An Efficient Algorithm Retrieval System using Soft Computing Mechanism

R. Suganya[1], J. Suguna[2]
[1]*Research Scholar,* [2]*Associate Professor,*
[12]*Department of Computer Science, Vellalar College for Women, Erode, Tamilnadu, India*

Abstract- Algorithms are found everywhere in Computer Science literature and offer a precise methodology to resolve problems. From Human Genome project to economics and internet, algorithms have influenced each and every aspect of research. The current state-of-the-art search engines however, are not optimized to search for algorithms. They do not distinguish between documents that contain an algorithm and those that do not. As a result, the search results contain a variety of unwanted and irrelevant documents. The main focus of this paper is to extract the algorithms from the collection of scholarly articles and create an information retrieval system to retrieve the relevant algorithms. It first analyses a document to check for the presence of an algorithm. If there is any algorithm found in the document, the algorithms are extracted from the text documents. Two different techniques are used to retrieve the relevant algorithm from the database such as hard computing and soft computing techniques. In hard computing, cosine similarity method is used to find the similarity between the query and the document collection. In soft computing, Zadeh's min-max operation is used to retrieve the relevant algorithm from the database. Performances of the two methods are compared with each other using some validity measures such as Precision, Recall and F-measure and it is found that the proposed soft computing method yields superior performance than the hard computing method.

Keywords- Cosine similarity, Hard computing, Information retrieval, Soft computing, Similarity measures, Zadeh's min-max operation.

I. INTRODUCTION

The extensive use of distributed information system stores enormous amount of information in the large database belongs to different organization. As a result, huge amount of data is stacked up in the electronic system of companies, government and research institution. The main problem is to infer some precious information from the observed data. The use of computing technology has helped researchers to accumulate very huge volumes of data. Such collections of data, whether their origin is business enterprise or scientific experiment, have recently stimulated a tremendous attention in the areas of knowledge discovery and data mining.

Text is one of the most commonly used multimedia data type in day-to-day use. Text is used for formal exchange of information by common people through electronic mail, Internet chat, World Wide Web, digital libraries, electronic publications, and technical reports, to name a few. Moreover, large volumes of text information exist in the so-called "gray literature" and they are not easily available to the users outside the normal book-selling channels. The gray literature includes technical reports, research reports, theses and dissertations, trade and business literature, conference and journal papers, government reports, and so on [1].

Gray literature is stored in text databases. The assets of information implanted in the vast volumes of text databases distributed all over is enormous, and such databases are rising exponentially with the revolt of current Internet and information technology. The trendy data mining algorithms have been developed to dig out information from well-structured classical databases, such as relational, transactional, processed warehouse data, etc. Multimedia data are not so structured and often less formal. The majority of the textual data spread all over the world are not very formally structured either. The structure of textual data formation and the underlying syntax vary from one language to another language (both machine and human), one culture to another, and possibly user to user.

Text mining can be classified as the special data mining technique mainly appropriate for knowledge and information discovery from textual data. Automatic understanding of the content of textual data, and hence the extraction of knowledge from it, is a long-standing challenge in artificial intelligence. There were efforts to develop models and retrieval techniques for semi structured data from the database community.

Text mining, is also called as text data mining, is the process of extracting meaningful information from large collection of textual data. Information is normally extracted by devising of patterns and trends through means such as statistical pattern learning. The text mining process usually involves parsing the input text, deriving patterns from the structured data, and finally evaluation and interpretation of the output.

Computer science and many of its applications are about developing, analyzing, and applying algorithms. Efficient solutions to important problems in various disciplines other than computer science usually involve transforming the problems into algorithmic ones on which standard algorithms are applied. Standard algorithms are usually collected and cataloged manually in textbooks, encyclopedias, and websites that provide references for computer programmers. Manually searching for the newly published algorithms is a nontrivial task. Researchers and others who aim to discover efficient and innovative algorithms would have to actively search and monitor relevant new publications in their fields of study to keep abreast of latest algorithmic developments.

