting a new idea.

3.3. Pairwise similarity

The similarity of one document to the other is a very important task in Information Retrieval because it may be used to know a set of document sharing the same idea. This is mainly used to determine different clusters in the documents collection. The proposed method determines the similarity of documents by using the similarity matrix: the index is scanned once to fill the similarity matrix which will determine the similarity rate of a given document to the rest in the corpus. After filling the similarity matrix, the only document with a similarity rate less the threshold will be discarded.

Index created by off-line operation contains only important phrases for documents in the collection. Some documents will not be represented in the index if they do not have important phrases. The similarity between two different documents is determined by the number of the important phrases they have in common. The proposed method goes through the index and for every posting list it counts the number of documents appearing together in that posting list. Here we may use Map-Reduce paradigm to efficiently count the number of document sharing the sentences if we take phrases to be basket and the documents Identification as items. A hash table is created in which the key is the concatenation of two documents ID and the value keep the number of phrases those two documents have in common. The following algorithm depicts the whole idea.

Algorithm: Document similarity

Input: The Corpus Index
Output: Document Similarity matrix

```
//Create Index for important phrases

D = Documents in data set
Ford Document in D
        Pr = Get Phrases for d
        For p phrase in Pr
                If p is in Index Then
                        PriorityQueue.Add(p, Index.p)
                End If
        End For
        //Add the important phrases of document to the new Index
        PrunedIndex.Add(n first elements in PriorityQueue)
End For

//Similarity of documents: obtained by counting shared phrases
For DID DocumentId in Corpus
        PostingDoc = Index.PostingList(Phrase in DID)
        While PostingDoc is not empty
                For each el In PostingDoc
                        SimilarityMatrix[DID,el] += 1
                End For
                PostingDoc.GetNexPostingList(Pphrase in DID)
        End While
End For
```

3.4. Similarity of Document Compared to the Document Collection

When comparing the documents in a large collection, sometimes there is a need to know the content of the document compared to the collection. It is important to know how the document content is related to the collection and to what extent. If the main idea of the document is shared by several documents, the rate will be high and this means that the content of the does not present a new idea compared to the collection. On the other

hand, if the rate of the similarity of the document to the collection is low, it means that the idea in the document is new compared to the collection.

The proposed method determines this similarity using the index of the important phrases. To determine the similarity we need the weight of each phrase compared with the document collection and the total of the weight should be 1. We use the index created previously to generate the weight of each phrase which will be used to compute document similarity compared with the collection. The formula to get these weight is weight = Phrase frequency / Total number of phrases in the collection.

The proposed method gets the important phrases of the document to be compared and then use the index to get the weight of every important phrase of the document. This weight determines the similarity ratio of the document to the collection.

3.5. Comparing a Document to the Corpus

Sometimes we want to know the input of the document to the entire set of documents in the corpus. When the document shares many phrases to the corpus, it means the document contain an old idea. If only a few phrases are shared, it means the document contain a new idea.

By creating an index of important phrases, we assign each phrase a weight according to the corpus. This weight may be the rapport of the number of repetition of this phrase in the corpus by the total number of the phrases to be considered. The value will be situated between 0 and 1, and the total of the values of all phrases is equal to one. The algorithm to detect the similarity of document compared to the data set.

Algorithm: Similarity of Document to Data set

Input: *Doc the Document to check*
The Important Phrases Index
Output: The pourcentage of similarity of document to the entire data set
Sim = 0
List = GetImportantPhrases(Doc)
For *s* Phrase in *List*
Sim =*Sim* + *Index*.s(value)
End For
Return (**Sim**)

4. Experimental Results

In order to test the effectiveness and efficiency of the phrase-based document similarity, we conducted a set of experiments using our proposed data model and different similarity measures. In these experiments, we compare our method to the traditional keyword tf-idf similarity measure to evaluate our algorithm while comparing document to corpus or pairwise document similarity. In the comparison process, we use six data sets composed by different number of documents generated from the public benchmark document corpus for Text Retrieval Conferences (TREC), namely RCV1.

The document similarity calculated by our method was used to create an index which determines the pairwise similarity of all documents in the collection: the key for the index is the concatenation of two documents ID and the value is the number of phrases these documents are sharing. As seen before, the index will maintain only documents having a value greater than or equal to the threshold given to determine the similarity among documents.

