A Review on Text Similarity Technique used in IR and its Application

Nitesh Pradhan
M.Tech Scholar Computer Science Department
Maulana Azad National Institute of Technology
BHOPAL (M.P)

Manasi Gyanchandani, PhD
Asst. Professor Computer Science Department
Maulana Azad National Institute of Technology
BHOPAL (M.P)

Rajesh Wadhvani, PhD
Asst. Professor Computer Science Department
Maulana Azad National Institute of Technology
BHOPAL (M.P)

ABSTRACT
With large number of documents on the web, there is a increasing need to be able to retrieve the best relevant document. There are different techniques through which we can retrieve most relevant document from the large corpus. Similarity between words, sentences, paragraphs and documents is an important component in various tasks such as information retrieval, document clustering, word-sense disambiguation, automatic essay scoring, short answer grading, machine translation and text summarization. Text similarity means user's query text is matched with the document text and on the basis on this matching user retrieves the most relevant documents. Text similarity also plays an important role in the categorization of text as well as document. We can measure the similarity between sentences, words, paragraphs and documents to categorize them in an efficient way. On the basis of this categorization, we can retrieve the best relevant document corresponding to user's query. This paper describes different types of similarity like lexical similarity, semantic similarity etc.

General Term
Text Similarity, Text Mining, Text Summarization

Keyword
Text similarity, Lexical similarity, semantic similarity, Corpus based similarity and Knowledge based similarity.

1. INTRODUCTION
Information Retrieval is an activity of obtaining information resources relevant to an information need from a collection of information resources. Information Retrieval has different type of applications out of these applications, Blog search is one of the important application. The searching can be done on the basis of similarity. Similarity is a process through which we determine the relationship between text snippets. Text similarity is defined in two ways; these are lexical similarity and semantic similarity. Lexical similarity provide the similarity on the basis of character or statement matching, for e.g. "Put" and "Cut" are lexical similar to each other. Whereas Semantic similarity provide the similarity on the basis of meaning, for e.g. "Support Vector Machine" and "SVM" both are semantic similar to each other. There are several applications or areas where we use the text similarity; these areas are Information retrieval, clustering, text categorization, topic detection, question answer session, machine translation, text summarization etc.

The further sections are as follows: Second section describes all the Lexical similarity measure technique. All the Semantic similarity technique is describe in third section. Fusion similarity and conclusion is described in fourth and fifth section respectively.

2. LEXICAL SIMILARITY

In Lexical similarity [19] provides the similarity on the basis of character and statement matching. Lexical similarity is a measure of the degree to which word set of two given string are similar. A Lexical similarity of 1(means 100%) would mean a total overlap between words, Whereas Lexical similarity of 0 means there are no common word in given string.

This survey represents the most popular lexical similarity measure which was implemented in SimMetrics [1] package. Lexical similarity is categorized in Character based similarity and statement based similarity. In character based similarity four different algorithms are described and in statement based similarity five different algorithms are described as shown in Figure 1.

2.1 Character based similarity

2.2 .1 Longest common subsequence (LCS) Similarity

LCS [2] matching is a commonly used technique to measure the similarity between two string (i, j). LCS measure the longest total length of all the matched substring between two string where these sub-string appear in the same order as they appear in the other string. LCS similarity of Given Two string (i, j) will be

$$LCS(i,j) = \begin{cases} 0 & if\ i = 0\ or\ j = 0 \\ 1 + LCS(i-1, j-1) & if\ x[i] == y[j] \\ max\begin{cases} LCS(i, j-1) \\ LCS(i-1, j) \end{cases} & if\ x[i] \neq y[j] \end{cases}$$

Longest common subsequence is based on dynamic programming approach which takes O(n). LCS represents a distance matrix and can be used for indexing in database but the problem with LCS is space complexity. LCS uses recursion approach which uses stack that takes lots of space

