Document Clustering with Concept Based Vector Suffix Tree Document Model

Dr. N. Sandhya  
Professor, VNRVJIET  
Email: sandhyanadela@gmail.com

J Sireesha  
Assoc. Prof., MREC(A),  
Email: sireeshajasti@yahoo.co.in

Subir Kumar Sarkar  
Dept. of ETCE, Jadavpur Kolkata, India University

Abstract – This work aims to extend a document representation model which is elegant by combining the versatility of the vector space model, the increased relevance of the suffix tree document model, which takes word ordering into account and retaining the relationship between words like synonyms. The effectiveness and the relevance of this document model can be evaluated by a partitioning clustering technique K-Means and then a systematic comparative study of the impact of similarity measures in conjunction with different types of vector space representation on cluster quality may be performed. This document model will be called the concept based vector suffix tree document model.

Keywords – Wireless Network, Failures, Checkpoints, Rollback Recovery.

I. INTRODUCTION

The most popular way for representing documents is the VSM, because of its speed and versatility. TF-IDF score can be computed easily, so this model is fast. In the worst case the document needs to be traversed twice for computing document frequencies and TF-IDF values. To overcome the bag of words problems, text documents are treated as sequence of words and documents are retrieved based on sharing of frequent word sequences from text databases. The sequential relationship between the words and documents is preserved using suffix tree data structure [1]. Syntax based disambiguation is attempted by enriching the text document representations by background knowledge provided in a core ontology-WordNet.

This paper is organized as follows. The section 2 deals with the related work in text document clustering, section 3 describes the preprocessing, POS tagging and the use of WordNet for replacing words with their Synset IDs. Section 4 and Section 5 discusses the document representation, similarity measures and their semantics. Section 6 describes the steps in the CBVSTDM.

II. RELATED WORK

An approach that combines suffix trees with the vector space model was proposed earlier [2] [3]. They use the same Suffix Tree Document Model proposed by Zamir and Etzioni but they map the nodes from the common suffix tree to a M dimensional space in the Vector Space Model. Thus, a feature vector containing the weights of each node can be used to represent the documents. This model can be extended by attempting the word sense disambiguation using WordNet. Once the vector of weights is obtained, similarity measures and K-Means clustering algorithm with the Vector Space Model are applied. The lack of an effective measure for the quality of clusters which is an important problem with the original suffix tree is overcome with this model.

III. DOCUMENTS PREPROCESSING USING WORDNET

Initially we need to preprocess the documents. This step is imperative. The next step is to analyze the prepared data and divide it into clusters using clustering algorithm. The effectiveness of clustering is improved by adding the Part-Of-Speech Tagging and making use of the WordNet for synsets. The most important procedure in the preprocessing of documents is to enrich the term vectors with concepts from the core ontology. WordNet [7] covers semantic and lexical relations between word forms and word meanings. The first preprocessing step is POS tagging. POS tagging is a process of assigning correct syntactic categories to each word. Tag set and word disambiguation rules are fundamental parts of any POS tagger. The POS tagger relies on the text structure and morphological differences to determine the appropriate part-of-speech. POS Tagging is an essential part of Natural Language Processing (NLP) applications such as speech recognition, text to speech, word sense disambiguation, information retrieval, semantic processing, parsing, information extraction and machine translation. WordNet contains only nouns, verbs, adjectives and adverbs. Since nouns and verbs are more important in representing the content of documents and also mainly form the frequent word meaning sequences, we focus only on nouns and verbs and remove all adjectives and adverbs from the documents. For those word forms that do not have entries in WordNet, we keep them in the documents since these unidentified word forms may capture unique information about the documents. We remove the stop words and then stemming is performed. The morphology function provided with WordNet is used for stemming as it only yields stems that are contained in the WordNet dictionary and also achieves improved results than Porter stemmer. The stemmed words are then looked up in the WordNet the lexical database to replace the words by their synset IDs. These preprocessing steps aim to improve the cluster
quality. These steps lead to the reduction of dimensions in the term space.

