High Dimensional Feature Based Word Pair Similarity Measuring For Web Database Using Skip-Pattern Clustering Algorithm

Dr.C.Senthilkumar
Associate Professor, Department of Computer Science,
Erode Arts & Science College (Autonomous), Erode, Tamil Nadu, India.
Email: csincseasc@gmail.com
R.Navin Kumar
M.Phil Scholar (Part Time), Department of Computer Science,
Erode Arts & Science College (Autonomous), Erode, Tamil Nadu, India.
Email: navinsoccer07@gmail.com

Abstract— Measuring the semantic similarity between words is an important component in various tasks on the web such as relation extraction, community mining, document clustering, and automatic metadata extraction. Accurately measuring the semantic similarity between words is an important problem in web mining, information retrieval, and natural language processing. In information retrieval, one of the main problems is to retrieve a set of documents that is semantically related to a given user query. Text processing plays an important role in information retrieval, data mining, and web search. In text processing, the bag-of-words model is commonly used. In this paper a new scheme proposes an empirical method to estimate semantic similarity using page counts and text snippets retrieved from a web database for two words. Specifically, it defines various word co-occurrence measures using page counts and integrates those with lexical patterns extracted from text snippets.

Keywords—Web mining, Text processing, Sematic Similiarity, Pattern matching, Skip Lexical Pattern.

1. INTRODUCTION

Text mining, also referred to as text data mining, roughly correspondent to text analytics, refers to the process of deriving high-quality information from text [1][2] . High-quality information is typically derived through the devising of patterns and trends through means such as statistical pattern learning system. Text mining usually involves the process of structuring the input, deriving patterns within the structured data, and finally evaluation and interpretation of the output. 'High quality' in text mining usually refers to some combination of relevance, novelty, and interestingness [3] [4]. Typical text mining tasks include text categorization, text clustering, and entity extraction, production of granular taxonomies, sentiment analysis, document summarization, and entity relation modeling.

Text analysis involves information retrieval, lexical analysis to study word frequency distributions, pattern recognition, tagging/annotation, information extraction, data mining techniques including link and association analysis, visualization, and predictive analytics. The overarching goal is, essentially, to turn text into data for analysis, via application of natural language processing (NLP) and analytical methods [5] [6] [7].

Information retrieval or identification of a corpus is a preparatory step: collecting or identifying a set of textual materials, on the Web or held in a file system, database, or content management system, for analysis although some text analytics systems apply exclusively advanced statistical methods, many others apply more extensive natural language processing, such as part of speech tagging, syntactic parsing, and other types of linguistic analysis [8] [9]. The following results main contribution for the proposed system

The new system integrates different web-based similarity measures using a machine learning approach.

The new system extracts synonymous word pairs from WordNet and synsets

Positive training instances and automatically generates negative training instances.

Data can be retrieved easily and Retrieval of data based on the similarity and the Page ranking

Highlight the word and Reduces the manual sensation

The remainder of this paper is organized as follows. Section II reviews the similarity measures and related works in web mining. Section III we briefly discuss similarity mechanism and then the proposed lexical pattern technologies in are presented in section IV. Section V performance analysis and section VI concludes this paper.

2. RELATED WORKS

In information retrieval, one of the main problems is to retrieve a set of documents that is semantically related to a given user query. Efficient estimation of semantic similarity between words is critical for various natural language processing tasks such as word sense disambiguation (WSD) and automatic text summarization [10] [11] [12].

Semantically related words of a particular word are listed in manually created general-purpose lexical ontologies such as WordNet. In WordNet, a synset contains a set of synonymous words for a particular sense of a word. However, semantic similarity between entities changes over time and across domains. For example, apple is frequently associated with computers on the web. However, this sense of apple is not listed in most general-purpose thesauri or dictionaries [13] [14] [15].

A user, who searches for apple on the web, might be interested in this sense of apple and not apple as a fruit. New

words are constantly being created as well as new senses are assigned to existing words.