The objective of the system is to identify and extract the algorithms from the collection of scholarly articles and made it available for the users. Hard computing and soft computing

methods are used to retrieve the relevant algorithms from the database. The performances of both methods are compared with each other.

II. RELATED WORKS

Akram Roshdi, et al., [2] has proposed various models like Boolean model, Vector Space model, and Probabilistic model for information retrieval. Two indexing techniques such as Signature file and Inversion indices are described for reducing search space and various searching techniques like linear search algorithm, Brute force search and binary search algorithm was also narrated. They also provided the overview of traditional IR models.

Vijayarani et al., [1] has discussed about the text mining and its pre-processing techniques. Text mining techniques are used in various types of research domains like natural language processing, information retrieval, text classification and text clustering. They also discussed about various stemming methods like truncating method, statistical method and mixed method. They also described some stop word removal methods like classic model, methods based on Zip's Law, mutual information model and term based random sampling.

Jitendra Nath Singh et al., [3] has proposed three different approaches of Vector Space Model (VSM) such as Term count model, Tf-Idf model and Vector space model based on normalization in order to compute the similarity score of hits from search engine. They provided a clear understanding of the issues and problems in using the vector space model in information retrieval and discussed the main aspects of VSM. Comparison of three approaches of VSM showed that Tf-Idf model provided the good results for long documents as compared to small documents.

Anjali Ganesh et al., [4] discussed that the purpose of stemming is to reduce different grammatical forms or word forms of a word like its noun, adjective, verb, adverb etc. The goal of stemming is to decrease inflectional forms and at times derivationally related forms of a word to a common base form. They also discussed different methods of stemming and their comparisons in terms of usage, advantages as well as limitations. The basic difference between stemming and lemmatization is also discussed.

Gourav Bathla et al., [5] have used the nearest neighbour algorithm with cosine similarity to categorize patents and research papers. The advantage of the approach is that the search area becomes very small and so waiting time of user to get answer of query reduced to a large extent. The experimental results showed that if a user wants to search for the patent or research paper in any particular field or category, then user would get better results. They have calculated threshold based on the similarity of terms between query and research paper or patent.

Bhatia, et al., [6] has proposed a technique for automatic recognition of pseudo-codes (PCs) in Computer Science distributions. Every pseudo-code in the article comes with the subtitle. Such a PC can be finding with the help of pattern arrangement. PCs are an essential piece of computer science literature. However, none of the current search engines offer specialized algorithm search facility. They depicted a vertical internet searcher that quests the PC in articles, recovers and lists the related metadata and literary portrayal of the recognized PC. The extracted metadata information is then used for algorithm ranking in response to user queries. Test result demonstrates the predominance of their framework on other prominent web indexes.

Palakorn Achananuparp et al., [7] has proposed three existing text similarity measures such as Word Overlap Measures, TF-IDF Measures, Linguistic Measures, which was used to calculate similarity score between sentences in many text applications. The evaluation is conducted on three different data sets, TREC9 question variants, Microsoft Research paraphrase corpus, and the third recognizing textual entailment data set. They investigated the performance of several classes of sentence similarity measures on multiple sentence pair data sets. Their result suggested that TF-IDF measures are superior in identifying paraphrases than word overlap and linguistic measures.

III. SYSTEM ARCHITECTURE

The objective of the system is to identify and extract the algorithms from the given document collection and made it available for the users to search for the relevant algorithm. It is shown in Fig. 1.