IV. DOCUMENT REPRESENTATION

The representation of a set of documents as vectors in a common vector space is known as the vector space model. Documents in vector space can be represented using Boolean, Term Frequency and Term Frequency – Inverse Document Frequency. In Boolean representation, if a term exists in a document, then the corresponding term value is set to one otherwise it is set to zero. Boolean representation is used when every term has equal importance and is applied when the documents are of small size. In Term Frequency and Term Frequency Inverse Document Frequency the term weights have to be set. The term weights are set as the simple frequency counts of the terms in the documents. This reflects the intuition that terms occur frequently within a document may reflect its meaning more strongly than terms that occur less frequently and should thus have higher weights. Each document d is considered as a vector in the term-space and represented by the term frequency (TF) vector:

\[ d = [t_1, t_2, \ldots, t_m] \]  

where \( t_i \) is the frequency of term \( i \) in the document and \( D \) is the total number of unique terms in the text database. The tf–idf representation of the document

\[ d_{tf-idf} = [t_1 \log(D/t_1), t_2 \log(D/t_2), \ldots, t_m \log(D/t_m)] \]  

To account for the documents of different lengths, each document vector is normalized to a unit vector (i.e., \(|d|=1\)). In the rest of this paper, we assume that this vector space model is used to represent documents during the clustering. Given a set \( C_j \) of documents and their corresponding vector representations, the centroid vector \( c_j \) is defined as:

\[ c_j = \frac{1}{|C_j|} \sum_{i \in C_j} d_i \]  

where each \( d_i \) is the document vector in the set \( C_p \) and \( j \) is the number of documents in cluster \( C_j \). It should be noted that even though each document vector \( d_i \) is of unit length, the centroid vector \( c_j \) is not necessarily of unit length.

V. SIMILARITY MEASURES

Document clustering groups similar documents to form a coherent cluster. However, the definition of a pair of documents being similar or different is not always clear and normally varies with the actual problem setting. Accurate clustering requires a precise definition of the closeness between a pair of objects, in terms of either the pair wise similarity or distance [20]. A variety of similarity or distance measures have been proposed and widely applied, such as cosine similarity, Jaccard coefficient, Euclidean distance and Pearson Correlation Coefficient. The details of different similarity measures are described below.

5.1 Cosine Similarity Measure

Classification The most commonly used is the cosine function [8]. For two documents \( d_i \) and \( d_j \), the similarity between them can be calculated

\[ \cos(d_i, d_j) = \frac{d_i \cdot d_j}{\|d_i\| \cdot \|d_j\|} \]  

Where \( d_i \) and \( d_j \) are m-dimensional vectors over the term set \( T = \{t_1, t_2, \ldots, t_m\} \). Each dimension represents a term with its weight in the document, which is non-negative. As a Result the cosine similarity is non-negative and bounded between \([0, 1]\). Cosine similarity captures a scale invariant understanding of similarity and is independent of document length. When the document vectors are of unit length, the above equation is simplified to:

\[ \cos(d_i, d_j) = d_i \cdot d_j \]

![Fig.1. Suffix tree for strings “cat ate”](image)

When the cosine value is 1 the two documents are identical, and 0 if there is nothing in common between them. (i.e., their document vectors are orthogonal to each other).

5.2 Jaccard Coefficient

The Jaccard coefficient, which is sometimes referred to as the Tanimoto coefficient [9, 10] measures similarity as the intersection divided by the union of the objects. For text document, the Jaccard coefficient compares the sum weight of shared terms to the sum weight of terms that are present in either of the two documents but are not the common terms.

\[ \text{Jaccard Coff}(d_i, d_j) = \frac{d_i \cap d_j}{d_i \cup d_j} \]

The Jaccard Coefficient ranges between \([0, 1]\). The Jaccard value is 1 if two documents are identical and 0 if the two documents are disjoint. The Cosine Similarity may be extended to yield Jaccard Coefficient in case of Binary attributes.

