International Journal of Advanced Technology & Engineering Research (IJATER)
www.ijater.com

# DOCUMENT SUFFIX TREE MODEL USING AFFINITY CLUSTERING FOR FAST SEARCH

Ashish Saxena[1], Aditi Chaturvedi[2]
Asst. Professor, Dept. of IT[1], Asst. Professor, Dept. of Computer Engineering[2]
DYPCOE, Akurdi[1], BSIOTR, Wagholi[2]
Pune, India
ashishsaxenaofc@gmail.com[1], Aditichaturvedi.cse@gmail.com[2]

## Abstract

Phrase has been considered as a more informative feature term for improving the effectiveness of document clustering. In this work, we propose a phrase-based document similarity to compute the pair wise similarities of documents based on the Suffix Tree Document (STD) model. By mapping each node in the suffix tree of STD model into a unique feature term in the Vector Space Document (VSD) model, the phrase-based document similarity naturally inherits the term tf-idf weighting scheme in computing the document similarity with phrases. This work apply the phrase-based document similarity to affinity propagation for better document clustering as much lower error than other methods, and less amount of time.

*Index Terms*—Component, formatting, style, styling, insert. *(key words)*

## I. INTRODUCTION

Document clustering has long been studied as a post retrieval document visualization technique to provide an intuitive navigation and browsing mechanism by organizing documents into groups, where each group represents a different topic [1], [2]. In general, the clustering techniques are based on four concepts: data representation model, similarity measure, clustering model, and clustering algorithm. Most of the current document clustering methods is based on the Vector Space Document (VSD) model [3]. The common framework of this data model starts with a representation of any document as a feature vector of the words that appear in the documents of a data set. A distinct word appearing in the documents is usually considered to be an atomic feature term in the VSD model, because words are the basic units in most natural languages (including English) to represent semantic concepts. In particular, the term weights (usually tf-idf, term-frequencies and inverse document-frequencies) of the words are also contained in each feature vector [4]. The similarity between two documents is computed with one of the several similarity measures based on the two corresponding feature vectors, e.g., cosine measure.

To achieve a more accurate document clustering, a more informative feature term phrase has been considered in recent research work and literature. A phrase of a document is an ordered sequence of one or more words [5]. Bigrams and trigrams are commonly used methods to extract and identify meaningful phrases in statistical natural language processing [6]. Yamamoto and Church [7] presented a method to compute all substrings' (phrases) term frequencies and document frequencies in large document corpora by using suffix array [8]. Reference [9] proposed a phrasebased document index model namely Document Index Graph (DIG), which allows for the incremental construction of a phrase-based index for a document set. The quality of clustering achieved based on this model significantly surpassed the traditional VSD model-based approaches in the experiments of clustering Web documents.

Particularly, the Suffix Tree Document (STD) model and Suffix Tree Clustering (STC) algorithm were proposed by Zamir et al. [10] and Zamir and Etzioni [11]. The STC algorithm was used in their meta searching engine to real time cluster the document snippets returned from other search engines. 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. However, the STD model and STC algorithm have not been analyzed in their work as pointed out by Sven Meyer zu Eissen and Potthast's paper [12]. The STC algorithm got poor results in clustering the documents in their experimental data sets of RCV1 corpus [13]. By studying the STD model, we find that this model can provide a flexible n-gram method to identify and extract all overlap phrases in the documents. The STD model considers a document as a sequence of words, not characters. A document is represented by a set of suffix substrings, the common prefixes of the substrings are selected as phrases to label the edges (or nodes) of a suffix tree. Despite all previous work on the STD model [10], [11], [5], [12], an effective method has not been found to evaluate the effect of each phrase in document clustering algorithms. In contrast, the VSD model uses a feature vector to represent a document. The statistical features of all words are taken into account of the term weights (usually tf-idf) and similarity measures, whereas the sequence order of words is rarely considered in the clustering approaches based on the VSD model. The two

document models are isolated in the current information retrieval techniques.

Thus, we focus our work on how to combine the advantages of two document models in document clustering. As a result of our work, a phrase-based document similarity is presented in this paper. By mapping each node of a suffix tree (excludes the root node) into a unique dimension of an M-dimensional term space (M is the total number of nodes except the root node), each document is represented by a feature vector of M nodes. Consequently, we find a simple way to compute the document similarity: First, the weight (tf-idf) of each node is recorded in building the suffix tree, then the cosine similarity measure is used to compute the pairwise similarities of documents.