In order to evaluate the quality of the presented method, we adopt *F-Measure*, a quality measure widely used in the Information Retrieval and Text Mining for the purpose of document similarity for large data sets. *F-Measure* combines *precision* and *recall* ideas in Information Retrieval. Intuitively, *recall* measures how well a retrieval method is

doing at finding all the relevant documents for a query, and precision measures how well it is doing at rejecting non-relevant documents. The *precision* and recall for sets of documents are defined as:

$$P = Precision\,(a,b) = \frac{N_{ab}}{N_a} \tag{6}$$

$$R = Recall\ (a,b) = \frac{N_{ab}}{N_b} \tag{7}$$

Where N_a is the number of relevant documents retrieved, N_b is the number of all retrieved documents and N_{ab} is the number of document retrieved which are relevant (intersection between relevant document and retrieved documents). The *F-Measure* combines both values by the following formula:

$$F - Measure = \frac{2 \times Precision\,(a,b) \times Recall\,(a,b)}{Precision\,(a,b) + Recall\,(a,b)} \tag{8}$$

4.1. Document Collection

We generate a document collection from RCV1 published by the Reuters Corporation for information retrieval research purposes [13]. From this corpus, we construct six different data sets to analyze the effectiveness of our method on different size of data sets. Figure 2 shows the impact of the phrase size to the effectiveness of the method. Intuitionally phrases should provide a better indication of the content of a document, this implies that increasing the size of the phrase should increase also the quality of document retrieved to some extent. But when the phrase becomes too large the effectiveness falls down. We conduct experiments for different phrase size as shown in Figure 2. For the different data set, we have seen that a phrase made up of six words gives a better effectiveness for our method.

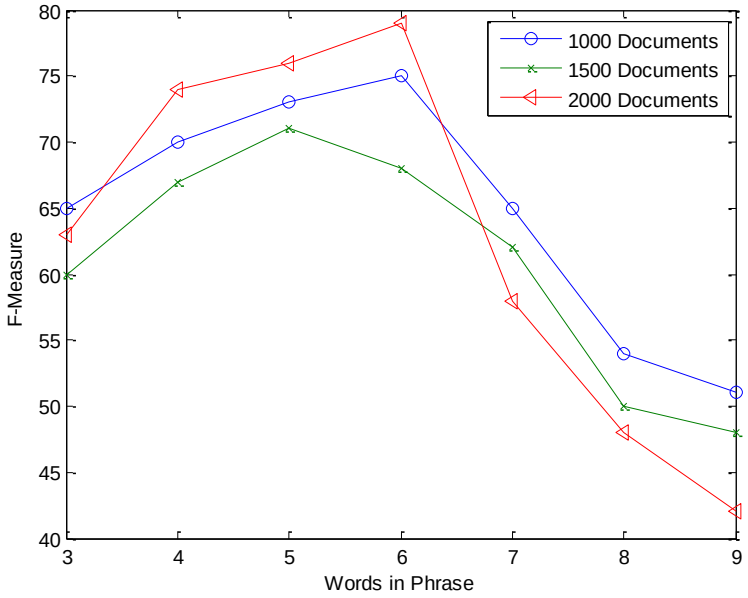

Figure 2. Comparison of F-Measure for Different Size of Data Set

We compare phrase-based document similarity detection with the traditional keyword tf-idf document similarity. Six data sets of different number of documents are used to test the effectiveness of our method while compared to tf-idf. Fig. 3 shows that for Phrase-based document similarity detection has an average increase of 16% of F-measure compared to tf-idf method using single words. The comparison results indicate that the use of phrases in computing documents similarity will not compromise the quality of similarity results. It is obvious that the phrase based similarity plays an important role in accurately judging the relation between documents.