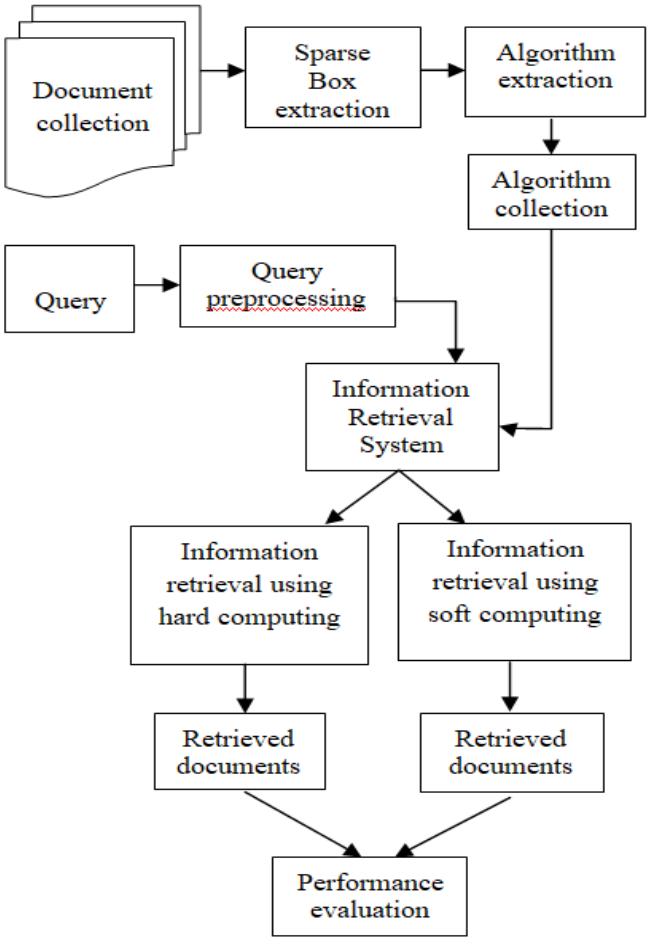

Fig.1: System Architecture

A. Algorithm extraction

Algorithm extraction is the first process which involves the following steps:

Step1: Convert all pdf documents into text document.

Step 2: Search for the keywords algorithm, pseudo-code, procedure on each and every document in the document collection

Step 3: If there is any such keywords present in the text document, select 30 lines following those keywords

Step 4: Algorithms are extracted from the document using the feature extraction methods namely Sparse box extraction and Writing style of algorithm

- **Sparse Box Extraction**: A sparse box is a set of n consecutive sparse lines [7]. A sparse line can be detected with the help of the following formula

$$\text{Sparse} = \frac{\text{Number of non} - \text{space characters}}{\text{Average number of characters per line}}$$

If the sparse value is less than the threshold then the line is said to be sparse line. Fig. 2 shows the extracted text lines which are sparse lines (highlighted in yellow), and non-sparse lines (highlighted in white).

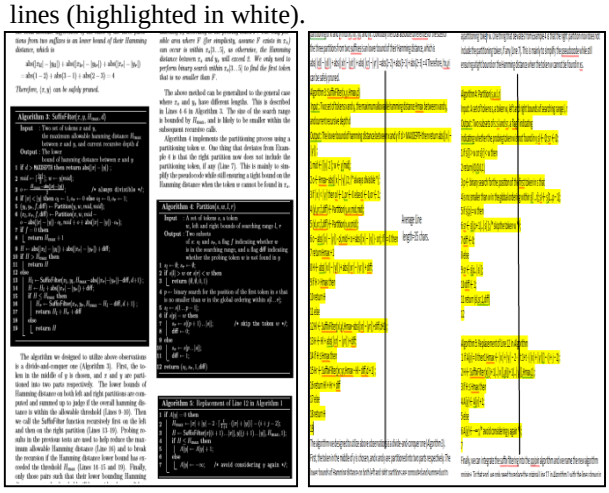

Figure.2 Sparse Box Extraction

- **Writing style of algorithm:** Algorithms are generally written in a programming style, with details omitted. Symbols, Greek letters, mathematical operators, and programming keywords (such as 'for', 'begin', 'end', 'return', etc.) are usually used to compose algorithms [7]. Therefore if the sparse box contains more number of special characters and programming keywords, then it is said to be an algorithm.