The document similarities are mainly decided by the overlap nodes (phrases), which appear in at least two different documents in the document set. All other nodes or phrases are trivial to the phrase-based document similarities, with a slight effect on the overall quality of document clustering.

## II. Literature Review

The paper [22] proposed system utilizes clustering and re-ranking algorithms in order to organize the web documents and provide an order to the results displayed to the user. The fetched documents are further clustered using suffix tree clustering algorithm, which enhances the performance of the web search engine. The results are organized using Page Re-Rank algorithm which considers hyperlink and link structure information to bring an order to the web.

For decades, the most widely used post-retrieval document visualization technique is found to be clustering [30],[31],[32]. In most document clustering algorithms, documents which represents similar ideas are grouped in to similar clusters which greatly helps users to identify the topic of interest. The traditional clustering methods [35] are not considered to be suitable for today's internet users. Unsupervised clustering techniques are not given pre-defined input categories like traditional method So they are considered to be suitable. Conversely, today's search engines pose more challenges demanding new innovations.

It is not possible to utilize traditional clustering algorithms for offline search clustering due to some realistic reasons. There are several other approaches to search results clustering including Semantic On-line Hierarchical Clustering (SHOC) approach [25], Tolerance Rough Set [27], and DisCover [28]. There are academic search engine which uses several clustering algorithms and can robotically organize small collections of the web snippets into clusters [29]. Clustering of web page documents into a hierarchical structure of cluster labels has been proposed by Chuang and Chien [24]. The binary tree structure is constructed using a Hierarchical Agglomerative Clustering (HAC) algorithm, In, Yahoo [34] and DMOZ [26] a multiway-tree cluster hierarchy is created from the binary-tree hierarchy. This is achieved by partitioning the hierarchies in to create sub hierarchies. Baeza-Yates et al. [23] proposed a method which utilizes semantic information. Similar queries are grouped according to their semantics. In this method, a vector representation Q is created for a given search query q. The vector Q has details from the user click actions. To find out the similar queries, a similarity measure involving mathematical cosine function is applied to the query vectors.

Zhang and Nasraoui [36] proposed methods to discover similar queries. The users' sequential search behavior is used as a factor in determining the similarity. It is assumed that successive user queries will be related to each other. Existing content based similarity method is used in addition to above described method to balance the high sparsity of log data collected in real time.

Paper [21] apply the phrase-based document similarity to the group-average Hierarchical Agglomerative Clustering (HAC) algorithm and develop a new document clustering approach. Our evaluation experiments indicate that the new clustering approach is very effective on clustering the documents of two standard document benchmark corpora OHSUMED and RCV1.The quality of the clustering results significantly surpasses the results of traditional single-word tf-idf similarity measure in the same HAC algorithm, especially in large document data sets. Furthermore, by studying the property of STD model, we conclude that the feature vector of phrase terms in the STD model can be considered as an expanded feature vector of the traditional single-word terms in the VSD model. This conclusion sufficiently explains why the phrase-based document similarity works much better than the single-word tf-idf similarity measure.

Text document clustering has been traditionally investigated as a means of improving the performance of search engines by preclustering the entire corpus [4], and a postretrieval document browsing technique as well [1], [2], [11]. Hierarchical Agglomerative Clustering (HAC) algorithm might be the most commonly used algorithm among numerous document clustering algorithms. Generally, there are three variants from this algorithm: single-link, complete-link, and group-average. In practice, the HAC algorithm can often generate high-quality clusters with a tradeoff of the higher computation complexity [15]. K-Nearest Neighbor (K-NN) algorithm is well known for classification [16]. It has also been used for document clustering [9]. In the traditional document models such as the VSD model, words or characters are considered to be the basic terms in statistical feature analysis and extraction.

To achieve a more accurate document clustering, developing more informative features has become more and more important in information retrieval literature recently. Bigrams, trigrams, and much longer n-grams have been commonly used in statistical natural language processing [6].