Some measures which have been popularly adopted for computing the similarity between two documents are presented in existing system. Let d1 and d2 be two documents represented as vectors. The Euclidean distance measure is defined as the root of square differences between the respective coordinates of D1 and D2, i.e.,

DEUC [D1, D2] = [(D1-D2). (D1-D2)]1/2

where A·B denotes the inner product of the two vectors A and B. Cosine similarity [25] measures the cosine of the angle between d1 and d2 as follows:

SCos [D1, D2] = D1.D2/ (D1.D1)½ (D2.D2)1/2

The Jaccard coefficient for data processing:

SEJ [D1, D2] = D1.D2/ D1.D1 + D2.D2 –D1.D2

While the Dice coefficient looks similar to it and is defined as follows:

SDic [D1, D2] = 2D1.D2 / D1.D1 +D2.D2

IT-Sim, an information-theoretic measure for document similarity:

$$S_{IT}(d_1, d_2) = \frac{2\sum_{w_i} \min(p_{1i}, p_{2i}) \log \pi(w_i)}{\sum_{w_i} p_{1i} \log \pi(w_i) + \sum_{w_i} p_{2i} \log \pi(w_i)},$$

Where wi represents feature i, pji indicates the normalized value of wi in document dj for j = 1or j = 2, and π(wi) is the proportion of documents in which wi occurs.

A novel similarity measure between two documents and similarity degree increases when the number of presence-absence features pairs decreases. Two documents are least similar to each other if none of the features have non-zero values in both documents. Besides, it is desirable to consider the value distribution of a feature for its contribution to the similarity between two documents. The proposed scheme has also been extended to measure the similarity between two sets of documents. To improve the efficiency, we have provided an approximation to reduce the complexity involved in the computation [16] [17] [18].

3. EXISTING METHODOLOGY

It is important to emphasize that getting from a collection of documents to a clustering of the collection, is not merely a single operation, but is more a process in multiple stages. These stages include more traditional information retrieval operations such as crawling, indexing, weighting, filtering etc. Some of these other processes are central to the quality and performance of most clustering algorithms, and it is thus necessary to consider these stages together with a given clustering algorithm to harness its true potential [19] [20]. We will give a brief overview of the similarity with clustering process, before begin our literature study and analysis.

A. Clustering

A general definition of clustering stated by Brian Everitt et al. [19] Given a number of objects or individuals, each of which is described by a set of numerical measures, devise a classification scheme for grouping the objects into a number of classes such that objects within classes are similar in some respect and unlike those from other classes. The number of classes and the characteristics of each class are to be determined.

B. Feature Base Clustering

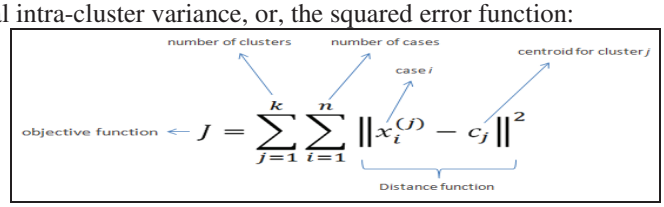
Two types of clustering have been studied - clustering the documents on the basis of the distributions of words that co-occur in the documents, and clustering the words using the distributions of the documents in which they occur. In this algorithm I have used a double-clustering approach in which I first cluster the words and then use this word cluster to cluster the documents [20] [21]. The clustering of words reduces the feature space and thus reduces the noise and increases the time efficiency.

In general, this algorithm can be used for clustering of objects based on their features. A recently introduced principle, termed the information bottleneck method is based on the following simple idea. Given the empirical joint distribution of two variables, one variable is compressed so that the mutual information about the other is preserved as much as possible. Here the two variables are the object and the features. First, the features are clustered to preserve the information of objects and then these clusters are used to reduce the noise in the object graph [22] [23].

The main advantage of this procedure lies in a significant reduction of the inevitable noise of the original co-occurrence matrix, due to its very high dimension. The reduced matrix, based on the word-clusters, is denser and more robust, providing a better reflection of the inherent structure of the document corpus.

C. K-Mean Clustering

K-Means clustering intends to partition n objects into k clusters in which each object belongs to the cluster with the nearest mean [25] [26]. This method produces exactly k different clusters of greatest possible distinction. The best number of clusters k leading to the greatest separation (distance) is not known as a priori and must be computed from the data. The objective of K-Means clustering is to minimize total intra-cluster variance, or, the squared error function:



Clusters the data into k groups where k is predefined.