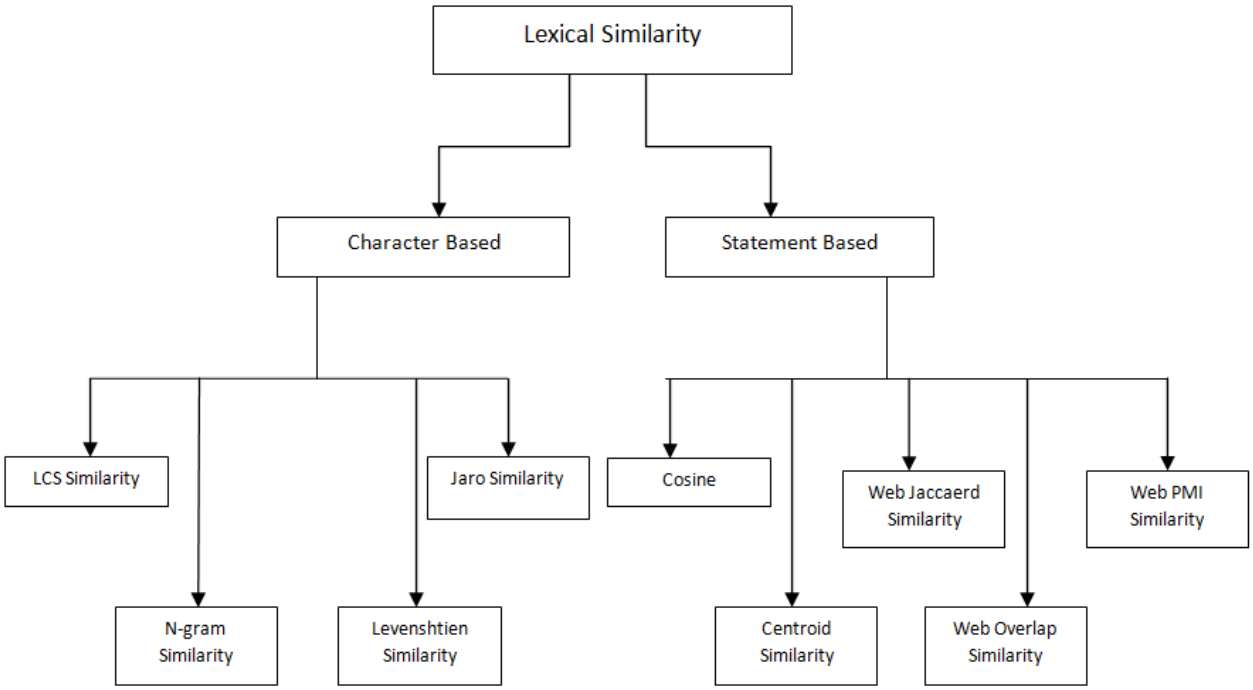

Figure 1: Categorization of Lexical Similarity

2.2.2 N-gram similarity

If there is a sequence of text is given, the N-gram [3] is a technique through which we can determine the similarity of sub-sequence of n items from given text sequence. In N-gram similarity technique, we compute the similarity on the basis of distance between each character in two strings. This distance is computed by dividing the number of similar grams by maximal number of n-grams.

$$P(w) = P(w_1)P(w_2/w_1)P(w_3/w_2, \dot{w_1}) \dots\dots\dots\dots$$

Where w, w1, w2, w3…….are different words. The N-gram similarity technique use to design kernels that allow machine learning algorithm such as support vector machine to learn from string data. The performance of N-gram similarity technique is high but the accuracy is very less.

2.2.3 Levenshtein distance similarity

The Levenshtein distance [4] technique also use the distance factor to calculate the similarity between given two string. In actual, this distance is counting the minimum number of operation needed to transform one string into other string. The Levenshtein distance between two string a, b is given by $lev_{a,b}(|a|, |b|)$

$$lev_{a,b}(i,j) = \begin{cases} max(i,j) & if\ min(i,j) = 0 \\ min\begin{cases} lev_{a,b}(i-1,j)+1 \\ lev_{a,b}(i,j-1)+1 \\ lev_{a,b}(i-1,j-1)+1 \end{cases} & otherwise \end{cases}$$

Where i, j are the index for word a, b respectively. The operation may be insertion, deletion or substitution of a single character. This operation takes constant time. Levenshtein distance similarity gives best result in case of short string but in case of long string cost of levenshtein distance is same as the length of string.