Step 5: The extracted lines are then stored as a text document for further processing.

B. Document representation and query preprocessing

Document Representation:- The text document must be represented in some model in order to use it for information retrieval process. The vector model is widely used model in information retrieval and text mining. Text document can be represented as a vector of words, where words are assumed to appear independently and the order is immaterial. Each word corresponds to a dimension in the resulting data space and each document then becomes a vector consisting of non-negative values on each dimension. With documents presented as vectors, it is easy to measure the degree of similarity of two documents as the correlation between their corresponding vectors, which can be further quantified as the cosine of the angle between the two vectors.

Query Preprocessing and Representation:- Initially, the stop words are removed from the query. There are words that are non-descriptive for the topic of a document, such as a, and, are, do, etc. Then, words were stemmed using Porter's suffix-stripping algorithm, so that words with different endings will be mapped into a single word. Finally, all uppercase letters are converted into lowercase letters. The pre-processed query is now represented as a vector.

C. Term document matrix representation.

Term Frequency:- Term Frequency is used to measure how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones. Thus, the term frequency is often divided by the document length (the total number of terms in the document) as a way of normalization.

TF (t) = (Number of times term t appears in a document) / (Total number of terms in the document).

Normalized Term Frequency:- Normalized Term Frequency must be calculated for each and every document based on its size. The way to normalize the document is

$$\text{Normalize term frequency} = \frac{\text{Term Frequency}}{\text{Total number of terms}}$$

Inverse Document Frequency:- Inverse Document Frequency (IDF) is used to find the weight of every term in a document. IDF can be computed with the help of following formula.

$$\text{IDF} = 1 + \log_e \frac{\text{Total no. of documents}}{\text{No. of documents with term in it}}$$

Term Frequency- Inverse Document Frequency:- TF-IDF stands for term frequency-inverse document frequency, and the TF-IDF weight is a weight often used in information retrieval and text mining. This weight is a statistical measure used to evaluate how important a word is to a document in a collection or corpus. The importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. Variations of the tf-idf weighting scheme are often used by search engines as a central tool in scoring and ranking a document's relevance given a user query.

$$\text{Weight (t, D)} = \text{tf (t, D)} * \text{idf (t)}$$

Term Document Matrix:- A **term-document matrix** is a mathematical matrix that describes the frequency of terms that occur in a collection of documents. In a document-term matrix, rows correspond to documents in the collection and columns correspond to terms. The calculated TF-IDF values are represented as term document matrix.

D. Similarity measures

Cosine Similarity Measure:- Cosine similarity is one of the most popular similarity measure applied to text documents, such as in numerous information retrieval applications and clustering too. When documents are represented as term vectors, the similarity of two documents can be easily calculated. This is quantified as the cosine of the angle between vectors, that is, the so-called cosine similarity. Given a document (D) and query (Q), their cosine similarity is

	No. of documents retrieved	
Query	**Cosine Similarity**	**Fuzzy Similarity**
1	14	18
2	18	19
3	10	13
4	9	12
5	14	16

$$\text{Similarity} = \frac{\square.\square}{\|D\|.\|Q\|} = \frac{\sum_{i=1}^{n} D_i\, Q_i}{\sqrt{\sum_{i=1}^{n} D_i^2}\ \sqrt{\sum_{i=1}^{n} Q_i^2}}$$

Fuzzy Similarity Measure - Zadeh's min-max measure

The Zadeh's min-max model tries to soften the Boolean operators by considering the query-document similarity to be a linear combination of the *min* and *max* document weights [9]. For a document DOC with index-term weights doc_{A1}, doc_{A2}, doc_{A3}, …, doc_{An} for the each terms and the queries,

Q_{or}= (A_1 or A_2 or A_3 or… or A_n)

Q_{and}= (A_1 and A_2 and A_3 and… and A_n)

