ISSN: 2229-6956 (ONLINE)
DOI: 10.21917/ijsc.2013.0096

MEASURING THE PERFORMANCE OF SIMILARITY PROPAGATION IN AN SEMANTIC SEARCH ENGINE

S. K. Jayanthi[1] and S. Prema[2]
[1]*Department of Computer Science, Vellalar College for Women, India*
E-mail: jayanthiskp@gmail.com
[2]*Department of Computer Science, K.S.R College of Arts and Science, India*
E-mail: premashanmuga11@gmail.com

Abstract

In the current scenario, web page result personalization is playing a vital role. Nearly 80 % of the users expect the best results in the first page itself without having any persistence to browse longer in URL mode. This research work focuses on two main themes: Semantic web search through online and Domain based search through offline. The first part is to find an effective method which allows grouping similar results together using BookShelf Data Structure and organizing the various clusters. The second one is focused on the academic domain based search through offline. This paper focuses on finding documents which are similar and how Vector space can be used to solve it. So more weightage is given for the principles and working methodology of similarity propagation. Cosine similarity measure is used for finding the relevancy among the documents.

Keywords:

Semantic Web, BookShelf Data Structure, Similarity Propagation, Cosine Similarity measure, Vector Space Model

1. INTRODUCTION

Information Retrieval is a problem of selecting the relevant information from a document database in response to search queries given by a user. Information Retrieval Systems (IRSs) deal with document database that usually consist of textual information and process user queries to provide the user with access to relevant information within a reasonably acceptable time interval. After retrieving the results it is essential for a search engine to rank-order the documents matching a query. To perform this, the search engine calculates, for each matching document, a score with respect to the given query. This research work initiate the study of assigning a score to a (query, document) pair.

Two types of mode are discussed in this research work. One is an online web search and another is an offline search. For both the mode of research the similarity between the retrieved documents are calculated using the cosine similarity measure.

In summary, this architecture gives much higher quality results yet with similar response time to other systems. The rest of the paper is organized as follows: related work is discussed in Section 2 and provides the preliminaries on web search engines. Section 3 provides an idea for finding similarity propagation among documents .It also focus on vector space model and cosine similarity measure. Term frequency and Inverse document frequency for similarity calculation is discussed. Section 4 explains the experimental results of Online and Offline research. Arranging the retrieved results in BookShelf Data Structure is also reported in this section. Finally, Section 5 gives the concluding remarks.

2. RELATED WORK

Qingtian Han and Xiaoyan Gao [1] analyses a Web mining algorithm based on usage mining. A novel and efficient approach for the detection of nearest duplicate web pages was designed by V. A. Narayana et al. [2].The documents with similarity scores greater than a threshold value are considered as near duplicates. J. Akilandeswari and N. P. Gopalan [3] analyzed an architectural framework of a crawler for locating deep web repositories using learning multi-agent Systems. Crawling and Page Rank Algorithms for Internet Searches are designed by Animesh Tripathy and Prashanta K Patra [4]. An architecture and implementation prototype of web data mining system based on web service was developed by Chunying Chen et al. [5].

This approach takes a source-centric perspective on the information-seeking process, aiming to identify trustworthy sources of relevant information from within the user's social network by Tom Heath [6]. Where an individual encounters a problem or task for which their current knowledge is inadequate, they may engage in information-seeking in order to change their knowledge state (Belkin) [7].

Louis S. Wang [8] presents a modified vector space model for measuring similarity between the query and the document. Huilian Fan et al. [9] designed a new crawling strategy which combined the advantages of hyperlinks structure and web content strategies. Topic keywords based VSM is used to evaluate individual fitness, and imports new URLs to implement crossover and mutation, and the URLs that have the same prefix are regarded as niche. Mehran Sahami et al. [10] proposed a similar kernel function, for measuring the similarity between short text snippets.

Sean A. Golliher [11], two classes (on-page and off-page variables) of search engine ranking factors and their possible implications for ranking web documents are discussed. Albert Bifet et al. [12] analyzed the result rankings for several queries of different categories using statistical methods in Google (via its API) as testbed. Chowdhury Farhan Ahmed et al. [13] designed an efficient mining of utility-based web path traversal patterns. Hazem Elmeleegy et al. [14] provides a novel technique for extracting tables from lists. The technique is domain-independent and operates in a fully unsupervised manner.