2.2.4 Jaro distance

Jaro similarity [5] technique determines the similarity between two strings on the basis of common character. This technique is used mainly in the area of duplicate detection. The Jaro distance (d_j) of two string s1 and s2 is

$$d_j = \begin{cases} 0 & if\ m = 0 \\ \frac{1}{3}\left[\frac{m}{|S_1|} + \frac{m}{|S_2|} + \frac{m-t}{|m|}\right] & otherwise \end{cases}$$

Where m = number of matching character
t = half the number of transpositions

$$t = \left\lfloor \frac{max(|S_1|.|S_2|)}{2} \right\rfloor - 1$$

Jaro distance gives better result in case of hybrid method. If the data size is too much large then jaro distance similarity not gives efficient results.

2.2 Statement based similarity

2.2.1 Cosine similarity

Cosine similarity [16] is widely used approach to find the similarity between two texts. To find the similarity between two texts, each text is represented in the form of vector. Each word in text defines a dimension in Euclidean space and the frequency of each word corresponds to the value in the dimension.

The Cosine similarity between two text (t_1,t_2)

$$SIM(t_1,t_2) = \frac{\sum_{i=1}^{n} t_{1_i} t_{2_i}}{\sqrt{\sum {t_{1_i}}^2} \times \sqrt{\sum {t_{2_i}}^2}}$$

Cosine similarity is domain-depended and easy to implement but reduction of cosine similarity is not substantial.

2.2.2 Centroid based similarity

Centroid based similarity [17] is a statement based similarity in which we form vector of each statement of a document. This similarity technique only considers the salient word for the distance between a sentence and the entire document. In this technique, two main conditions are checked. First one is for each word in a sentence S_i is checked to see if it is occur in query q and second is TFIDF value of this word is greater than a predefine threshold.

$$Score_{centroid}(i) = \sum_{w_j \in S_j} bool(w_j \in T) * bool(t_w(w_j) > v) * t_w(w_j)$$

If both conditions are true then term weight of this word is added to the Centroid score for the sentence. Beyond similarity measurement, Centroid method used to summarize the text and cluster the document. For Centroid similarity requires TF-IDF value of each word which is very time consuming.

2.2.3 Web Jaccard Similarity

This is a count based co-occurrence measure technique. Web jaccard similarity [6, 7] is used to find the similarity between words. Web jaccard coefficient can be computed based on number of element in the intersection set divided by the number of element in the union set.

$$Web\ Jaccard\ (P,Q) = \begin{cases} 0 & if\ H(P \cap Q) \leq C \\ \dfrac{H(P \cap Q)}{H(P) + H(Q) - H(P \cap Q)} & otherwise \end{cases}$$

Where H(P) and H(Q) are the page count for query P&Q respectively and C is predefine threshold. It's suitability would depend on the statistics of document size and the preprocessing done on their contexts so if the document set is large in size then web jaccard similarity technique not produce an appropriate result.

2.2.4 Web Simpson similarity

Web Simpson [21] measure is a count based measure in which Web Simpson coefficient consider two string a full match if one is a subset of the other. If P and Q are two query then Web Simpson (P, Q) are define as –

$$Web\ Simpson(P,Q) = \begin{cases} o & if\ H(P \cap Q) < C \\ \dfrac{H(P \cap Q)}{\min(H(P), H(Q))} & otherwise \end{cases}$$

In Web Simpson similarity, non-significant word is removed so that they will not interfere during retrieval and size of total text can be reduced between 30% and 50%. Web Simpson similarity technique use for original query suggestion and that may be totally off topic.