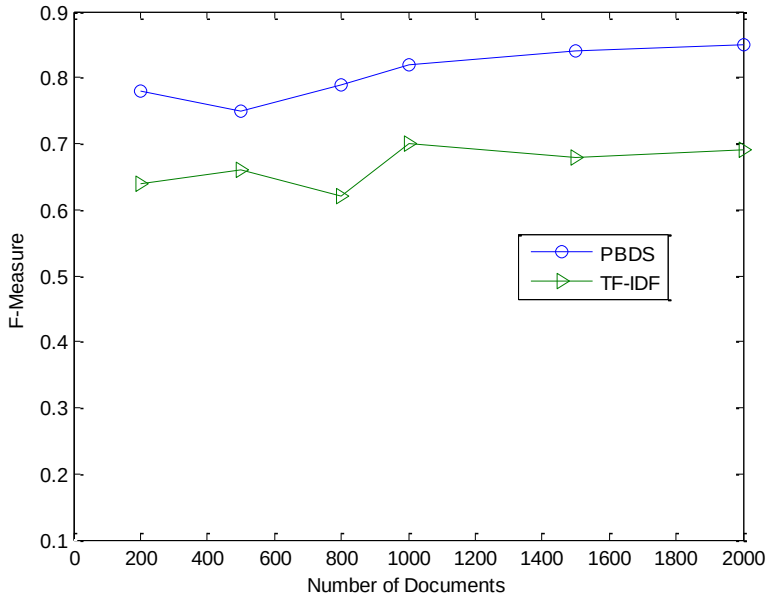

Figure 3. F-Measure comparison between PBDS and TF-IDF Method

4.2. Efficiency

To understand the run-time complexity of the proposed method, we first analyze the decomposition of our method. Our method can be separated in two main procedures: the first one is to create the index and then deduct another compressed index which will be used in document similarity, and contains only important phrases for the document collection. The second procedure is the computation of pairwise documents similarity itself.

Six data sets of a different number of documents were used for this test. We experiment the behavior of our method on different size of data set for indexing and documents comparison. We use different numbers of words for the size of a phrase (between 3 and 9) and give a different number of the document in every data set. The resulting times of test's execution are visualized in Fig. 4. One can notice that the length of a phrase has an impact on execution time; this is obvious because for the long phrase the method creates more terms to figure in the index. We recall the efficiency results where we have seen that short phrases tend to behave like single word tf-idf method. After experiments, we have seen that a phrase length of six consecutive words can provide more efficiency in reasonable time if we compare to the traditional keyword tf-idf method.

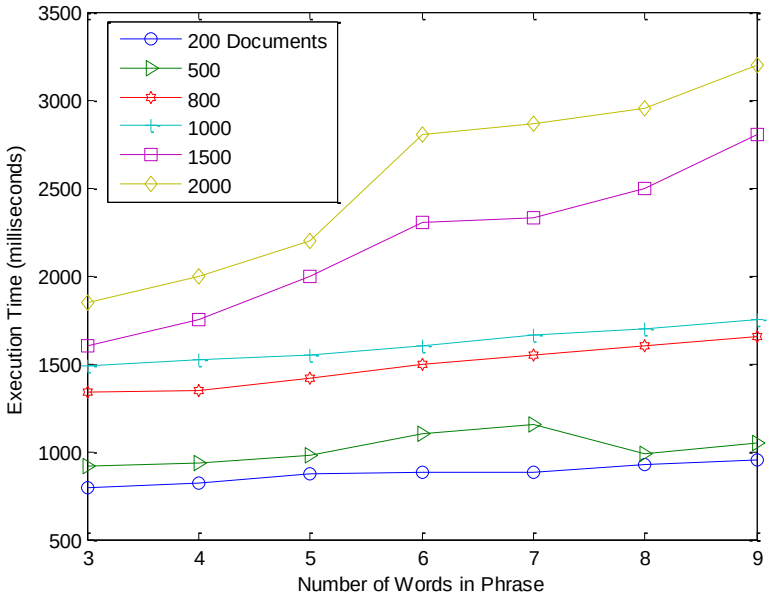

Figure 4. Time Execution Comparison on Different Size of Data Set

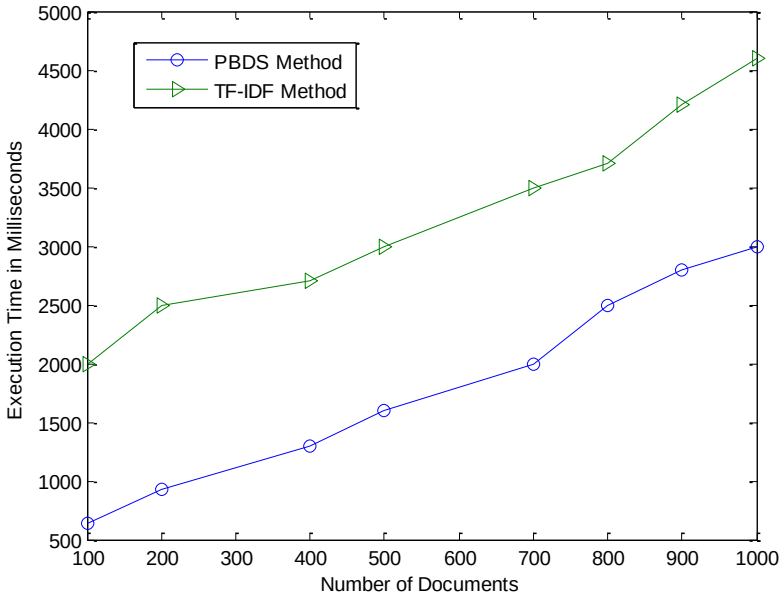

Figure 5: Execution Time Comparison between PBDS and TF-IDF Method

After the creation of index, the method performs the pairwise document similarity detection. We experiment the time used to compare one document or a group of documents against a data set of different size. As the index remains unchanged, the execution time is reduced. Fig. 5 shows the trend of execution time of our method on different size of data sets. We can notice that the execution time is almost similar for different size of data set, this is because we have cut the size of documents to remain only with n important phrases for every document. The cut has an impact on time execution because it reduces the index and also the number of phrases to be compared. With this experiment, we notice the improvement of execution time when we compare our method