The link structure of a web site can be visualized in a link hierarchy consisting of web pages on multiple conceptual levels for user navigation [15]. Durand and Kahn [16] developed a system called MAPA to extract a hierarchical structure from an arbitrary web site for navigation.

3. SIMILARITY PROPAGATION

The main contribution of this paper is to find the similarity propagation among the documents. The approach discussed in this section is common for both the online web search and offline domain-centric approach. To workout this approach one should have the basic idea about the following: tf-idf calculation, Vector space model, Query as vector, tf-idf weight calculation and cosine similarity measure. All those fundamental concepts are discussed in the following sections.

The weight of the components of a document vector can be represented by Term Frequency or combination of tf and idf. Number of occurrences of a term t in the document D is referred by tf. Number of documents, where a particular term t occurs is noted as df. In idf i.e. log (n/df) word with rare occurrences has more weight.tf-idf is calculated as tf-idf= tf ×idf

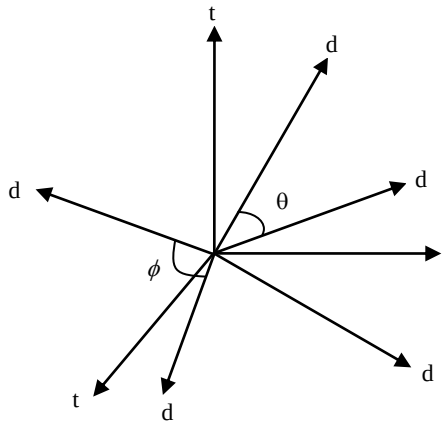

Fig.1. Documents as vector

Documents D are points or vectors in this space. Terms are axes of the space. Documents that are "close together" in vector space talk about the same things. View each document as a vector with one component corresponding to each term in the dictionary , together with a weight for each component that is given by $tf\text{-}idf_{t,d} = tf_{t,} \times idf_{t}$ .For dictionary terms that do not occur in a document, this weight is zero as in Fig.1.

3.1 VECTOR SPACE MODEL

The representation of a set of documents as vectors in a common vector space is known as the vector space model and is fundamental to a host of information retrieval operations ranging from scoring documents on a query, document classification and clustering.

The tf-idf values can now be used to create vector representations of documents. Each component of a vector corresponds to the tf-idf value of a particular term in the corpus dictionary. Dictionary terms that do not occur in a document are weighted zero. Using this kind of representation in a common vector space is called vector space model, which is not only used in information retrieval but also in a variety of other research fields like machine learning (e.g. clustering, classification). A vector space provides the possibility to perform calculations like computing differences or angles between vectors. Since documents are usually not of equal length, simply computing the difference between two vectors has the disadvantage. The documents of similar content but different length are not regarded as similar in the vector space.

Each term from the collection becomes a dimension in an n-dimensional space. A document is a vector in this space, where term weights serve as coordinates. It is important for scoring documents for answering queries, Query by example, Document classification and Document clustering (Prabhakar) [17].

Formalizing vector space proximity doesn't use Euclidean distance because it is large for vectors of different lengths. But at the same time using cosine similarity will result in ranking the documents in increasing order of cosine (query, document).

3.2 RELEVANCY MEASURE

The vector space representation of text is an incredibly powerful tool. Any text can be treated as a vector in a V-dimensional vector-space (Jaime Arguello) [18]. Documents are matched with a query based on their similarity. If a document is similar to the query, it is likely to be relevant. Non-binary weights for index terms in queries and documents are used in the calculation of degree of similarity. Decreasing order of this degree of similarity for the retrieved documents gives the ranked documents with partial match (Manwar et al.) [19].

For the vector model, the weight $w_{i,j}$ associated with a pair (k_i, d_j) is positive and non-binary. Further, the index terms in the query are also weighted. Let $w_{i,q}$ be the weight associated with the pair(k_i, q) ,where $w_{i,q} \geq 0$. Then, the query vector $\vec{q}$ is defined as $\vec{q} = (w_{1,q,} w_{2,q,\ldots\ldots} w_{n,q})$ where n is the total number of index terms in the system. The vector for a document $\vec{d}_j$ is represented by $\vec{d}_j = (w_{1,j,} w_{2,j,\ldots\ldots} w_{n,j})$ by Manwar et al. The main objective is to retrieve more documents like those labeled relevant and fewer documents like those labeled irrelevant.