The similarity between the query (Q) and document (DOC) using Min-Max Model can be calculated as

$$\square\square\square\square(\square_{\square\square}, \square\square\square) = \square_{\square\square 1} * \max(\square\square\square_{\square 1}, \square\square\square_{\square 2}, .\ \square\square\square_{\square\square}) + \square_{\square\square 2} * \min(\square\square\square_{\square 1}, \square\square\square_{\square 2} \ldots . \square\square\square_{\square\square})$$

$$\square\square\square\square(\square_{\square\square\square}, \square\square\square) = \square_{\square\square\square 1} * \max(\square\square\square_{\square 1}, \square\square\square_{\square 2}, .\ \square\square\square_{\square\square}) + \square_{\square\square\square 2} * \min(\square\square\square_{\square 1}, \square\square\square_{\square 2}, \ldots \square\square\square_{\square\square})$$

Where C_{or1} and C_{and1} are the softness coefficients.

IV. RESULTS AND DISCUSSION

A. Data sources

CiteSeer is a public search engine and digital library and repository for scientific and academic papers primarily with a focus on computer and information science. It currently has over 6 million documents with nearly 6 million unique authors and 120 million citations. 200 documents are randomly collected from the repository and used for the proposed work.

B. Algorithm extraction and retrieval

All PDF documents are converted into text documents. Algorithms are extracted from the text documents and stored in a database. Then, user requested algorithms are retrieved from the database using cosine similarity measure and fuzzy similarity measure. Number of documents retrieved by both methods is shown in Table 1. Fig.3 represents the number of documents retrieved by two different methods.

Table.1 Number of documents retrieved

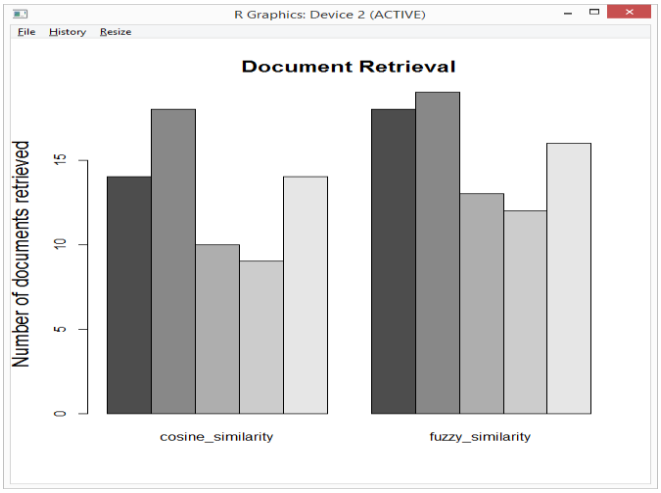

Fig.3: Number of documents retrieved

Evaluation metrics

The accuracy of two different methods is calculated by using the evaluation metrics Precision, Recall and F-measure.

Precision

Precision is the fraction of retrieved documents that are relevant to the query [9]. Table.2 represents the Precision values for five different queries.

$$\text{Precision} = \frac{\{\text{REL}\} \cap \{\text{RET}\}}{\{\text{RET}\}}$$

where REL is relevant document and RET is retrieved document. Precision measures the system's ability to reject any non-relevant documents in the retrieved set. Fig.4 represents the comparison between the precision values of two different similarity measures.

Table.2 Precision values

Query	Cosine Similarity	Fuzzy Similarity
1	0.74	0.91
2	0.77	0.85
3	0.79	0.88
4	0.69	0.84
5	0.75	0.92