2.2.5 Web PMI similarity

PMI is stand for Point Wise Mutual Information. Web PMI similarity [8] measure is intended to reflect the dependence between two probabilistic events. The basic application of Web PMI is in information theory and statistics.

$$Web\ PMI(P,Q) = \begin{cases} o & if\ H(P \cap Q) < C \\ \log_2\left(\dfrac{H(P \cap Q)}{H(P)\,H(Q)}\right) & otherwise \end{cases}$$

Where P, Q are two string. PMI similarity results in performance comparable or better (between 62.5% and 73.75%) then Latent Semantic similarity. PMI similarity accuracy of 81.25% with a window size of 16 to 32 words but we require predefine threshold to make the cluster of words using Web PMI similarity techniques.

3. SEMANTIC SIMILARITY

Semantic similarity [11] determines the similarity between text and document on the basis of their meaning rather than character by character matching. Semantic similarity is computed on the basis of corpus based and knowledge based measures. Each of these measures describe in figure 2.

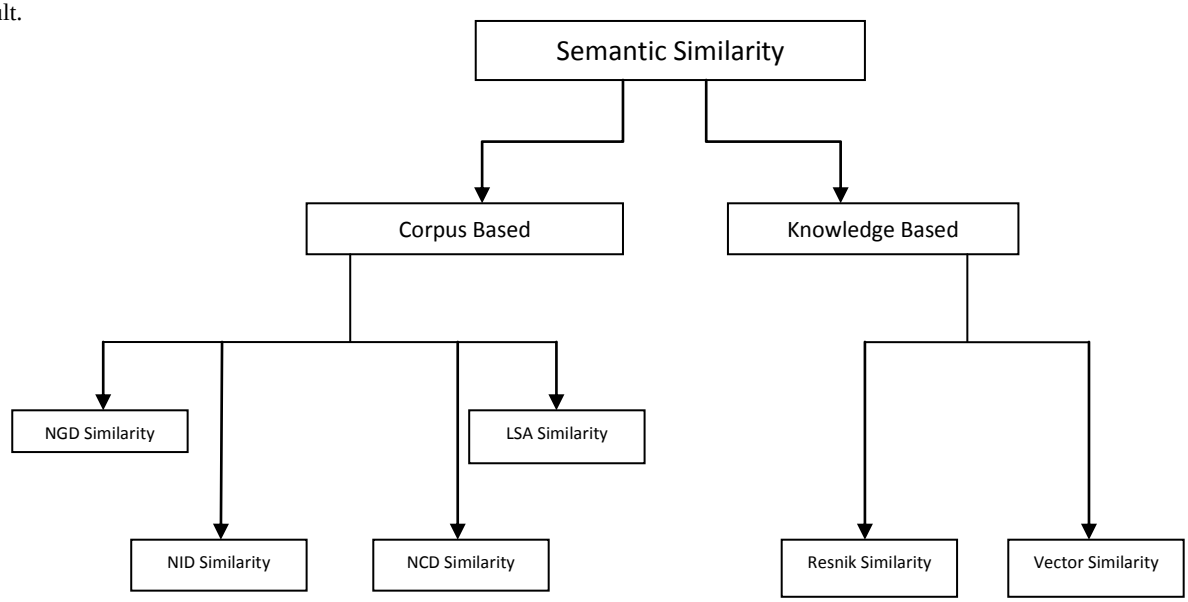

Figure 2: Categorization of Semantic Similarity

3.1 Corpus based Similarity

This is a semantic based similarity, in which words are similar on the basis of their meaning. Corpus based similarity [20] determine the similarity between word on the basis of information gained from large corpus. This large corpus contains different type of document of different knowledge domain.

3.1.1 Normalized Google Distance (NGD)

NGD [14] similarity is a corpus based similarity which compute similarity between keyword on the basis of number of hits returned by Google search engine for a set of keyword.

The normalized Google distance between two search keyword x and y is defined as:-

$$NGD(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log N - \min\{\log f(x), \log f(y)\}}$$

Where N is the total number of web pages search by Google search engine. F(x) and f(y) are the number of hits for search term x and y respectively, and f(x, y) is the number of web pages on which both x and y occur.