Suffix tree is a data structure that admits efficient string matching and querying. It has been studied and used extensively in fundamental string problems and applications such as large volumes of biological sequence data searching [14], approximate string matches and text features extraction in spam e-mail classification [15]. The STD model was first proposed in 1997 [10], [11]. Different from document models which treat a document as a set of words and ignore the sequence order of the words, the

STD model considers a document to be a set of suffix substrings, and the common prefixes of the suffix sub-strings are selected as the phrases to label the edges of the suffix tree. The STC algorithm is developed based on this model and works well in clustering Web document snippets. However, the properties of STD model and STC algorithm have not been analyzed in their papers [5], [10], [11]. Sven Meyer zu Eissen and Potthast's paper [12] continued the work and pointed out that the STC algorithm was a fusion heuristic that efficiently evaluated the graph-based similarity measure for the document collections.

Furthermore, their work also proposed several new graph-based similarity measures to compute the document similarities. Their experimental evaluation results showed that the similarity measures, especially the hybrid similarity measure had achieved significant performance improvements in the MajorClust algorithm and the GHAC.

Yamamoto and Church [7] presented a method to compute all substrings' (phrases) term frequencies and document frequencies in large document corpora by using suffix array [8]. Their approach provided an efficient way for extracting phrases and computing their tf-idf weights in a static document set. Another relevant work is the DIG proposed by Hammouda and Kamel [9], which allows for the incremental construction of a phrase-based index for a document set. The quality of clustering achieved by using the hybrid similarity measure based on this model sig-nificantly surpassed the approaches based on the traditional VSD model.

# III. Phrase-Based Document Similarity

Throughout this paper, we use the symbols N, M, and k to denote the number of documents, the number of terms, and the number of clusters, respectively. We use the symbol D to denote the document set of N documents that we want to cluster, the C1, C2, . . . , Ck to denote each one of the k clusters. In text-based information retrieval, a document model is a concept that describes how a set of meaningful features is extracted from a document. Most of the current document clustering methods uses the VSD model to represent documents. In the model, each document d is considered to be a vector in theM-dimensional term space. In particular, we usually employ the term tf-idf weighting scheme [4], [16], in which each document can be represented as

$$\vec{d} = \{w(1,d), w(2,d), \ldots, w(M,d)\}$$

Where $w(i,d) = (1 + \log tf(i,d)) \cdot \log(1 + N/df(i))$, $tf(i,d)$ the frequency of the ith is term in the document d, and is the number of documents containing the ith term.

In the VSD model, the cosine similarity is the most commonly used measure to compute the pair wise similarity of two document di and dj, which is defined as

# IV. Suffix Tree Document Model

The STD model considers a document d as a string consisting of words w1w2 . . .wm, not characters. The suffix tree of a document d is a compact trie containing all suffix substrings of the document d. Fig. 1 is an example of a suffix tree composed from three documents. The nodes of the suffix tree are drawn in circles. There are three kinds of nodes in the suffix tree: the root node, internal nodes, and leaf nodes. Each internal node has at least two children. Each edge is labeled with a nonempty substring of a document called a phrase. Then, each leaf node in the suffix tree designates a suffix substring of a document; each internal node represents a common phrase shared by at least two suffix substrings. The similarity of two documents is defined as the more internal nodes shared by the two documents, the more similar the documents tend to be.

In Fig. 1, each internal node is attached to an individual box. The numbers in the box designate the documents that have traversed the corresponding node. Each upper number designates a document identifier, the number below designates the traversed times of the document. (In the implementation of our approach, the node data structure has a list storing the numbers directly.)

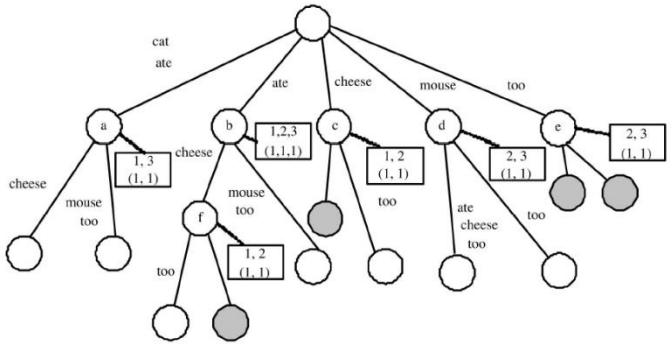

Fig. 1. The suffix tree of tree documents "cat ate cheese," "mouse ate cheese too," and "cat ate mouse too."