to the traditional keyword tf-idf counterpart. The Fig. 5 shows that there is an improvement on time execution of our method, and the trend of tf-idf method shows that it need more time when the number of documents increases. Fig. 6 shows the execution time comparison of our method and tf-idf in general. The trend of both method show the increase on time execution when the number of documents increases, but still there is an improvement on execution time of our method compared to the traditional keyword ft-idf.

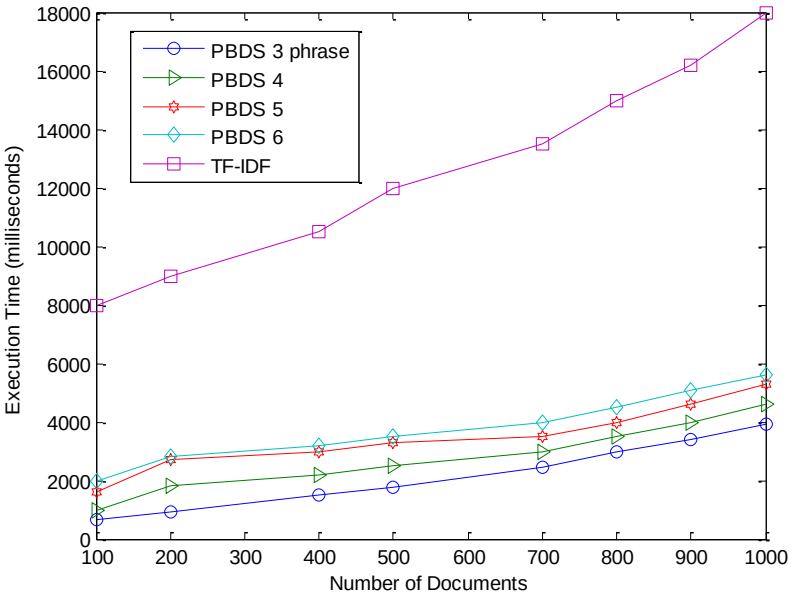

Figure 6. Execution time comparison between PBDS (with different data set and phrase length) and TF-IDF method

5. Conclusion

In this paper, we presented a Phrase-Based Document similarity in an attempt to improve the pairwise document similarity in a large document collection. To achieve more accurate results for similarity among documents in a large collection, the method presented in this paper gives another way to solve the problem of document similarity by opting to represent a document with a phrase as an important feature to represent a basic property of a document. A phrase is a more informative feature term and by intuition, document similarity accuracy should be improved by using the phrase rather than using words as document feature terms.

In this paper, we presented a method which analyze documents and identify the weights of different documents phrases and breaking down the document into its important sentences for further processing. The method creates an index of these important phrases which is used for document similarity computation. The most important component of the index created by our method that has the most impact on performance is the indexing of phrases and their levels of significance. This model enables us to perform phrase matching and similarity calculation between documents in an efficient and accurate way: once the index is created, to compare incoming document to the corpus require only extract the important phrases for the document and lookup in the index to identify its similarity to another document in the same collection. The similarity level was then defined by the weight of important phrase of the document found in the index. We have conducted different experiments and realized that the method presented in this paper effectively outperform the term-based tf-idf. The experiments

conducted show that our method have an increase of an average of 12% F-Measure if we compare to tf-idf counterpart. This is obvious because a phrase should represent better the feature term of the document than an individual word. The experimental results validate that the new method is more effective by producing more documents similar or near duplicate. Through the experiments we realized that the implementation of phrase-base document similarity is resources consuming in some cases: even though it is more accurate, its efficiency starts to degenerate when the size of the phrase becomes big. With a phrase size of six consecutive words, we have found that our method can performs better.

Acknowledgements

Project supported by the National Natural Science Foundation of China (Grant No. 61379109, M1321007) and Research Fund for the Doctoral Program of Higher Education of China (Grant No. 20120162110077).