Following are the properties of NGD algorithm:

1. The range of NGD scores from 0 to ∞.
2. NGD score is always non-negative such that NGD(x, x)=0
3. NGD gives us an objective semantic relation between search terms, it needs to become stable when the number N grows unboundedly.

If total number of Web pages (N) grows unboundedly then results of NGD similarity will not be stable.

3.1.2 Normalized Information Distance (NID)

Information distance [12, 13] is define as for two string x, y the length of the shortest binary program in the reference universal computing system such that the program computes output y from input x and also compute output x from input y. The Information distance is denoted by E(x, y).

$$E(x,y) = \{k(x,y) - \min\{k(x), k(y)\}\}$$

The NID (Normalized Information Distance) has value between *0* and *1*, and expresses that similarity on this scale, where *0* being the same and *1* being completely different.

NID uses a special function called *Kolmogorov Complexity K(x)*[22], where *x* is the string. The *Kolmogorov Complexity K(x)* of a string *x* is the length, in bits, of the shortest computer program of the fixed reference computing system that produce *x* as output. It is mostly viewed as the length, in bits, of the ultimate compressed version from which *x* can be recovered by a general decompression program. Some of the compressor programs taken in use for *K(x)* are Gzip, bzip2, ppmz. *K(x)* gives the ultimate value of the length of a compressed version of x, and our task is to design better and better compressors.

So finally NID for string x and y after calculate Kolmogorov complexity is:

$$NID(x,y) = \frac{k(x,y) - \min\{k(x), k(y)\}}{\max\{k(x), k(y)\}}$$

Normalized information distance similarity is capable for robust to noise. In NID similarity knowledge about similar records, in their about identical values can't be used during cleaning.

3.1.3 Normalized Compression Distance (NCD)

The drawback of NID measure is that if Kolmogorov complexity is incomputable then NID is also incomputable. So we introduce a Compression technique [15, 23] to define a computable function from strings to the length of the compressed version of those strings. If C is a compressor and we use C(x) to denote the length of the compressed version of a string x, then the Normalized Compressor Distance is:

$$NCD(x,y) = \frac{c(x,y) - \min\{c(x), c(y)\}}{\max\{c(x), c(y)\}}$$

With the help of NCD similarity algorithm we can perform the classification and clustering of natural data but in NCD similarity, we require different type of compressor for e.g. "gzip", "bzip" and "ppmz" etc.

3.1.4 Latent Semantic similarity (LSA)

It is the most popular technique of Vectorial semantics. It is assumed by LSA similarity [18] that if two words whose meaning is similar, will be occur in similar piece of text. A matrix is formed in LSA measure, in which row present the unique word and colon represent the document or paragraph. To reduce the size of matrix, we use Singular Value Matrix (SVM) approach. After formed the matrix, we use cosine similarity or web jaccard similarity algorithm to get the similarity between words.
LSA similarity is faster compare to other dimensionality reduction models. LSA similarity model is not humanly readable.

3.2 Knowledge Based Similarity

Knowledge based similarity [10] is a semantic based similarity that work identify the degree of similarity between words using information derived from semantic networks. The popular semantic network is "Word Net" and Natural language Toolkit (NLTK) to measure the knowledge based similarity between words. Knowledge based similarity also provide the similarity on the basis of word relatedness.

3.2.1 Resnik Similarity

It is a knowledge based similarity which is based on Information Content. The measure value of Resnik similarity [9] is equal to the Information Content (IC) of the most informative subsume. The Resnik similarity returns the Information Content of the LCS of two concepts.

Sim_{res} = -log P(c) (LCS)

Where P(c) is the probability of encounting an instance of concept c in a large corpus. Resnik similarity uses empirical information from corpora. Resnik similarity works on Word net noun only.

3.2.2 Vector similarity

In vector similarity, a matrix is formed to determine the similarity between words. A vector is create for each word used in the Word Net glosses from a given corpus and then represents each concept with a vector that is the average of this co-occurrence vector.
In this similarity technique, term weight is not in binary so complexity is less. Vector similarity technique allows computing a continuous degree of similarity between queries and documents and also allows partial matching between words.