Select k points at random as cluster centers.

Assign objects to their closest cluster center according to the Euclidean distance function.

Calculate the centroid or mean of all objects in each cluster.

Repeat steps 2, 3 and 4 until the same points are assigned to each cluster in consecutive rounds.

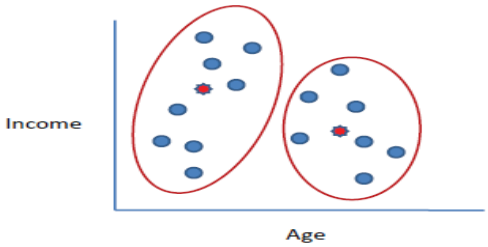


Fig 3.1 K-Mean Clustering

K-Means is relatively an efficient method. However, we need to specify the number of clusters, in advance and the final results are sensitive to initialization and often terminates at a local optimum. Unfortunately there is no global theoretical method to find the optimal number of clusters. A practical approach is to compare the outcomes of multiple runs with different k and choose the best one based on a predefined criterion. In general, a large k probably decreases the error but increases the risk of over fitting [24] [25].

D. CLUSTERING PROCESS

We have divided the offline clustering process into the five stages outlined below Fig 3.2:

Collection of Data includes the processes like crawling, indexing, filtering etc. which are used to collect the documents that needs to be clustered, index them to store and retrieve in a better way, and filter them to remove the extra data , for example, stopwords

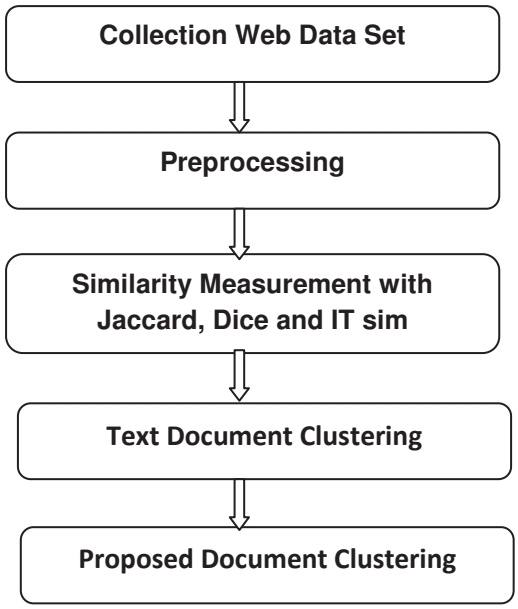


Fig 3.2 Clustering Process

Preprocessing is done to represent the data in a form that can be used for clustering. There are many ways of representing the documents like, Vector-Model, graphical model, etc. Many measures are also used for weighing the documents and their similarities.

Similarity measure is implemented in this paper, called SMTP (Similarity Measure for Text Processing), for two documents with using Jaccard, Dice and IT sim

Document clustering proposed algorithm is a generalization of K-Means algorithm in which the set of K centroids as the model that generate the data. It alternates between an expectation step, corresponding to reassignment, and a maximization step, corresponding to re computation of the parameters of the model.

Proposed algorithm the patterns can be clustered using the lexical pattern clustering algorithm. The patterns are clustered and then the count and co-occurrence of the word can be considered. Based on this the word can be extracted. The cluster can be grouped based on the threshold value.

The proposed scheme along with all existing system approach has also been extended to measure the similarity between 'n' sets of documents. To improve the efficiency, it has provided an approximation to reduce the complexity involved in the computation. Sequential Clustering is also provided to group the documents into Clusters Set.

Stemming is applied before text documents are taken.

Stop word removal is applied which reduces the content size.

Synonym word replacement is applied so that related documents with varying contents also coincide.

Two web documents may have a certain value of cosine similarity, but if neither of them is in the other one's neighborhood, they have no connection between them. In such a case, the proposed system applied some context-based knowledge or relativeness property by modifying the text content.

'N' group of documents set can be prepared as Cluster result.