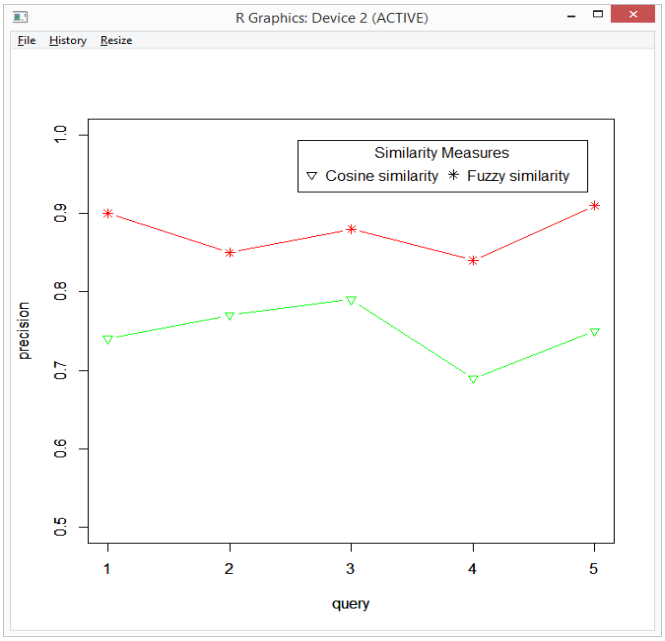

Fig.4: Precision Vs Query

Recall

Recall is the fraction of the relevant documents that are successfully retrieved [9]. Table.3 represents the Recall values for five different queries.

$$\text{Recall} = \frac{\{\text{REL}\} \cap \{\text{RET}\}}{\{\text{REL}\}}$$

where REL is relevant document and RET is retrieved document. Recall measures the ability of the system to find all the relevant documents. Fig.5 represents the comparison between the Recall values of two different similarity measures.

Table.3 Recall values

Query	Cosine similarity	Fuzzy Similarity
1	0.49	0.69
2	0.52	0.61
3	0.51	0.62
4	0.47	0.60
5	0.43	0.73

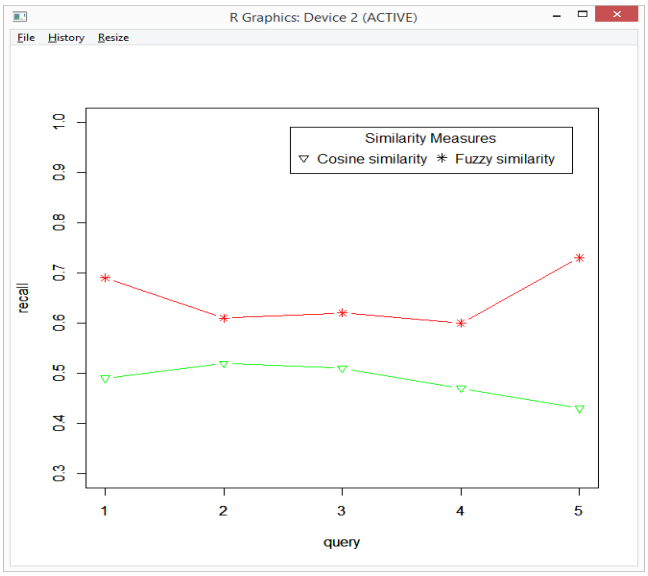

Fig.5: Recall Vs query

F-Measure

A measure that combines precision and recall is the harmonic mean of precision and recall. The traditional F-measure or balanced F-score:

$$\text{F} - \text{measure} = 2.\frac{\text{precision}.\,\text{recall}}{\text{precision} + \text{recall}}$$

It is high only when both recall and precision are high. It is equivalent to recall when weight is 0 and precision when weight is 1. The F-measure is 0 when no relevant documents have been retrieved, and is 1 if all retrieved documents are relevant and all relevant documents have been retrieved. Fig.6 represents the comparison between the F-Measure values of two different similarity measures.

