Text Clustering Using a Suffix Tree Similarity Measure

Chenghui HUANG\textsuperscript{1,2}
\textsuperscript{1} School of Information Science and Technology, Sun Yat-sen University, Guangzhou, P.R.China
\textsuperscript{2} Department of Computer Science and Technology, Guangdong University of Finance, Guangzhou, P.R.China
Email:hch.gduf@163.com

Jian YIN\textsuperscript{*}
School of Information Science and Technology, Sun Yat-sen University, Guangzhou, P.R.China
*correspondence author: issjyin@mail.sysu.edu.cn

Fang HOU
Department of Computer Science and Technology, Guangdong University of Finance, Guangzhou, P.R.China
Email: hfhoufang@126.com

Abstract—In text mining area, popular methods use the bag-of-words models, which represent a document as a vector. These methods ignored the word sequence information, and the good clustering result limited to some special domains. This paper proposes a new similarity measure based on suffix tree model of text documents. It analyzes the word sequence information, and then computes the similarity between the text documents of corpus by applying a suffix tree similarity that combines with TF-IDF weighting method. Experimental results on standard document benchmark corpus RUTERS and BBC indicate that the new text similarity measure is effective. Comparing with the results of the other two frequent word sequence based methods, our proposed method achieves an improvement of about 15% on the average of F-Measure score.

Index Terms—clustering algorithm, suffix tree, document model, similarity measure

I. INTRODUCTION

With information era coming, almost all the traditional papery documents are undergoing a transition to electronic document gradually. This transition is irreversible because electronic documents are safer and easier to save and use than papery documents. Every organization has a database, which contains huge volumes electronic documents. The Word Wide Web is such a database, and how to search and utilize this kind of text databases is a hot research topic. These requirements stimulate us to develop corresponding methods to help the users to browse and organize these electronic documents more effective. The ultimate goal is to help user to get what he or she needs from vast information ocean.

Text clustering is known as an unsupervised and automatically procedure of grouping text documents, so those similar documents will be grouped into a cluster, but not those dissimilar. Therefore, how to define a more accuracy text similarity to group text documents is critical to the urgent requirement with the current information society. There are extensive literatures on measuring the similarity between the texts [2][5]. The TF-IDF [6] (Term Frequency-Inverted Document Frequency) model is a popular representation model of text documents. The similarity between two text documents is computed with one of several similarity measures based on two vectors, e.g. cosine, Jaccard, and Euclidean distance measure. The major drawbacks of the TF-IDF method are as follows: Firstly, data analysis becomes more difficult with the increasing of dimensions while using the traditional word frequency analyzing methods. Secondly, this method ignores word sequences information of documents. In order to understand document more accurately, developing a similarity measure that contains features, which are more informative, has received considerable attention recently.

The similarity among text documents has many applications in natural language processing, information retrieval, and text mining. For example, in web page retrieval of search engine, text similarity has been proved one of the best techniques for improving retrieval results of a search engine [1]. Furthermore, search engines use text retrieval technology to rank search result documents according to similarity among documents from high to low. The use of text similarity also can be contribution to text categorization [2], duplicate detection in web pages [3], document summarization [4], etc.

The motivation of adopting suffix tree model for document clustering can be attributed to two aspects. The first one is the demand of dimensionality reduction for text model. In bag-of-words model, it always has a huge dimensionality, and inevitably results sparse vectors of text documents. The second one is that suffixes of document can include more information than word frequency. A frequent suffix is a set of individual words that includes more conceptual and contextual meanings than an individual word. To address these arguments, this paper attempts to propose a text similarity measure based on suffix tree model, which get better effectiveness with
word sequence information in documents comparing to traditional word frequency method.

The remainder of this paper is organized as follows. Section 2 presents a brief overview of the related work. Our proposed method is described in Section 3. Evaluation and experimental results are discussed in Section 4. Finally, the conclusion is given in Section 5.

II. RELATED WORKS

There are two general categories of text similarity methods: word frequency/vector-based document model methods and frequent word sequence based document model.

A. TF-IDF Model

TF-IDF is a method that weights the frequency of a term in a document with a factor that discounts its importance for its appearance in corpus. To illustrate, figure 1 (adapted from [7].) shows a set of example documents, with the corresponding term-document matrix representation. In the matrix, the rows correspond to the keywords, while the columns represent the documents. The nonzero cells indicate clearly what terms appear in each document. At runtime, each text document is transformed into the same representation, in order to use cosine or Jaccard coefficient similarity measures to compute the text similarity.