4. PROPOSED METHODOLOGY

Clustering is being studied since a long time, and many state-of-art algorithms have been applied till date but still the results are not very satisfactory and we are looking for some better algorithms. This gave us the motivation to think out of box and try something simple but different. While calculating the similarity between the documents we first tried to use synonyms of the words also as same words but in case of documents like news articles, its not the synonymy but the co-occurrence of words which plays importance in the similarity [26] [27].

A. Feature Extraction

This is used for extraction of features (important words and phrases in this case) from the documents. We have used Named-Entity and frequency of unigrams and bigrams to extract the important words from the document [28].

B. Feature Clustering

This is the most important phase in which the extracted features are clustered based on their co-occurrence. For this we tried many algorithms and found Squeal clustering algorithms to be best for large data set as it reduces the time taken to a large extent.

C. Document Clustering

This is the final phase in which documents are clustered using the feature clusters. For this we have used a simple approach in which a document is assigned to the cluster of words of which it has the maximum words.

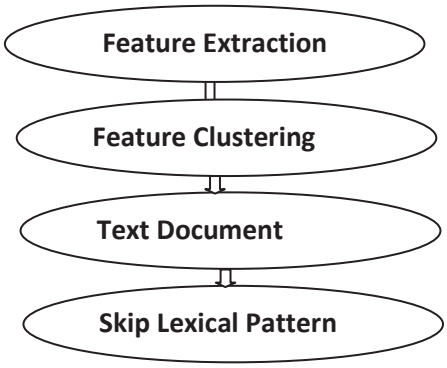


Fig 4.1 Proposed Similarity Clustering

D. Skip Lexical Pattern Clustering

The document retrieval problem in Information Retrieval [31] [32] is as follows: given a query | typically represented as a set of query terms | return a ranked list of documents from some set, ordered by relevance to the query. Terms here can be words or lemmas, or multi-word units, depending on the lexical pre-processing being used. The complete set of documents depends on the application; in the internet-search case it could be the whole web.

One of the features of most solutions to the document retrieval problem, and indeed Information Retrieval problems in general, is the lack of sophistication of the linguistic modeling employed: both the query and the documents are considered to be \bags of words", i.e. multi-sets in which the frequency of words is accounted for, but the order of words is not. From a linguistic perspective, this is a crude assumption (to say the least), since much of the meaning of a document is mediated by the order of the words, and the syntactic structures of the sentences.

However, this simplifying assumption has worked surprisingly well, and attempts to exploit linguistic structure beyond the word level have not usually improved performance. For the document retrieval problem perhaps this is not too surprising, since queries, particularly on the web, tend to be short (a few words) and so describing the problem as one of simple word matching between query and document is arguably appropriate.

Once the task of document retrieval is described as one of word overlap between query and document, then a vector space model is a natural approach[33] [34]: the basis vectors of the space are words, and both queries and documents are vectors in that space.

The coefficient of a document vector for a particular basis vector, in the simplest case, is just the number of times that the word corresponding to the basis appears in the document. Queries are represented in the same way, essentially treating a query as a "pseudo-document". Measuring word overlap, or the similarity of a document vector $d \rightarrow$ and query vector $q \rightarrow$, can be achieved using the dot product:

$$Sim(d\rightarrow, q\rightarrow) = d\rightarrow . q\rightarrow \quad \text{---------- 1}$$
$$= \sum_i d_i \times q_i$$

Where vi is the ith coefficient of vector $v\rightarrow$

The term-document matrix introduced in the previous section gives us the basic structure for determining word similarity. There the intuition was that words or terms are similar if they tend to occur in the same documents. However, this is a very broad notion of word similarity, producing what we might call topical similarity, based on a coarse notion of context. The trick in arriving at a more refined notion of similarity is to think of the term-document matrix as a term-context matrix, where, in the IR case, context was thought of as a whole document. But we can narrow the context down to a sentence, or perhaps even a few words either side of the target word[35].

Term Definition

The first order of business to define few important terms: term, term weight, recall, precision, and tolerance. Term, to begin with, is basically the keyword or important concept associated with a given document. The importance of a term in representing the semantic of the document is the term weight. One way to calculate it is to count the frequency with which the term occurs in document. As for the recall and precision, they are measures of performance. Higher the precision and recall, better the Information Retrieval system is.

0 < Recall = #Relevant Document Retrieval/ #Relevant Document Collection

Finally, there is the tolerance. It is the benchmark for which only the documents with relevance score higher than it are retrieved.