# V. Phrase-Based Document Similarity Based on the STD Model

As mentioned in the previous section, there are three different kinds of nodes in a suffix tree. In particular, there exist some leaf nodes labeled with an empty phrase (usually a NULL in the suffix tree implementations). They are generated by Ukkonen's algorithm [17], which is used to build suffix tree, and denote the end of the corresponding documents. For example, in Fig. 1, the four leaf nodes represented by gray circles are such nodes. We call these leaf nodes "terminal nodes" in our work. With the exception of the terminal nodes and the root node, each node in the suffix tree, either an internal node or a leaf node represents a nonempty phrase that appears in at least one document in the data set. The same phrase might occur in different edges of the suffix tree. For instance, there are three different edges labeled with the same phrase of "cheese" in the suffix tree of Fig. 1. The definition of the phrase-based document similarity is simple and understandable: By mapping each node v labeled by a nonempty phrase into a feature of M-dimensional term space, each document d can be represented as a feature vector of the weights of M node terms in the VSD model as illustrated by (1). It is very easy to understand that the document frequency of each node df(v) is the number of the different documents that have traversed node v; the term frequency tf(v,d) of a node v with respect to document d is the total traversed times of the document d through node v. In the example of Fig. 1, the df of node b is df(b)=3, the tf of the node with respect to the document 1 is tf(b,1)=1 (assuming the document identifiers of three documents to be 1, 2, 3). Therefore, we can calculate the weight of node b with respect to document 1 as . After obtaining the term weights of all nodes, it is easy to apply traditional similarity measures such as the cosine similarity to compute the similarity of any two documents. In this paper, the cosine similarity measure is used to compute the pairwise similarities of all documents.

Let vectors , and denote two documents dx and dy, where xi and yi are the weights of corresponding node term vi, respectively. Then, the similarity of two documents is calculated by the following formula:

$$sim_{x,y} = \frac{\vec{d_x} \bullet \vec{d_y}}{|\vec{d_x}| \times |\vec{d_y}|} = \frac{\sum_{i=1}^{M} x_i y_i}{\sqrt{\sum_{i=1}^{M} x_i^2 \sum_{i=1}^{M} y_i^2}}$$

# VI. Affinity Clustering Based on Similarity

Data mining, or exemplars, is traditionally found by randomly choosing an initial subset of data points and then iteratively refining it, but this only works well if that initial choice is close to a good solution.

Affinity propagation [19] is a new algorithm that takes as input measures of similarity between pairs of data points and simultaneously considers all data points as potential exemplars. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. hence affinity propagation to solve a variety of clustering problems and found that it uniformly found clusters with much lower error than those found by other methods, and it did so in less than one-hundredth the amount of time. Because of its simplicity, general applicability, and performance, we believe affinity propagation will prove to be of broad value in science and engineering.

Clustering data by identifying a subset of representative examples is important for processing data clustering and detecting patterns in data [20]. Such "exemplars" can be found by randomly choosing an initial subset of data points and then iteratively refining it, but this works well only if that initial choice is close to a good solution. We devised a method called "affinity propagation," which takes as input measures of similarity between pairs of data points. Real-valued messages are exchanged between data points until a high-quality set of exemplars and corresponding clusters gradually emerges. We used affinity propagation to cluster images of faces, detect genes in microarray data, identify representative sentences in this manuscript, and identify cities that are efficiently accessed by airline travel. Affinity propagation found clusters with much lower error than other methods, and it did so in less than one-hundredth the amount of time.

# VII. PROPOSED WORK

### *A. Simulation Tool*

This work may implement using Matlab Simulation tool by using its compatibility with large dataset files handling easily.

### *B. Document Preprocessing*

Before the document clustering, a document "cleaning" procedure is executed for all documents on the selected data sets

1. First, all non word tokens are stripped off.
2. Second, the text is parsed into words.
3. Third, all stop words are identified and removed.
4. Fourth, the Porter's suffix-stripping algorithm [18] is used to stem the words.

5. Finally, all stemmed words are concatenated into a new document.

Since the length of a word is variable, it is quite difficult to implement a suffix tree based on words directly. To solve the problem, we build a wordlist to store all keywords in alphabetical order. The similar ideas are often used in some text retrieval approaches for simplifying the computation complexity, such as the inverted index systems. In the wordlist, a unique integer number (called a word_id) is assigned to each keyword so that we can use the word_id to replace the corresponding word in the "cleaned" document. Finally, each document becomes an array of word_ids for the suffix tree construction.