The TF-IDF model ignores words sequence and structure of documents. Furthermore, amount of words and documents are always huge for most of document databases. With TF-IDF model, we must process a vector set, which has amount of vectors and each vector has a dimension amount to words number, and therefore inevitably leads to a lower efficient computing.

Although TF-IDF method has been proved to be good in practice, the massive document data require accurate description rather than just term frequency. The concept of the frequent word set is based on the frequent item set of the transaction data set. The sequential order of words in a document plays a key role of delivering the meaning of the document. The change of the relative positions of two words may change the content of a document. For example, “association rule” is an important concept of data mining. If these two words, “association” and “rule”, appear in a reverse order within a document, like “The rule of their association is ...” that represents a very different meaning. However, both of these documents will be equal to each other by using TF-IDF method.

B. Frequent Word Sequence Document Model

For the reason of that document represented by frequent word sequences can get better effect than frequent words, [8][9] analyzed frequent word sequences, and proposed a suffix-tree method for text clustering. The suffix tree serves document as a string rather than a bag of words. It analyzed sharing document fragment by creating a suffix tree. However, this method did not reduce dimension of document model and did not consider semantic information of frequent words in documents. [10] further investigated and discussed frequent word sequences based on suffix tree, and modeled documents as a model of frequent word sequences, which reduced dimensions of document model effectively. [11] overcame the drawbacks of the above research works had that two terms can have the same frequency in their documents, but one term contributes more to the meaning of its sentences than the other term. It analyzed verb structures of sentence and combined traditional TF-IDF technology, and given different concepts with different weighting. However, a problem of these algorithms is that none of them defined a practical similarity to measure the similarity of query and documents. [12] proposed a suffix tree similarity model for clustering algorithm, nevertheless, their method did not apply pruning strategy to decrease the dimension of suffix tree model.

[13] proposed Maximal Frequent Sequences (MFS) used for text clustering. A frequent sequence is maximal if it is not a subsequence of any other frequent sequence. The main idea of MFS is to use maximal frequent sequences of documents as text features in vector space model and then employed k-means to cluster documents. MFS is a method that converts word frequency to frequent word sequence in document clustering. Its performance depends on the effectiveness of using MFS for document representation in clustering, and the effectiveness of k-means.

[14] proposed another Frequent Term-Based Clustering (FTC) method for document clustering. The motivation of FTC is to produce document clusters that overlap between clusters as few as possible. FTC starts with an empty set, and it continues selecting one cluster description from the set of remaining frequent word sequences until all the frequent word sequences are chosen. In each step, FTC selects one of the remaining
frequent word sequences that have the smallest entropy overlap (EO) value. In FTC, a cluster candidate is represented by a frequent word sequence and each candidate’s EO is calculated. Consequently, FTC tends to select cluster candidate as a document cluster, of which its number of documents is small while occurrence frequencies of these documents are large. However, it will cause large amount clusters with only several documents, often a cluster has one isolated document.

Our approach is different from those methods in several ways: First, we use natural language processing tool to prepare text documents such as removing stop word and recognizing named entity etc. Next, we compute the TF-IDF value of each distinct word in a document and select those words that have high TF-IDF values. In the meantime, we keep the original word sequence of text documents. Consequently, we represent query and each document as different suffix trees and then compute similarity of query and documents with proposed suffix tree similarity measure. When we get similarity of query and documents, we rank the retrieved documents according to their similarity with query from high to low. This method just chooses those terms that have high TF-IDF values, therefore, it can reduce dimension of traditional document model and suffix tree model effectively. At the same time, it analyzes word sequence information of query and documents, and then makes text retrieval more effective. Experimental results on standard data sets demonstrate that it can promote F-Measure comparing to above frequent word sequences based methods.

III. A NEW SUFFIX TREE SIMILARITY MEASURE

In this section, we shall first introduce our motivation and review suffix tree document model. Next, we shall explain the definition of new suffix tree similarity measure and the proposed Suffix Tree Similarity Measure (STSM) algorithm in details.

A. Motivation

Our motivation of proposing suffix tree similarity measure for document clustering is to produce natural and comprehensible document clusters. For this purpose, we use frequent word sequences for representation and measure similarities of documents based on suffix tree model of frequent word sequences in documents. Moreover, there are two distinct characteristics attracting us to study suffix tree document model.