0< Precision = # Relevant Document Retrieval/Document Retrieval

Term-Document Matrix

After getting familiar with important term definition, the second order of business is to learn how the Vector Space Model is constructed. Basically, the documents are stored in a term-document matrix. A total of d documents with t terms, for instance, are stored in a t x d matrix. Each vector of the matrix represents each individual document. Each element on a column represents frequency a term occurs in a document. Thus, aij will represent the frequency that term i is presented in the document j.

Lexical Pattern extraction Algorithm
Input: Word Pair W
Output: Extraction Patterns A

Given a set W of word-pairs, Skip extract patterns
For each word-pair (P, Q) ε W
do A ← Get-Snippets ("P Q")
N ← null
For each snippet a ε A
do N ← N + Get-N-grams (a, P, Q)
Pats ← Count-Freq (N)
return (Pats)

Lexical Skip pattern Clustering
Input: patterns A (a1, … ,an), threshold θ
Output: Clusters C

SORT (A)
C ← {}
for pattern ai ∈ A do
max ← - ∞
c* ← null
for cluster cj ∈ C do
sim ← cosine (ai, cj)
if sim > max then
max ← sim
c ← cj
end if
end for
if max > θ then
c* ← c* ⊕ ai
else
C ← C ∪ {ai}
end if
end for
return C

The patterns can be clustered using the Skip Lexical Pattern Clustering Algorithm (SLPCA). The patterns are clustered and then the count and co-occurrence of the word can be considered. Based on this the word can be extracted. The cluster can be grouped based on the threshold value, the words are clustered and then the results are produced.

5. PERFORMANCES ANALYSIS

The following Table 5.1 shows Spearman's Rank Correlation Coefficient experimental result for existing system. The table contains word pair, word pair one value [W1], word pair one rank value [R1], word pair two value [W2], word pair two rank value

Table 5.1 Word Pair Relation

S.No	Word-pair	W1	R1	W2	R2	Word Pair [n]
1	cord-smile	38	2	14	12	13
2	monk-oracle	15	13	10	13	8
3	noon-string	22	8	16	11	13
4	glass-magician	25	7	17	10	13
5	monk-slave	15	13	18	9	8
6	coast-forest	27	5	17	10	15
7	crane-implement	18	11	21	7	11
8	car-automobile	30	3	18	9	17
9	brother-lad	19	10	38	1	19
10	bird-crane	29	4	18	9	17
11	bird-cock	29	4	25	4	23
12	coast-hill	27	5	30	2	22
13	car-journey	44	1	23	5	23
14	implement-tool	21	9	26	3	16
15	boy-lad	26	6	38	1	24
16	forest-graveyard	17	12	9	14	8
17	midday-noon	15	13	22	6	15
18	furnace-stove	17	12	20	8	17
19	magician-wizard	17	12	21	7	17
20	lad-wizard	38	2	21	7	21

Spearman's rank correlation coefficient (rs) is a reliable and fairly simple method of testing both the strength and direction (positive or negative) of any correlation between two word pair or document.

$$r_s = 1 - [N \Sigma d^2 / n^3 - n]$$

Where d2 = [R2-R1]2, n= Word Pair Count, N= Total number of word pair.

The Table 5.2 shows the spearman's rank correlation coefficient between two word pair for existing system. The table contains difference between rank values, square of rank values, word pair count and cube value of word pair values and spearman's rank correlation coefficient for each word pair (rs) details are shown. The overall word pair spearman's Rank Correlation Coefficient value is 0.8239.

The Table 5.3 shows experimental result for existing system analysis. The table contains word pair, word jaccard value, word overlab values, word dice values, word PMI values and its PMI max values details are shown. The word pair count details are measure the cor-relation coefficient score value in each word pair using precision and recall measure. The over all word pair coefficient values are jaccard value is 0.584, overlab value is 0.875, dice values is 0.695, PMI values are 2.846 and PMI max values are 4.025.