4. Fusion similarity measure

Fusion similarity measure describes the combine study of kernel based similarity and cosine based similarity. A kernel based similarity technique has introduced the concept of fusion similarity. In this kernel based similarity a query expansion QE(x) function is introduced.
In this approach each snippet is treated as a query to a web search engine to find a number of documents. Then we use this returned document to create a context vector. Such context vector can now be more robust compare with cosine similarity. Kernel similarity is work as follows:

1. Let x is a query to a search engine.
2. Let R(x) be the set of n retrieved document d_1, d_2, d_3….d_n.
3. Compute TF-IDF term vector v_i for each document $d_i \in R(x)$.

4. Trim each vector v_i to include its n highest weight.

5. Calculate C(x) be the Centroid of normalized vector v_i

$$C(x) = \frac{1}{n}\sum_{i=1}^{n}\frac{v_i}{\|v_i\|^2}$$

6. Let QE(x) be the normalization of Centroid C(x)

$$QE(x) = \frac{C(x)}{\|C(x)\|^2}$$

This algorithm apply for query x. Now same algorithm applies for query y, and find

$$k(x,y) = QE(x)\,.\,QE(y)$$

If $k(x,y) \geq 0.5$ then x and y are similar otherwise $K(x,y) \leq 0.3$ then x and y are dissimilar.

Text 1	Text 2 (Acronyms)	Kernel	Cosine
Support Vector Machine	SVM	0.812	0.0
Portable Document Format	PDF	0.732	0.0
Artificial Intelligence	AI	0.831	0.0
Artificial Insemination	AI	0.391	0.0
Inverse Document Frequency	IDF	0.831	0.0
Extensible Markup Language	XML	0.731	0.0
	Individuals and their positions		
UN Secretary-General	Kofi Annan	0.825	0.0
UN Secretary-General	George W. Bush	0.110	0.0
US President	George W. Bush	0.688	0.0
Microsoft CEO	Steve Ballmer	0.837	0.0
Microsoft CEO	Bill Gates	0.317	0.0
Microsoft Founder	Bill Gates	0.677	0.0
Google Founder	Bill Gates	0.096	0.0
Microsoft Founder	Larry Page	0.189	0.0
Web Page	Larry Page	0.123	0.5

Table 1: Kernel based similarity with Cosine similarity

5. CONCLUSION

In this survey paper, two types of similarity were discussed: lexical similarity and semantic similarity. The lexical similarity determines the similarity between word and text based on character by character matching. Nine algorithms were summarized; four of them are character based and remaining are statement based similarity. Semantic similarity determines the similarity between word and text based on their meaning. Six algorithms were reviewed; four of them are corpus based similarity and remaining are knowledge based similarity.

After describing different Lexical similarity and Semantic similarity algorithms, we discussed one fusion similarity measure which is a hybrid approach with the combination of kernel based similarity and cosine based similarity. As shown in Table 1, we can conclude that kernel based similarity gives better results as compared to cosine similarity. Semantic similarity can be used in biomedical ontology and can be applied to find similar geographic features.

6. REFERENCES

[1] Chapman, “SimMetrics: a java & c#.net library of similarity metrics”, http://sourceforge.net/projects/simmetrics, 2006.

[2] W. Irving, C.B. Fraser, ”Two algorithms for the longest common subsequence of three (or more) strings, Proceedings of the 3rd Annual Symposium on Combinatorial Pattern Matching”, pp. 214-229,1992.

[3] Alberto, B.Paolo, R.Eneko A. & Gorka L, “Plagiarism Detection across Distant Language Pairs”, In Proceedings of the 23rd International Conference on Computational Linguistics, pp 37–45, 2010.

[4] Navarro, Gonzalo, "A guided tour to approximate string matching", ACM Computing Surveys 33 (1): pp 31–88, 2001.

[5] Cohen, W. Ravikumar, P. Fienberg, "A comparison of string distance metrics for name-matching tasks", KDD Workshop on Data Cleaning and Object Consolidation, pp 73–8, 2003.