The combination of tf and df is the most popular weight used in case of document similarity exercises. The weight is high when *t* occurs many times within a small number of documents. The weight is low, when the term occurs fewer times in a document or in many documents. A vector V can be expressed as a sum of elements such as, $V = a_1 v_{i1} + a_2 v_{i2} + \ldots\ldots + a_n v_{in}$ where, a_k are called scalars or weights and v_{in} as the components or elements.

Consider two document vectors d_1, d_2 and a query vector Q. The space contains terms $\{t_1, t_2, t_3,\ldots,t_n\}$.The document d_1 has components $\{t_1, t_3,\ldots..\}$ and d_2 has components $\{t_2, t_4,\ldots.\}$.So $V(d_1)$ is represented closer to axis t_1 and $V(d_2)$ is closer to t_2.The angle θ represents the closeness of a document vector to the query vector. Its value is calculated by cosine of θ.

3.3 COSINE SIMILARITY MEASURE

To avoid the bias caused by different document lengths, a common way to compute the similarity between the two documents is using the cosine similarity measure. The inner product of the two vectors is divided by the product of their vector lengths. This has the effect that the vectors are normalized to unit length and only the angle, more precisely the cosine of the angle, between the vectors account for their similarity.

Documents not sharing a single word get assigned a similarity value of zero because of the orthogonality of their

vectors while documents sharing a similar vocabulary get higher values. Because a query can be considered a short document, it is of course possible to create a vector for the query, which can then be used to calculate the cosine similarities between the query vector and those of the matching documents. Finally, the similarity values between the query and the retrieved documents are used to rank the results.

For any two given documents d_j and d_k, their similarity is:

$$sim(d_j, d_k) = \frac{\vec{d}_j . \vec{d}_k}{\left|\vec{d}_j\right|\left|\vec{d}_k\right|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2}\sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

where, w_i is a weight of the documents. The most popular way to measure the similarity between two frequency vectors (raw or weighted) is to take their cosine value as in Fig.2. It is necessary to compute the cosine angle between A (the query) and each document and sort these in decreasing order of cosine angles. This treatment can be extended to the entire collection of documents.

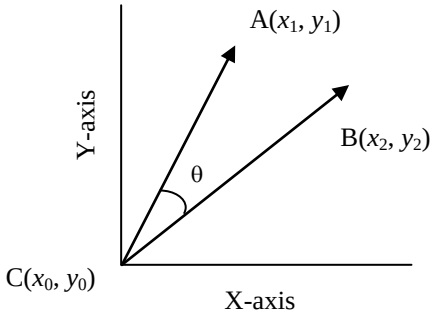

Fig.2. Cosine angle

3.4 TF AND IDF WEIGHT FOR SIMILARITY CALCULATION

Cosine similarity between two documents is given by,

$$sim(d_1, d_2) = \frac{v(d_1).v(d_2)}{|v(d_1)||v(d_2)|}$$

If the vector d_1 has component weights {w_1, w_2, w_3} and vector d_2 has component weights {u_1, u_2}, then the dot product = w_1*u_1 + w_2*u_2.

Since there is no third component, w_3*null = 0.

Euclidean length of $d_1 = \sqrt{w_1^2 + w_2^2 + w_3^2}$.

Consider there are 3 documents,

D_1 = "The file contains operating concepts"

D_2 = "My laptop is operating under windows operating system"

D_3 = "This system is not working properly"

Q = "Operating System"

Number of documents = 3; inverse document frequency IDF = log (D/df$_i$) is calculated as in Table.1.