Table 5.2 Spearman's Rank Correlation Coefficient

R2-R1	d2	n	n3	rs
10	100	13	2197	0.0842
0	0	8	512	1.0432
3	9	13	2197	0.9175
3	9	13	2197	0.9175
-4	16	8	512	0.3650
5	25	15	3375	0.8511
-4	16	11	1331	0.7575
6	36	17	4913	0.8529
-9	81	19	6859	0.7631
5	25	17	4913	0.8978
0	0	23	12167	1.0567
-3	9	22	10648	0.9830
4	16	23	12167	0.9736
-6	36	16	4096	0.8235
-5	25	24	13824	0.9637
2	4	8	512	0.8412
-7	49	15	3375	0.7083
-4	16	17	4913	0.9346
-5	25	17	4913	0.8978
5	25	21	9261	0.9458
Spearman's Rank Correlation Coefficient				**0.8239**

The Fig 5.1 shows experimental result for existing system analysis. The figure contains word pair, word jaccard value, word overlab values, word dice values, word PMI values and its PMI max values details are shown. The word pair count details are measure the correlation coefficient score value in each word pair using precision and recall measure. The over all word pair coefficient values are accard value is 0.584, overlab value is 0.875, dice values is 0.695, PMI values are 2.846 and PMI max values are 4.025

The Table 5.4 shows examples are described the proposed system methodology in relation independent and relation specific word pair relation measurement process.

The Table 5.5 shows and takes word pair details in proposed system analysis. The proposed system measures the relation independent and relation specific co efficient values.

Table 5.3 Comparisons of Jaccard, Overlab, Dice, PMI and PMI max and Previous Measures on the Mc Data Set

S. NO	WORD-PAIR	JACCARD	OVERLAB	DICE	PMI	PMI Max
1	cord-smile	0.33	0.93	0.50	2.69	3.93
2	monk-oracle	0.47	0.80	0.64	3.47	4.27
3	noon-string	0.52	0.81	0.68	3.10	4.23
4	glass-magician	0.45	0.76	0.62	2.91	4.11
5	monk-slave	0.32	0.53	0.48	2.88	3.91
6	coast-forest	0.52	0.88	0.68	2.88	4.20
7	crane-implement	0.39	0.61	0.56	2.86	4.02
8	car-automobile	0.55	0.94	0.71	2.94	4.22
9	brother-lad	0.50	1.00	0.67	2.76	4.13
10	bird-crane	0.57	0.94	0.72	2.98	4.24
11	bird-cock	0.74	0.92	0.85	2.95	4.34
12	coast-hill	0.63	0.81	0.77	2.80	4.21
13	car-journey	0.52	1.00	0.69	2.62	4.09
14	implement-tool	0.52	0.76	0.68	2.87	4.15
15	boy-lad	0.60	0.92	0.75	2.68	4.16
16	forest-graveyard	0.44	0.89	0.62	3.45	4.25
17	midday-noon	0.68	1.00	0.81	3.31	4.42
18	furnace-stove	0.85	1.00	0.92	3.41	4.54
19	magician-wizard	0.81	1.00	0.89	3.36	4.50
20	lad-wizard	0.55	1.00	0.71	2.76	4.17
Standard Deviation		**0.584**	**0.875**	**0.695**	**2.846**	**4.025**

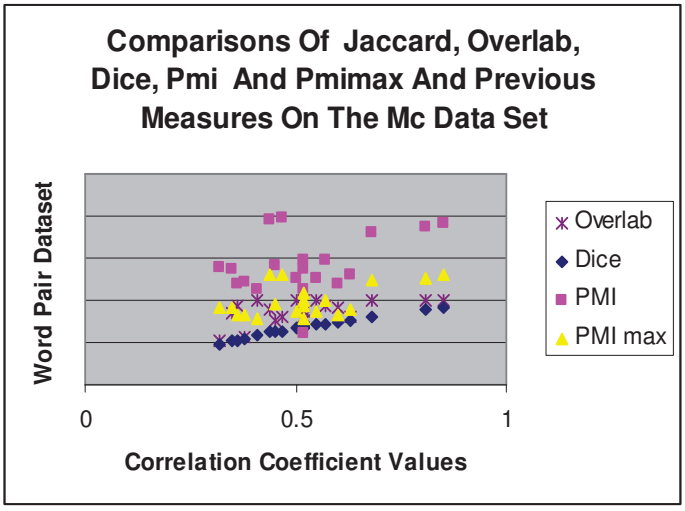


Fig 5.1 Comparisons of Jaccard, Overlab, Dice, PMI and PMI max and Previous Measures on the Mc Data Set