*C. Suffix Tree Construction*

After document preprocessing cleaned document use for constructing suffix tree as mention suffix tree could be generate for every document we want to cluster and then a generalized suffix tree may created using these suffix tree for every document as shown in fig. 1.

*D. Similarity measure*

Using generalized suffix tree of tf and idf are calculated this value used for finding similarity between documents that how many a document i similar to other document j this similarity value goes from 0 to 1. If sim(I,j)=1 then both document completely common if this similarity value is 0 both document doesn't have any common phrase. Similarity matrix is for storing similarity between all documents in data set to each other where rows and columns both are documenting Id.

*E. Document Clustering*

This work proposed Affinity clustering algorithm to cluster documents according to similarity group of documents having more similarity form a separate cluster according to topic wise. Such as all documents having description of computer are similar to each other then they form a cluster having document related to computer become member of this cluster. This work uses affinity clustering for document cluster for better document clustering as much lower error than other methods, and less amount of time.

# VIII. CONCLUSIONS

Both the traditional VSD model and STD model play important roles in text-based information retrieval. However, the two models are used in two isolated ways: Almost all clustering algorithms based on the VSD model ignore the occurring position of words in the documents and the different semantic meanings of a word in different sentences are unavoidably discarded. The STD model keeps all sequential characteristics of the sentences in each document, the phrases consisting of one or more words are used to designate the similarity of two documents. Through analyzing the original STC algorithm, we become interested in the STD model. As a result of studying the property of STD model, we propose a phrase-based document similarity for document clustering using affinity propagation clustering model as efficient clustering techniques with less error.

The concept of the suffix tree and the new document similarity are quite simple, but the implementation is complicated. To improve the performance of the phrase based document similarity, we investigated the STD model with the affinity clustering algorithmic for optimization. In fact, the phrases in documents are independent to the extraction methods and tools. For the first time, feature vectors of phrases' tf-idf weights are used in computing document similarities and are proven to be very effective in clustering documents. Our work has presented a successful approach to extend the usage of tf-idf weighting scheme: the term tf-idf weighting scheme is suitable for evaluating the importance of not only the keywords but also the phrases in document clustering.

# REFERENCES

[1] P.O.R. Allen and M. Littman, "An Interface for Navigating Clustered Document Sets Returned by Queries," Proc. ACM Conf. Organizational Computing Systems (COCS '93),pp. 166-171, 1993.

[2] W.B. Croft, "Organizing and Searching Large Files of Documents," PhD dissertation, Univ. of Cambridge, 1978.

[3] G. Salton, A. Wong, and C.S. Yang, "A Vector Space Model for Automatic Indexing,"Comm. ACM,vol. 18, no. 11, pp. 613-620, 1975.

[4] C.J. van Rijsbergen, Information Retrieval. Butterworths, 1975.

[5] O. Zamir and O. Etzioni, "Grouper: A Dynamic Clustering Interface to Web Search Results," Computer Networks,vol. 31, nos. 11-16, pp. 1361-1374, 1999.

[6] E. Charniak,Statistical Language Learning.MIT Press, 1993.

[7] M. Yamamoto and K.W. Church, "Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus," Computational Linguistics, vol. 27, no. 1, pp. 1-30, 2001.

[8] U. Manber and G. Myers, "Suffix Arrays: A New Method for On-Line String Searches," SIAM J. Computing, vol. 22, no. 5, pp. 935-948, 1993.

[9] K.M. Hammouda and M.S. Kamel, "Efficient Phrase-Based Document Indexing for Web Document

Clustering," IEEE Trans. Knowledge and Data Eng.,vol. 16, no. 10, pp. 1279-1296, Oct. 2004.

[10] O.M. Oren Zamir, O. Etzioni, and R.M. Karp, "Fast and Intuitive Clustering of Web Documents, "Proc. Third Int'l Conf. Knowledge Discovery and Data Mining (KDD), 1997.

[11] O. Zamir and O. Etzioni, "Web Document Clustering: A Feasibility Demonstration," Proc. 21st Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR),1998.

[12] D.S. Sven Meyer zu Eissen and M. Potthast, "The Suffix Tree Document Model Revisited," Proc. Fifth Int'l Conf. Knowledge Management (I-Know '05),pp. 596-603, 2005.