Calculating the vector lengths

Euclidean length of the Vector is $|D| = \sqrt{\sum_i (W_{i,j})^2}$

$$|D_1| = \sqrt{(0.4771)^2 + (0.4771)^2 + (0.4771)^2 + (0.4771)^2 + (0.1760)^2}$$
$$= 0.9702$$

$$|D_2| = \sqrt{(0.1760)^2 + (0.4771)^2 + (0.4771)^2 + (0.352)^2 + (0.1760)^2 + (0.4771)^2}$$
$$= 0.9320$$

$$|D_3| = \sqrt{(0.1760)^2 + (0.4771)^2 + (0.4771)^2 + (0.1760)^2 + (0.4771)^2}$$
$$= 0.8630$$

$$|Q| = \sqrt{(0.1760)^2 + (0.1760)^2}$$
$$= 0.2489$$

Table.1. tf-idf calculation

Term	tf$_i$				df$_i$	D/df$_i$	idf$_i$	Weights=tf$_i$*idf$_i$			
	Q	D1	D2	D3				Q	D1	D2	D3
concepts	0	1	0	0	1	3	0.4771	0	0.4771	0	0
contains	0	1	0	0	1	3	0.4771	0	0.4771	0	0
file	0	1	0	0	1	3	0.4771	0	0.4771	0	0
is	0	0	1	1	2	1.5	0.1760	0	0	0.1760	0.1760
laptop	0	0	1	0	1	3	0.4771	0	0	0.4771	0
My	0	0	1	0	1	3	0.4771	0	0	0.4771	0
not	0	0	0	1	1	3	0.4771	0	0	0	0.4771
operating	1	1	2	0	2	1.5	0.1760	0.1760	0.1760	0.352	0
properly	0	0	0	1	1	3	0.4771	0	0	0	0.4771
system	1	0	1	1	2	1.5	0.1760	0.1760	0	0.1760	0.1760
The	0	1	0	0	1	3	0.4771	0	0.4771	0	0
This	0	0	0	1	1	3	0.4771	0	0	0	0.4771
under	0	0	1	0	1	3	0.4771	0	0	0.4771	0
Windows	0	0	1	0	1	3	0.4771	0	0	0.4771	0

Calculate the dot products of the query vector with each Document vector,

$$Q.D_i = \sqrt{W_{Q,j} * W_{i,j}}$$

$Q.D_1$ = 0.1760 * 0.1760 = 0.030976

$Q.D_2$ = 0.1760 * 0.352+ 0.1760 * 0.1760 = 0.0929

$Q.D_3$ = 0.1760 * 0.1760 = 0.030976

Cosine value calculation

Cosine $\theta(d_1) = Q.D_1/|Q|*|D_1|$=0.030976/(0.2489*0.9702)=0.1282

Cosine $\theta(d_2) = Q.D_2/|Q|*|D_2|$=0.0929/(0.2489*0.9320) = 0.4004

Cosine $\theta(d_3) = Q.D_3/|Q|*|D_3|$=0.030976/(0.2489*0.8630)=0.1442

Document D_2 is more similar to the query. Cosine formula gives a score which can be used to order documents. Documents with a partial match are also identified and the problem is that positional information about the terms is missing.

The implementation of cosine value calculation is done in Java and the source code is:

```
if(len1>len22)
 {
  for(int p=0;p<len22;p++)
  {
    System.out.println(" if ss1 "+ss1[p]+" "+ss11[p]+" "+ss2[p]+" "+ss22[p]);
    d1=(Double.parseDouble(ss1[p]));
    d2=(Double.parseDouble(ss11[p]));
    d3=(Double.parseDouble(ss2[p]));
    d4=(Double.parseDouble(ss22[p]));
    double dif1=(d1/d3);
    double dif2=(d2/d4);
    double sdf=(Double.parseDouble(sdd5[p]));
    double sdf1=(Double.parseDouble(sdd6[p]));
if(dif1>dif2)
{
  sim=(dif1*sdf*sdf1);
}
if(dif1<dif2)
{
  sim=(dif2*sdf*sdf1);
}
}
int dt=(e+1);
String fit=w+" "+dt+" "+sim;
System.out.println("Dis "+fit);
System.out.println("insert into Similarity values("+tt+",'"+w+" "+dt+"','"+sim+"')");
//db.st.executeUpdate("insert into Similarity values("+tt+",'"+w+" "+dt+"','"+sim+"')");
tt++;
res.add(fit.trim());
ta.append(fit+"\n");
}
else
{
boolean bt=true,bt11=true;
for(int p1=0;p1<len1;p1++)
{
System.out.println("ss1 "+ss1[p1]+" "+ss11[p1]+" "+ss2[p1]+" "+ss22[p1]);
d1=(Double.parseDouble(ss1[p1]));
//d2=(Double.parseDouble(ss11[p1]));
d3=(Double.parseDouble(ss2[p1]));
```

4. EXPERIMENTAL RESULTS

4.1 ONLINE SEARCH

Online results for the search query “operating system” are considered for the experimental dataset. The similarity propagation for the retrieved web documents are calculated following the procedure as in section 3.