Table 5.4 Word Pair Example

S.NO	WORD PAIR
1	Pain
2	Cord
3	Smile
4	String
5	Noon
6	Blue
7	Car
8	Automobile
9	Forest
10	Coast

Table 5.5 Word Pair Co-efficient Values

S.No	Word-pair	Jaccard	Overlap	Dice	PMI	PMI max
1	cord-smile	0.33	0.93	0.50	2.69	3.93
2	noon-string	0.52	0.81	0.68	3.10	4.23
3	Journey-voyage	0.67	0.94	0.80	3.21	4.38
4	lad-wizard	0.55	1.00	0.71	2.76	4.17
5	coast-forest	0.52	0.88	0.68	2.88	4.20
6	midday-noon	0.68	1.00	0.81	3.31	4.42
7	coast-hill	0.63	0.81	0.77	2.80	4.21
8	monk-oracle	0.47	0.80	0.64	3.47	4.27
9	magician-wizard	0.81	1.00	0.89	3.36	4.50
10	car-automobile	0.55	0.94	0.71	2.94	4.22
		0.573	0.911	0.719	3.052	4.253

The Table 5.6 takes word pair details in core-smile and Noon-string. The proposed system measures the relation independent and relation specific co efficient values. The Joint probability Values are given below.

Table 5.6 Joint probability values- Word Pair [Relation dependent and Independent Values]

S.NO	Word pair relation	Joint probability values
1	[pain, cord, smile]	0.32
2	[pain, cord, String]	0.32
3	[pain, Noon, smile]	0.36
4	[pain, Noon, String]	0.36
5	[Blue, cord, smile]	0.18
6	[Blue, cord, String]	0.18
7	[Blue, Noon, smile]	0.07
8	[Blue, Noon, String]	0.15
9	[car, automobile ,car]	0.16
10	[forest, coast , forest]	0.77

Table 5.7 shows the table joint probability values in word pair for relation dependent and independent co-efficient values. Joint probability values are measure using the following equation.

$$p(\rho, R) = \frac{\sum_{(A,B)\in R} f(\rho, A, B)}{\sum_{\rho\in\Phi} \sum_{R\in\Omega} \sum_{(A,B)\in R} f(\rho, A, B)}.$$

Total frequency of a pattern ρ in a particular relation type R is defined as the sum of the frequencies of ρ in all entity pairs Table 5.8 shows the table mutual information values in word pair for relation dependent and independent co-efficient values. Mutual information values are measure using the following equation

$$I(\rho, R) = p(\rho, R) \log_2 \left(\frac{p(\rho, R)}{p(\rho)p(R)} \right).$$

Table 5.7 & 5.8 Mutual Information values-Word Pair [Relation dependent and Independent Values]

S.NO	Word pair relation	Mutual Information values
1	[pain, pain]	0.654
2	[pain, Blue]	0.663
3	[Blue, pain]	0.351
4	[Blue, Blue]	0.320
5	[cord, smile]	0.234
6	[Noon, String]	0.765
7	[car, Automobile]	0.457
8	[forest, coast]	0.556

Table 5.9 shows the table entropy values in word pair for relation dependent and independent co-efficient values. Entropy values measurement using the following equation.

$$H(\rho) = \sum_{R\in\Omega} \sum_{(A,B)\in R} p(\rho, A, B) \log_2 p(\rho, A, B).$$

Table 5.9 Entropy values- Word Pair [Relation dependent and Independent Values

S.NO	Word pair relation	Entropy values
1	Pain	14873.942
2	Cord	68385.919
3	Smile	87654.332
4	String	65435.221
5	Noon	12666.223
6	Blue	76543.222
7	Car	34562.115
8	Automobile	23457.223
9	Forest	34593.104
10	Coast	34925.112

The Fig 5.2 takes word pair details in proposed word pair. The proposed system measures the relation independent and relation specific co efficient values.