References

[1] Cornelis H.A. Koster, Jean G. Beney, "Phrase-based Document Categorization revisited", Proceedings of the.2nd International Workshop on Patent Information retrieval, Hong Kong, China ***(2009)***, November 6.
[2] Chao-Lin Liu and Chwen-Dar Hsieh, "Exploring Phrase-Based Classification of Judicial Documents for Criminal Charges in Chinese", Proceedings of Foundation of Intelligent Systems, 16th International Symposium, Bari, Italy, **(2006)**, September 27 - 29
[3] 0. Zamir, 0. Etzioni, and 0. Madanim, and R. M. Karp, "Fast and intuitive clustering of web documents", Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining, Newport Beach, CA, **(1997)** August 14-17.
[4] Khaled M. and Harnmouda Mohamed S. Kame1, "Phrase-based Document Similarity Based on an Index Graph Model", Proceedings of IEEE International Conference on Data Mining ICDM, Vol16, no 10, **(2002)** .pp. 1279-1297
[5] M. Yamamoto and K.W. Church, "Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus", Journal of. Computational Linguistics, vol. 27, no. 1, **(2001)**, pp. 1-30.
[6] Hung Chim and Xiaotie Deng, "Efficient Phrase-Based Document Similarity for Clustering", Proceedings of IEEE Transactions on Knowledge and Data Engineering, vol. 20, no. 9, **(2008)**, pp. 1217-1229
[7] Brants T, "Natural language processing in information retrieval", Proceedings CLIN 2003, pp 1–13.
[8] Cohen W and Singer Y, "Context sensitive learning methods for text categorization", Journal of ACM Transactions on Information Systems (TOIS), Vol. 17 no. 2, **(1999)**, pp. 307–315.
[9] Man Lan and Chew Lim Tan and Jian Su, Yue Lu, "Supervised and Traditional Term Weighting Methods for Automatic Text Categorization", Journal of IEEE Transaction on Pattern Analysis and Machine Intelligence, vol. 31 no. 4, (**2009)**, pp. 721-735.
[10] Qinglin Guo, "The similarity computing of documents based on VSM", Proceedings of 2nd International Conference on Network Based Information System, Berlin, Germany **(2008)**, July 28 – August 1.
[11] Frank, E. and Paynter, G.W Witten, I.H., "Domain-Specific Keyphrase Extraction", Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, Stockholm, Sueden, **(1999)**, July 31 – August 6.
[12] Yang, Ruilong Zhu, Qingsheng Xia, Yunni, "A novel weighted phrase-based similarity for Web documents clustering", Journal of Software, vol. 6 **(2011)**, pp. 1521-1528.
[13] Gao Chuancong and Michel, Sebastian, "Top-k Interesting Phrase Mining in Ad-hoc Collections using Sequence Pattern Indexing", In Proceedings of ACM 15th International Conference on Extending Database Technology, Berlin, Germany (**2012)**, March 26 - 30.
[14] Dariusz Ceglarek and Linearithmic "Corpus to Corpus Comparison by Sentence Hashing Algorithm SHAPD2", In Proceedings of The Fifth International Conference on Advanced Cognitive Technologies and Applications, Valencia Spain, **(2013)**, May 27 – June 1.
[15] Jialu Liu and Jingbo Shang and Chi Wang, Xiang Ren and Jiawei Han, "Mining Quality Phrases from Massive Text Corpora", In Proceedings of the 2nd CCF Conference on Natural Language Processing and Chinese Computing, NLPCC, Melbourne, Australia (**2015**), May 31 – June 4.
[16] ElKishky, Ahmed , Song, Yanglei, Wangx, Chi Voss, Clare R. Han, Jiawei, "Scalable topical phrase mining from text corpora", Journal of VLDB Endowment, vol. 8, no. 3, (**2014)**, pp. 305-316.

[17] K.M Hammouda and Mohamed S Kamel, "Efficient phrase-based document indexing for Web document clustering", Journal of IEEE Transactions on Knowledge and Data Engineering, vol. 16 no. 10, **(2004)**, pp. 1279-1296.
[18] Bedathur and Srikanta, Berberich and Klaus Dittrich and Jens Mamoulis and Nikos Weikum, Gerhard, "Interesting-phrase mining for ad-hoc text analytics", Journal of VLDB Endowment, vol. 3 no. 1 **(2010)**, pp. 1348-1357.
[19] Kunmei Wen and Ruixuan Li and Jing Xia nd Xiwu Gu. "Optimizing ranking method using social annotations based on language model", Journal of Artificial Intelligence Review vol. 41 (**2014**), pp. 81-96.
[20] Li, Sheng and Xian, Hu. "Document similarity calculation based on words' semantic information", Journal of Computational Information Systems. Vol 10 **(2014)**, pp. 9555-9564.