Firstly suffix tree document model proposed a new flexible n-grams approach to identify all overlap word sequences among the documents as longest common prefixes. Secondly, one or several phrases are naturally selected to generate a cluster description to summary the corresponding cluster while building the clusters. After studying the suffix tree clustering algorithm with the description of (Zamir et al., 1997), by using the suffix tree clustering algorithm we can obtain quite good results in clustering standard documents as well as document snippets. However, the suffix tree clustering algorithm sometimes generates some large-sized clusters with poor quality of clustering standard documents.

The suffix tree model of document contributes more meaning to document information and contains more semantics than individual terms, so they will improve the accuracy of similarity measure of documents, and as a result, the quality of document clusters will be improved.

B. Suffix Tree Document Model

In text mining, a document model is that how we can extract a set of meaningful features from a document and represent these features. Suffix tree document model considers a document d = w1w2...wm as a string consisting of words wi(i = 1; 2;:::;m), not characters. A suffix tree of document d is a compact trie tree containing all suffixes of document d. Different from common suffix tree model of documents we construct a suffix tree for each document of corpus other than a suffix tree for the whole corpus. Figure 2 is an example of three suffix trees composed from three documents.

The nodes of the suffix tree are drawn in circles. Each edge is labeled with a non-empty word sequence of a document called a phrase, and its suffix node is labeled by the phrase too. Each leaf node in the suffix tree designates a word sequence of a document, and each internal node represents a phrase shared by at least two suffixes. The more internal nodes shared by two paths, the more important the word sequences tend to be. In Figure 2, each node except root node is attached with a box respectively. There are two numbers in the box. The first number in the box designate how many times the word sequence appeared in the current document, and the second number designate how many documents in the corpus contained the word sequence.

C. Suffix Tree Similarity Measure (STSM)

Our adapted suffix tree similarity measure is developed based on the suffix tree model. In detail, STSM includes following several steps.

![Figure 2. The suffix tree of three documents. D1: “Cat ate cheese”. D2: “mouse ate cheese too.” D3: “Cat ate mouse too” (The word “too” as stop word was removed during text preprocessing procedure.).]
Step 1. Preprocessing documents of corpus.
In this step, we preprocess text documents such as removing stop-words, using stemmer to find stemmed word, and recognizing named entity, etc.

Step 2. Weighting word sequence of each document by using TF-IDF method.
TF-IDF is a widely used information retrieval technique for weighting the importance of individual word terms appearing in all documents. Traditional TF-IDF method represented text document as a word frequency vector, which include all distinct words in documents. However, the word frequency vector will have a negative impact on the construction of the suffix tree model consequently because there are no recurrences of word sequences. In order to remove the negative effect of traditional TF-IDF methods, we keep the original word sequence of each document and do not remove those words reappearing in a document when computing TF-IDF values of the words. Then we ignore those words whose TF-IDF values is less than a threshold \( \mu \) (See section 4) and construct a suffix tree model for each document, and transform each document into a word sequence vector in the suffix tree node dimensional space consequently. The suffix tree set \( S= \{S_1; S_2; \ldots; S_n\} \) for each document in corpus \( C = \{d_1; d_2; \ldots; d_m\} \) will be constructed. Finally, we use TF-IDF method again to weight the importance of word sequences in a document, which are actually nodes of suffix tree.

By mapping all nodes of the common suffix tree to a \( M \) dimensional space of VSD model (\( n = 1; 2; \ldots; M \)), each document \( d \) can be represented as a feature vector of the weights of \( M \) nodes,

\[
d = \{w(1; d); w(2; d); \ldots; w(M; d)\}
\]

It is easy to understand that the document frequency of each node \( df(n) \) is the number of the total times of node \( n \) appeared in all documents. The term frequency \( tf(n; d) \) of a node \( n \) with respect to a document \( d \) is the number of how many times the word sequence, i.e., node \( n \), appears in current document \( d \).

For example in Figure 2, the \( df \) of node \( b \) is \( df(b) = 2 \), the \( tf \) of the node \( b \) with respect to document \( 1 \) is \( tf(b; 1) = 1 \) (assuming the document identifiers of the three documents to be 1, 2, 3). Therefore we can calculate the weight \( w(n; d) \) of node \( n \) in document \( d \) using the following TF-IDF formula:

\[
\text{TF-IDF}(n; d) = (1 + \log(tf(n; d))) \cdot \log(1 + N/df(n))
\]

(2)

Step 3. Constructing suffix tree for user’s query.
A suffix tree \( QT \) for user’s query will be constructed in this step.