The probability measure for all the retrieved documents are considered like,

(0,1),(0,2),(0,3)(0,4),(0,5),(0,6),(0,7),(0,8),(0,9),(0,10),
(1,0),(1,2),(1,3)(1,4),(1,5),(1,6),(1,7),(1,8),(1,9),(1,10),…….,(10,0),(10,1),(10,2),(10,3)
(10,4),(10,5),(10,6),(10,7),(10,8),(10,9) }

For every pair of document similarity, automatic updation of SQL table is created with column specification for document id of two documents and similarity value and the similarity propagation measure is given in Fig.3.

```
String qur="create table Hierarchical (id varchar (500), doc1 numeric (18, 10),";
for (int k=0;k<(out.length-1);k++)
    {
    if(k<(out.length-2))
        {
            qur+="doc"+(k+2)+" "+"numeric(18,10)"+",";
        }
    else
        {
            qur+="doc"+ (k+2) +" "+"numeric (18, 10)"+")";
        }
    }
```

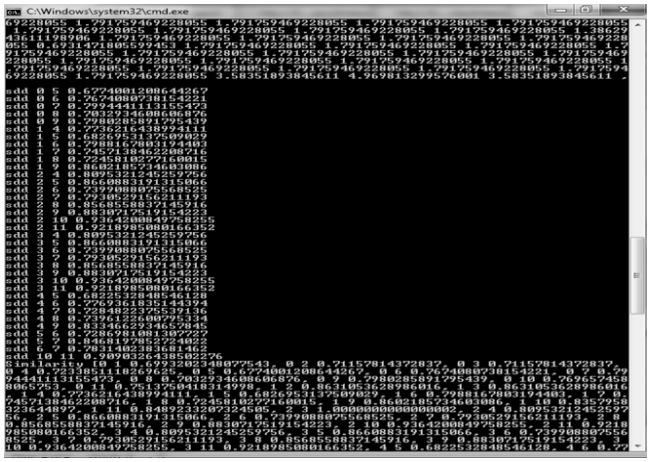

Fig.3. Similarity Propagation Measure

Similarity values are automatically calculated for each and every web search, i.e., dynamically stored in the Java file for further reference as in Fig.4.

The Fig.5 represents the plots of clustering quality against the number of clusters for the given query in online search using the similarity propagation representation. In each of the graphs, the curves corresponding to the two similarity measures are

shown (Jayanthi and Prema) [21]. It can be clearly seen that the quality of clustering increases monotonically with the number of clusters.