Table.4 F-Measure values

Query	Cosine Similarity	Fuzzy Similarity
1	0.61	0.88
2	0.64	0.81
3	0.65	0.82
4	0.54	0.80
5	0.64	0.81

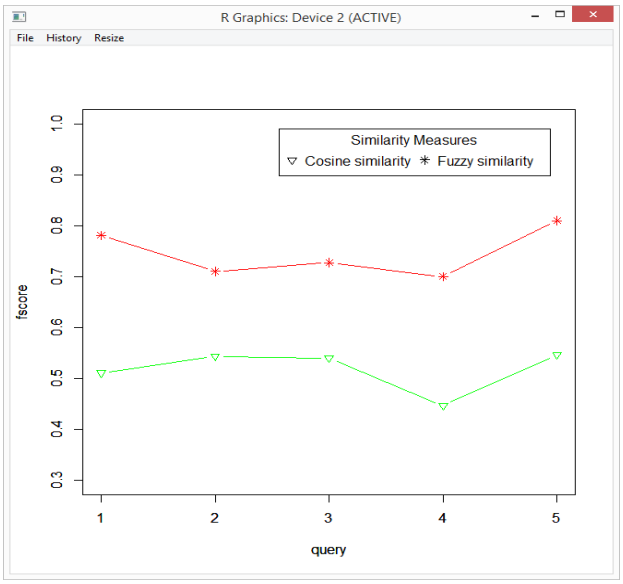

Fig.6: F-Measure Vs Query

V. CONCLUSION

In the proposed work, the algorithms are extracted from the scholarly articles and the search engine for algorithms was developed with the help of two methods such as hard computing and soft computing methods. The results of the two methods are compared and evaluated using external validity measures. The experimental result indicates that the soft computing method which uses Zadeh's min-max operation yields the better result.

VI. ACKNOWLEDGMENTS

I wish to extend my sincere thanks to all our Teaching and Non-Teaching faculties of the Department of Computer Science, Vellalar College for Women (Autonomous), Erode for their timely help at every stage of my thesis work. I express my heartfelt gratefulness and special thanks to my family members who have acted as a backbone throughout the research work. Last but not least I would like to thank my friends and well-wishers who have helped me to complete the thesis work completely.

VII. REFERENCES

[1]. Vijayarani, Ilamathi, Nithya, "Preprocessing Techniques for Text Mining - An Overview", *International Journal of Computer Science and Communication Networks*, Vol-5, pp.7-16, 2013.

[2]. S. Mitra , T. Acharya, *Data Mining Multimedia, Soft Computing, and Bioinformatics*, A John Wiley & Sons publication, United States of America, 2003.

[3]. A. Roshdi, A. Roohparvar, "Review: Information Retrieval Techniques and Applications", *International Journal of Computer Networks and Communications*, Vol.3, No. 9, pp.373 – 377, 2015.

[4]. J. N. Singh, S. K. Dwivedi, "Analysis of Vector Space Model in Information Retrieval", *IJCA Proceedings on National Conference on Communication Technologies & its impact on Next Generation*, Vol. CTNGC, No. 2, pp.14-18, 2012.

[5]. Jivani, A. Ganesh, "A Comparative Study of Stemming Algorithms", *International Journal of Computer Technology and Applications*, Vol. 2, No. 6, pp.1930-1938, 2011.

[6]. G. Bathla, R. Jindal, "Similarity Measures of Research Papers and Patents using Adaptive and Parameter Free Threshold", *International Journal of Computer Applications*, Vol.33, No. 5, pp.9-13, 2011.

[7]. S. Bhatia, S. Tuarob, P. Mitra, C. Lee Giles, "An Algorithm Search Engine for Software Developers", *International Conference on Software Engineering, USA*" pp. 13 - 16, 2011.

[8]. Palakorn Achananuparp, Xiaohua Hu, Shen Xiajiong, "The Evaluation of Sentence Similarity Measures", *10th International Conference on Data Warehousing and Knowledge Discovery, Turin, Italy*, pp. 305 - 316, 2008.

[9]. K. Vivekanandan, J. Suguna, "Fuzzy Similarity Measure for Document Retrieval", *Journal of Advanced Research in Computer Engineering*, Vol.1, pp. 25-32, 2007.