Step 4. Computing the similarity among documents in corpus.
In order to compute the similarity among documents in corpus, we should clarify some key concepts in the domain of suffix tree model.

The similarity between two documents can be described as the similarity of two suffix trees, which represent two documents respectively. However, the dimension of document model will rise considerably after we transformed the model from word frequency to suffix tree, so that we have to find a way to prune the dimension of suffix tree model. In our similarity method, we use a new concept “stop node”, which applies the same idea of stop words in the traditional text preprocessing procedure. A node with a high document frequency \( df \) can be pruned in the similarity measure. In this paper, those nodes occurred in more than half of documents in corpus will be pruned.

With above observations, we give the definition of similarity between two suffix trees as the following.

**Suffix of a word string:** Let a document \( D = (d_1, d_2, \ldots, d_m) \) and another document \( T = (t_1, t_2, \ldots, t_n) \), where \( d_i \) and \( t_j \) is a word respectively. Then the suffix \( S(d_i) \) is defined by \( d_{i1}, d_{i2}, \ldots, d_{in} \). \( S(t_j) \) is \( t_{j1}, t_{j2}, \ldots, t_{jn} \).

**Definition 1:** Let \( SD = \{S(d_1), S(d_2), \ldots, S(d_m)\} \) be the suffix set of a document \( D \), \( ST = \{S(t_1), S(t_2), \ldots, S(t_n)\} \) the suffix set of another document \( T \). Then the similarity \( \text{Sim}(D, T) \) between \( D \) and \( T \) can be defined as:

\[
\text{Sim}(D, T) = \sum_{i=1}^{m} \sum_{j=1}^{n} \text{Sim}(S(d_i), S(t_j))
\]

where \( S(q) \in SD, S(w) \in ST \).

In (3), \( \text{Sim}(S(d_i), S(t_j)) \) denotes the similarity of two suffixes, and it can be calculated according the algorithm of Suffix-Tree Similarity Measure (STSM).

After obtaining the term weights of all suffixes, it is easy to apply traditional similarity measures, such as, the cosine similarity to compute the similarity of two documents. In this paper, we use cosine similarity measure to compute the pair-wise similarities of all documents. The Direct K-Means (DKM), Bisection K-Means (BKM), and Agglomerate K-Means (AKM) are used to evaluate the effectiveness of the new suffix tree similarity measure in document clustering.

Algorithm STSM:
Input: two suffix tree set: \( SD, ST, TF-IDF \) threshold \( \mu \)
Output: similarity of two documents represented by suffix tree \( SD, ST \)
for each suffix \( S(di) \) in \( SD \), match suffix \( S(tj) \) in \( ST \) if \( S(di) = S(tj) \) and TF-IDF(\( S(di) \)) \( \geq \mu \) then
\[
\text{Sim}(S(di), S(tj)) = \text{TF-IDF}(S(di)) + \text{TF-IDF}(S(tj))
\]
else
\[
\text{Sim}(S(di), S(tj)) = 0;
\]
end for
return \( \text{Sim}(S(di), S(tj)) \);

Figure 3. The algorithm of Suffix-Tree Similarity Measure (STSM).

IV. EXPERIMENTS
Two public text data sets Reuters-21578 [15] and BBC dataset [16] are used for experiments. These data sets are different in the number of documents, the number of clusters and the distribution of document. We constructed 6 document test sets Re1, Re2, Re3 from Reuters-21578
and BBC1, BBC2, BBC3 from BBC, respectively. Every document already has one or more certain document type, which is used as reference to test the clustering effect. Table I summarizes the characteristics of all the data sets.

We use NLP tools LingPipe [17] to preprocess documents tasks such as stop-words removal and stemmer, and calculate document similarity by using STSM algorithm to get a document similarity matrix. With the similarity matrix, we cluster documents by using clustering algorithm tool kit CLUTO [18].

We choose F-Measure to verify the validity of our document similarity and compare other algorithms. The F-Measure is an index that balances precision and recall in information retrieval. After clustering, we check whether each document is divided into the correct class or whether the class includes documents of specific types. We calculate precision $P(i,j)$ of class $i$ that cluster $j$ belongs to and recall $R(i,j)$ of class $i$ that cluster $j$ belongs to.