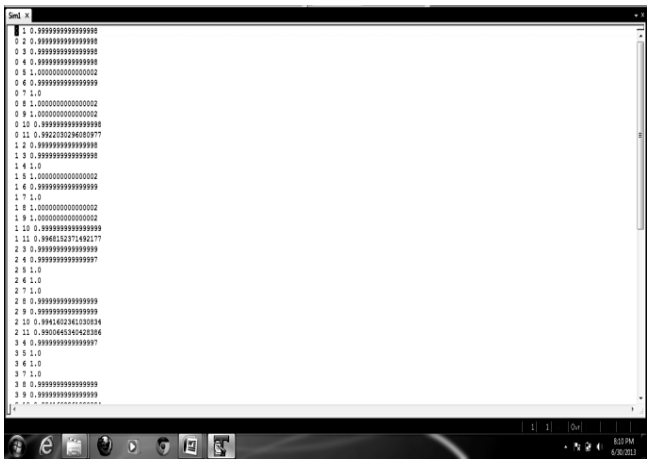

Fig.4. Automatic Backup of the Similarity Propagation File

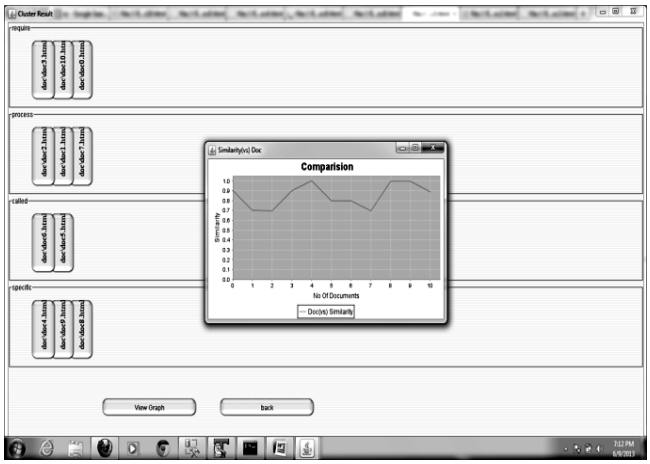

Fig.5. Documents versus Similarity measure

Initially, as expected, the increase in quality of clusters is rapid. It can also be observed that the curve for the quality of clusters flattens out somewhat for the cluster numbers close to the natural number of clusters for the corresponding dataset. The corresponding screenshots are given in Fig.6 and Fig.7.

Fig.6. Clustered Results

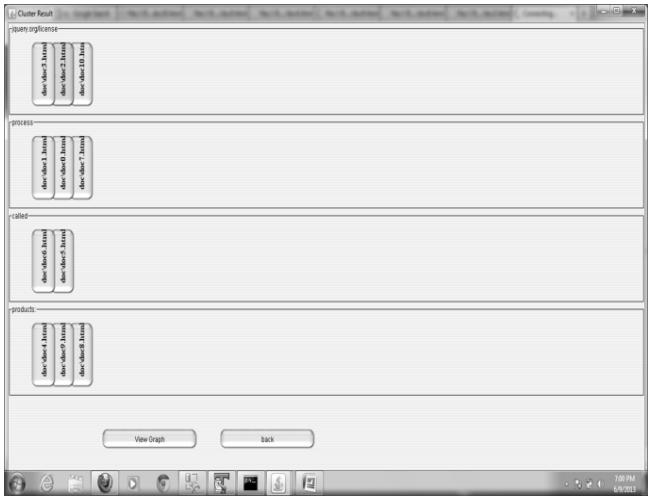

Fig.7. Cluster measure

4.2 OFFLINE SEARCH

Two types of modes are focused in an offline search. One is an IT professional search and the other is a non-professional search. Professional search is based on domain-centric system. Document collection and keywords focused in this section are all based on computer science domain. Main and sub domain sample documents collected for professional domain-centric search is in Table.2. Sample documents for non-professional search with their keywords are in Table.3.

Table.2. Professional Domain-Centric Clustering

Main domain	Sub-domain keywords
Operating System	“apple”,”windows”,”unix”,”kernel”
Network	"communication"," protocol","topology","layer"
DBMS	"dbms","sql","oracle"
Data Structure	"data structures","array","list","vector"
SOFT	"Software Testing”, “Software Design","Implementation","Software development"

Table.3. Non-Professional documents Clustering

Main domain	Sub-domain keywords
window	"door","window","design","interior"
apple	"history","apple","fruit","season"
mouse	"history","mouse","home"

The mode of search i.e., professional or non-professional is to be decided by the user. Professional search is focused on computer science domain. Non-professional search is focused on general concept.

5. CONCLUSION

There is a limited average time they will spend before giving up, or becoming very upset with the search technology available for them. It is found that people will not search for long on the

web. To speed up the web search, an alternative approach is discussed in this paper.

Two modes of searching options are provided in this research work. One is an online web search and another one is an offline search. Instead of disturbing all the documents, the web search with the most similar propagation is considered. The highly correlated documents are clustered using Agglomerative hierarchical clustering. The clustered documents are arranged in BookShelf Data Structure for an easy access (Jayanthi and Prema) [20]. For users, seeing clustered search results has the following benefits such as

- Bringing those search results into easy view that otherwise would remain invisible because they are far down the list.
- Allows users to examine nearly double the number of relevant documents than in the case of result lists of commercial search engines.
- Leads to effortless knowledge discovery as the user learns the types or subtopics of available information relating to the query.

Provides context by placing the related documents within a single folder for joint viewing. All of these factors have significant impact on the user's search productivity.

REFERENCES

[1] Qingtian Han and Xiaoyan Gao, "Research of Distributed Algorithm Based on Usage Mining", *Proceedings of Second International Workshop on Knowledge Discovery and Data Mining*, pp. 211-214, 2009.