[6] Manusnanth, Panyamee, Somjit Arj-in, “Document clustering results on the semantic web search”, In Proceedings of The 5th National Conference on Computing and Information Technology, 2009.

[7] Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des Jura", Bulletin dela Société Vaudoise des Sciences Naturelles, pp 547-579, 1901.

[8] Turney, "Mining the web for synonyms PMIIR versus LSA on TOEFL" In Proceedings of the Twelfth European Conference on Machine Learning (ECML), 2001.

[9] Resnik, "Using information content to evaluate semantic similarity", In Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1995.

[10] Budanitsky, Hirst, "Semantic distance in Word-Net: An experimental application-oriented evaluation of five measures". In Proceedings of the NAACL Workshop on Word-Net and Other Lexical Resources, 2001.

[11] Sahami, Mehran Sahami, 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.

[12] M.Li, P.M.B.Vitanyi, "An Introduction to Kolmogorov Complexity and Its Applications", 2nd Ed., Springer-Verlag, New York, 1997.

[13] M.Li, J.H.Badger, X.Chen, S.Kwong, P. Kearney, and H. Zhang, "An information-based sequence distance and its application to whole mitochondrial genome phylogeny", Bioinformatics, pp 149–154 2001.

[14] Cilibrasi, Rudi L. Cilibrasi, Paul M.B. Vitanyi, "The Google Similarity Distance", in IEEE Transactions on Knowledge and Data Engineering, pp: 370-383, 2007.

[15] H. Muir," Software to unzip identity of unknown composers", New Scientist, 12 April 2003.

[16] Gang Qian, Shamik Sural, Yuelong Gu, Sakti Pramanik, "*Similarity between euclidean and cosine angle distance for nearest neighbor queries*", Proceedings of ACM Symposium on Applied Computing, 2004.

[17] Radev Dragomir Radev, Hongyan Jing, and Malgorzata Budzikowska, "*Centroid-based summarization of multiple documents: sentence extraction, utility-based evaluation, and user studies*", in NAACL-ANLP Workshop on Automatic summarization, 2000.

[18] Susan T. Dumais, "Latent Semantic Analysis". *Annual Review of Information Science and Technology* 2005.

[19] Rensch, Calvin R. "Calculating lexical similarity", In Eugene H. Casad (ed.), *Windows on bilingualism*, pp 13-15 1992.

[20] Rada Mihalcea, Courtney Corley, and Carlo Strapparava. "Corpus-based and Knowledge-based Measures of Text Semantic Similarity", In Proceedings of AAAI, Boston, July, 2006.

[21] Manusnanth Panyamee, and Somjit Arj-in, "Document clustering results on the semantic web search", In Proceedings of The 5th National Conference on Computing and Information Technology, King Mongkut's University of Technology, 2009.

[22] Mahyuddin, "Kolmogorov Complexity: Clustering Objects and Similarity", Mathematic Department, In Proceedings of the 23rd International Conference on Computational Linguistics, 26 July 2012.

[23] Rudi Cilibrasi, Paul M.B. Vitanyi, "Normalized Web distance and word similarity", in NAACL-ANLP Workshop on Automatic summarization, May 2009.

7. AUTHORS PROFILE

NITESH PRADHAN, is currently pursuing M.Tech in Advanced Computing (Dept. Of Computer Sc. & Engineering) from Maulana Azad National Institute of Technology, Bhopal, (M.P) India.

DR. MANASI GYANCHANDANI, is currently working as an asst. professor in the Computer Science Department, Maulana Azad National Institute of Technology Bhopal (M.P) India. She has around 15 years of teaching experience and her area of interest includes Artificial Intelligence, Neural Networks and Intrusion Detection Systems and Information Retrieval. She has published paper in several International and national conferences and journals.

Dr. Rajesh Wadhvani, is currently working as an Asst. Professor in the Computer Science Department, Maulana Azad National Institute of Technology Bhopal, India. He has more than 12 years of teaching experience and has guided more than 12 M.Tech scholars. His research area includes domains of information retrieval, data mining and digital image processing.