Let $n_i$ be the number of documents in class $i$, $n_j$ be the number of document in cluster $j$, and $n_{ij}$ be the number of documents of class $i$ in class $j$, then $P(i,j)$ and $R(i,j)$ can be defined as:

$$P(i,j) = \frac{n_{ij}}{n_j}, \quad R(i,j) = \frac{n_{ij}}{n_i}$$

The corresponding F-Measure is defined as:

$$F(i,j) = \frac{2 \times P(i,j) \times R(i,j)}{P(i,j) + R(i,j)}$$

The F-Measure value for the whole clustering result is defined as:

$$F = \sum_i \frac{n_i}{n} \max(F(i,j))$$

where $n$ is the total number of document in the corpus. In general, the larger F-Measure, the better clustering results.

We conduct the experiments that survey the effectiveness by varying the TF-IDF threshold $\mu$ among the DKM, BKM, and AKM. Figure 4 (a), (b), and (c) present the F-Measure scores of the three clustering algorithms with different TF-IDF threshold $\mu$. The best result is obtained from Re3 document set under DKM algorithm, which is the most widely diverse data set in our document collection, and the F-Measure reaches the top score in all of three clustering algorithms while $\mu$ is set to three. It roughly equivalent to an internal node of suffix tree would appear in a document for more than two times and appear in about 50% of the documents in the corpus.

Figure 5 shows the F-Measure scores computed from the clustering results of five clustering algorithms on 6 document sets, where $\mu$ is set to be 3 and MFS designates the results of the maximal frequent sequences method. FTC designates the results of frequent term-based
clustering method; STSM-DKM designates the results of STSM algorithm with DKM clustering algorithm; STSM-BKM designates the results of STSM algorithm with BK clustering algorithm; and STSM-AKM designates the results of STSM algorithm with AKM clustering algorithm. Comparing with the results of MFS and FTC, STSM algorithm has a performance improvement of 15% on the average F-Measure scores of 6 document sets.

V. CONCLUSION

Both traditional word frequency vector space model and suffix tree model play key roles in text mining. However, they are used in isolated way: almost all clustering algorithms based on bag-of-words model ignore the word sequences in the document, and these methods discard the different semantic meanings of a word in different sentences. Suffix tree document model keeps all sequential characteristics of the sentences for each document, phrases consisting of one or more words are used to designate the similarity of two documents. However, the original suffix tree model algorithm increases the dimension of text document representation dramatically. It leads to an inefficient computation. This paper proposes a new suffix tree similarity measure that combines TF-IDF with suffix tree model.

Next STSM is proposed to evaluate document similarity and use to cluster documents. It includes several procedures: constructing suffix tree model for each document clusters, using TF-IDF to prune those words with low TF-IDF value, and calculating the similarity between two documents based on their suffix trees. On constructing suffix tree model of document, we use TF-IDF to reduce the dimension of suffix tree model. Furthermore, we pruned those nodes of suffix tree with low TF-IDF values. As a key to text document similarity, our STSM is succinct using TF-IDF to capture those most informative word sequences.

Finally, we conduct experiments to evaluate STSM in comparison with MFS and FTC. Two English corpuses are used in the experiments. The improvement of the clustering performance in our experiments clearly indicates that our similarity measure is superior to other word frequency based methods, and the new similarity measure is suitable for several clustering methods. With the above analysis, we conclude that STSM has favorable quality in clustering documents using frequent word sequences.

In future, we will further analyze semantic similarity features of documents, syntax and semantic structure of phrases and sentences for better effectiveness.

ACKNOWLEDGMENT

The authors would like to thank an anonymous reviewer for his/her useful comments that have helped to improve the quality of the manuscript. This work is supported by the National Natural Science Foundation of China (60573097, 60773198, 60703111), Program for New Century Excellent Talents in University of China (NCET-06-0727).

REFERENCES


Chenghui HUANG was born in 1976, and received the M.Eng. degree in software and theory of computer from Jinan University, Guangzhou, in 2005, and is a lecture at Guangdong University of Finance since 2005. He is a Ph. D candidate of SUN Yat-Sen University. His research interests include Text Mining, Information Retrieval, and Natural Language Processing.

Jian YIN was born in 1968, and is a professor at SUN YAT-SEN University since 2004, where he leads and participates in several national and provincial projects in areas of Data Mining. His main interests are focused on Information Retrieval, Machine Learning, and Data Mining.

Fang HOU received the M.Eng. degree in automatic control from Shanghai Maritime University in 2004. He is a PhD. candidate in Computer Science and Engineering School of South China University of Technology, Guangzhou, China. His research interests include computer architecture and storage system.