5.3 Pearson Correlation Coefficient

Correlation is a technique for investigating the relationship between two quantitative, continuous variables, for example, age and blood pressure. Pearson's
correlation coefficient \((r)\) is a measure of the strength of the association between the two variables. There are different forms of Pearson Correlation Coefficient (PCC) formula. It is given by:

\[
Pearson\ Similarity(d_i, d_j) = \frac{m\sum_{d_k} X_{d_k} \cdot TF_i \cdot TF_j}{\sqrt{(m\sum_{d_k} X^2_{d_k} \cdot |TF_i|)(m\sum_{d_k} X^2_{d_k} \cdot |TF_j|)}}
\]

(5)

Where \(TF_i = \sum_d d_k\) and \(TF_j = \sum_d d_k\) \(m\) is the no. of terms in document \(d\).

The measure ranges from +1 to -1. Positive correlation indicates that both variables increase or decrease together, whereas negative correlation indicates that as one variable increases, so the other decreases, and vice versa. When Pearson Similarity is ±1 the two documents are identical and there is no relation between variables if it is equal to zero.

The Euclidean distance is a distance measure, while the cosine similarity, Jaccard coefficient and Pearson coefficient are similarity measures. We apply a simple transformation to convert the similarity measure to distance values. Because both cosine similarity and Jaccard coefficient are bounded in \([0, 1]\) and monotonic, we take \(D = 1 - SIM\) as the corresponding distance value.

For Pearson coefficient, which ranges from \(-1\) to \(+1\), we take \(D = 1 - SIM\) when \(SIM \geq 0\) an d \(D = |SIM|\) when \(SIM < 0\).

VI. STEPS TO BUILD CBVSTDM

Constructing a Suffix Tree Document Model

Suffix tree document model considers a document \(d = w_1 w_2 \ldots w_m\) as a string consisting of words \(w_1\) and not characters \(i = 1; 2; \ldots m\). A suffix tree of document \(d\) is a compact trie containing all suffixes of document \(d\). Figure 6.1 is an example of a suffix tree composed from three documents. The nodes of the suffix tree are drawn in circles. Each internal node has at least two children. Each edge is labeled with a non-empty substring of a document called a phrase, and its suffix node is labeled by the phrase too. Then each leaf node in the suffix tree designates a suffix of a document and each internal node represents an overlap phrase shared by at least two suffixes. The more internal nodes shared by two documents, the more similar the documents tend to be.

In Figure 1, each internal node is attached with a box respectively. In the practical implementation, each node including leaf node maintains a list storing the numbers displayed in the box. The numbers in the box designate the documents which have traversed the corresponding node. Each upper number designates a document identifier and the number below designates the traversed times of the document.

Figure 1: Suffix tree for strings “cat ate cheese”. Initially suffix trees were introduced by Weiner in 1973 [4]. They represent the single most important data structure for string representation. Their first big advantage is a construction time \(O(m)\), linear in the size \(m\) of the processed string \(S\). Suffix trees are extremely useful because of the fact that search time is independent of the length of the string. A generalized suffix tree (GST) is constructed [5]. A GST [6] is a suffix tree that combines the suffixes of a set of strings. Once a generalized suffix tree of the document collection is built, the document-term matrix, or the TF-IDF matrix can be obtained. This document-term matrix can be built with a single traversal that is DFS traversal of the suffix tree as these values are already stored for each node.

The following steps are carried out in building CBVSTDM:

1. Collecting the document dataset
2. Perform POS tagging
3. Remove stop words
4. Apply stemming by using the WordNet stemmer
5. The stemmed words are then looked up in the WordNet, the lexical database to replace the words by their synset IDs which corresponds to a set of word forms which are synonyms
6. The words with the same synonyms are merged and are assigned a unique ID
7. Unique suffixes are generated
8. Build a concept based generalized suffix tree
9. Constructing a document-term matrix from the generalized suffix tree
10. Perform DFS traversal and obtain all word sequences
11. Retain those frequent word sequences which satisfy the minimum support. The minimum support of the frequent word sequences is usually in the range of 4-15%. When the minimum support is too large, the total number of frequent words would be very small, so that the resulting compact documents would not have enough information about the original data set
12. Thus the feature selection used is DF-thresholding
13. Attach weights to the obtained word sequences using either TF or TF-IDF method
14. The model is now evaluated using K-means clustering. The similarity measures that can be used in the analysis are Cosine, Jaccard, Euclidean and Pearson Correlation Coefficient

REFERENCES