[13] D.D. Lewis, Y. Yang, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research,"J. Machine Learning Research,vol. 5, pp. 361-397, 2004.

[14] J.R. Paul Bieganski and J.V. Carlis, "Generalized Suffix Trees for Biological Sequence Data: Application and Implementation, "Proc. 27th Ann. Hawaii Int'l Conf. System Sciences (HICSS '94),pp. 35-44, 1994.

[15] B.M. Rajesh Pampapathi and M. Levene, "A Suffix Tree Approach to Anti-Spam Email Filtering," Machine Learning, vol. 65, 2006.

[16] G. Salton and C. Buckley, "On the Use of Spreading Activation Methods in Automatic Information Retrieval,"Proc. 11th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR '88),pp. 147-160, 1988.

[17] E. Ukkonen, "On-Line Construction of Suffix Trees,"Algorithmica, vol. 14, no. 3, pp. 249-260, 1995.

[18] M. Porter, "New Models in Probabilistic Information Retrieval," British Library Research and Development Report, no. 5587, 1980.

[19] http://www.psi.toronto.edu/index.php?q=affinity%20 propagation

[20] Brendan J. Frey* and Delbert Dueck, "Clustering by Passing Messages between Data Points", report VOL 315 974-977 SCIENCE, FEBRUARY 2007

[21] Hung Chim and Xiaotie Deng, "Efficient Phrase-Based Document Similarity for Clustering", IEEE Transactions on knowledge and data engineering, vol. 20, no. 9, pp 1217-1229, 2008 IEEE.

[22] Ajitha Annadurai, Anitha Annadurai, "Architecture of Personalized Web Search Engine Using Suffix Tree Clustering", Proceedings of ICSCCN, 2011 IEEE.

[23] R.A. Baeza-Yates, C.A. Hurtado, and M. Mendoza, "Query Recommendation Using Query Logs in Search Engines," Proc.EDBT Workshop, vol. 3268, pp. 588-596, 2004.

[24] S. Chuang and L. Chien, "Automatic Query Taxonomy Generation for Information Retrieval Applications," Online Information Rev.vol. 27, no. 4, pp. 243-255, 2003.

[25] Dell Zhang, Yi-sheng Dong, "Semantic, Hierarchical, Online Clustering of Web Search Results", Proceedings of the 6th Asia Pacific Web Conference, Hangzhou, China, vol. 3007, pp. 69-78, 14-17 Apr 2004. http://www.dmoz.org/, 2008.

[26] P.W. Foltz and S.T. Dumais, "Personalized Information Delivery: An Analysis of Information Filtering Methods," Comm. ACM,vol. 35, no. 12, pp. 51-60, 1992.

[27] Y. Fu, S. Yan, and T. Huang, "Correlation metric for generalized feature extraction," IEEE Trans. Pattern Anal. Mach. Intell., pp.2229–2235, 2008.

[28] M. A. Hearst and J. O. Pedersen, "Reexamining the cluster hypothesis: Scatter/Gather on retrieval results",Proceedings of the 19th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96), 1996, pp.76-84.

[29] H. T. Nguyen and A. Smeulders, "Active learning using pre-clustering," in Proc. Int. Conf. Machine Learning, 2004, pp. 623–630.

[30] M. Spiliopoulou, B. Mobasher, B. Berendt, and M. Nakagawa,"A Framework for the Evaluation of Session Reconstruction Heuristics in Web Usage Analysis," INFORMS J. Computing,no. 2, p. 15, 2003.

[31] Sven Meyer zu Eissen, Benno Stein, Martin Potthast, "The Suffix Tree Document Model Revisited",Proceedings of the 5th International Conference on Knowledge Management, Universal Computer Science, pp. 596-603, 2005.

[32] I.H. Witten and E. Frank, Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann, 2000.

[33] http://www.yahoo.com/, 2008.

[34] Zamir O., Etzioni O., "Web Document Clustering: A Feasibility Demonstration", Proceedings of the 19thInternational ACM SIGIR Conference on Research and Development of Information Retrieval (SIGIR'98), 1998,pp.46-54.

[35] Z. Zhang and O. Nasraoui, "Mining Search Engine Query Logs for Query Recommendation," Proc. 15th Int'l World Wide Web Conf. (WWW),2008