[2] Narayana V.A, P. Premchand and A. Govardhan, "A Novel and Efficient Approach For Near Duplicate Page Detection in Web Crawling", *Proceedings of IEEE International Advance Computing Conference*, pp. 1492-1496, 2009.

[3] Akilandeswari J and Gopalan N. P, "An Architectural Framework of a Crawler for Locating Deep Web Repositories using Learning Multi-agent Systems", *Proceedings of Third International Conference on Internet and Web Applications and Services*, pp. 558-562, 2008.

[4] Tripathy A, Patra P. K, "A Web Mining Architectural Model of Distributed Crawler for Internet Searches Using Page Rank Algorithm", *Proceedings of IEEE Asia-Pacific Services Computing Conference*, pp. 513-518, 2008.

[5] Chunying Chen, Xiongwei Zhou and Jianzhong Zhang, "Web Data Mining System Based on Web Services", *Proceedings of Ninth International Conference on Hybrid Intelligent Systems*, Vol. 3, pp. 216-220, 2009.

[6] Tom Heath, "Information-seeking on the Web with Trusted Social Networks – from Theory to Systems", Ph.D Thesis, Department of Computer Science, 2008.

[7] N. J. Belkin, "Helping People Find What They Don't Know", *Communications of the ACM*, Vol. 43, No .8, pp. 58-61, 2000.

[8] Louis S. Wang, "Relevance Weighting of Multi-Term Queries for Vector Space Model", *IEEE Symposium on Computational Intelligence and Data Mining*, pp. 396-402, 2009.

[9] Huilian Fan, Guangpu Zeng and Xianli Li, "Crawling Strategy of Focused Crawler Based on Niche Genetic Algorithm", *Proceedings of Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, pp. 591-594, 2009.

[10] Mehran Sahami and Timothy D. Heilman, "A Web based Kernel Function for Measuring the Similarity of Short Text Snippets", *Proceedings of the 15th International Conference on World Wide Web*, pp. 377-386, 2006.

[11] Sean A. Golliher, "Search Engine Ranking Variables and Algorithms", *Semj.org*, Supplement Issue, Vol. 1, 2008.

[12] Albert Bifet, Carlos Castillo, Paul-Alexandru Chirita and Ingmar Weber, "An Analysis of Factors Used in Search Engine Ranking", AIRWeb 2005: 48-57. Available at: http://eprints.pascal-network.org/archive/00001583/.

[13] Ahmed C. F, Tanbeer S. K, Byeong-Soo Jeong and Young-Koo Lee, "Efficient Mining of Utility-Based Web Path Traversal Patterns", *Proceedings of International Conference on Advanced Communication Technology*, Vol. 3, pp. 2215-2218, 2009.

[14] Hazem Elmeleegy, Jayant Madhavan and Alon Halevy, "Harvesting Relational Tables from Lists on the Web", *The International Journal on Very Large Data Bases*, Vol. 20, No. 2, pp. 209-226, 2009.

[15] Jianhan Zhu, "Mining Web Site Link Structures for Adaptive Web Site Navigation and Search", Ph.D Thesis, University of Ulster at Jordanstown, 2003.

[16] David Durand and Paul Kahn, "MAPA: A System for Inducing and Visualizing Hierarchy in Websites", *Proceedings of the ninth ACM Conference on Hypertext and hypermedia: links, objects, time and space- structure in hypermedia systems*, pp. 66-76, 1998.

[17] Jaime Arguello, "Vector Space Model", INLS 509: Information Retrieval, Lecture Notes, 2011.

[18] A. B. Manwar, Hemant S. Mahalle , K. D. Chinchkhede and Vinay Chavan , "A vector space model for information retrieval: a matlab approach", *Indian Journal of Computer Science and Engineering*, Vol. 3, No. 2, pp. 222-229, 2012.

[19] S. K. Jayanthi and S. Prema, "Facilitating Efficient Integrated Semantic Web Search with Visualization and Data Mining Techniques", *Proceedings of International Conference on Information and Communication Technologies*, pp. 437 – 442, 2010.

[20] S. K. Jayanthi and S. Prema, "CIMG-BSDS: Image Clustering Based on Bookshelf Data Structure in Web Search Engine Visualization", *Proceedings of International Conference on Recent Trends in Computing, Communication and Information Technologies*, pp. 457-466, 2012.