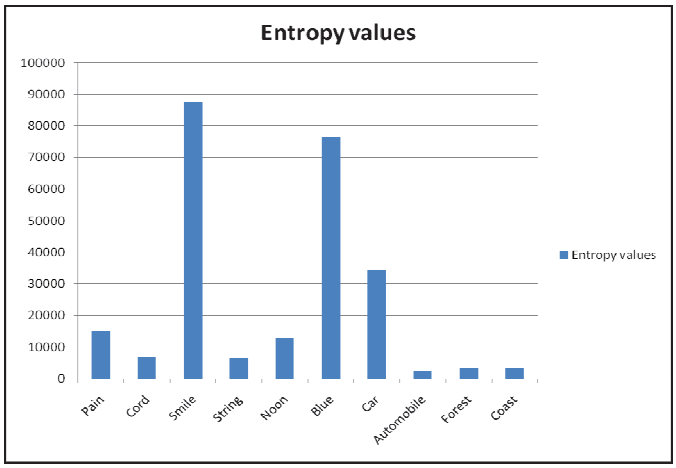


Fig 5.2 Lexical Skip Pattern Cluster Algorithm [Joint Probability Value]

Fig 5.3 shows the table mutual information values in word pair for relation dependent and independent co-efficient values

Table 5.10 is describing the comparison between spearman's Rank and existing system. The table contains spearman's Rank in all word pair details and word pair standard deviation details are shown.

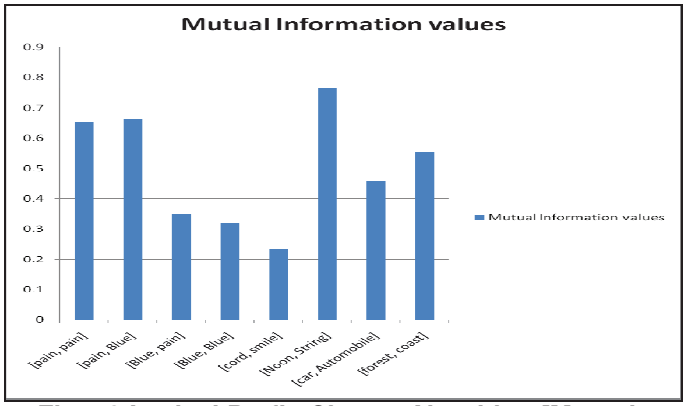


Fig 5.3 Lexical Prefix Cluster Algorithm [Mutual Information Value

Table 5.11 is shows the comparison between spearman's Rank and prefix span clustering algorithm. The table contains spearman's Rank in word pair and proposed word pair relation details are shown.

Table 5.10 Comparison of Spearman's Rank and Existing system

SPEARMAN'S RANK	EXISTING SYSTEM COEFFICIENT Standard Deviation [Word Pair]
0.8235 [All Word Pair]	0.584
	0.875
	0.695
	PMI: 2.846
	PMI Max: 4.025

Table 5.11 Comparison of Spearman's Rank and Lexical Pattern Clustering Algorithm

SPEARMAN'S RANK	PROPOSED LEXICAL PATTERN CLUSTERING ALGORITHM [Word Pair Relation]
0.8235 [All Word Pair]	0.654
	0.663
	0.351
	0.320

Table 5.12 is describing the comparison between Existing Coefficient and prefix span clustering algorithm. The table contains Existing Word pair Coefficient in word pair and proposed word pair relation details are shown.

Table 5.12 Comparison of Existing Word pair Coefficient and Prefix Span Clustering Algorithm

Word pair relation	EXISTING WORD PAIR COEFFICIENT	PROPOSED LEXICAL PATTERN CLUSTERING ALGORITHM [Word Pair Relation]
Pain	0.8235	0.654
Cord	0.8153	0.663
Smile	0.8003	0.351
String	0.8213	0.320
Noon	0.8441	0.322
Blue	0.8521	0.567
Car	0.8457	0.775
Automobile	0.8235	0.234
Forest	0.8235	0.789
Coast	0.8235	0.586

Fig 5.4 is shows the comparison between Existing Coefficient and Prefix Span Clustering Algorithm. The figure contains Existing Word pair Coefficient in word pair and proposed word pair relation details are shown. [Word pair Cord-Smile] [Pain and Blue]

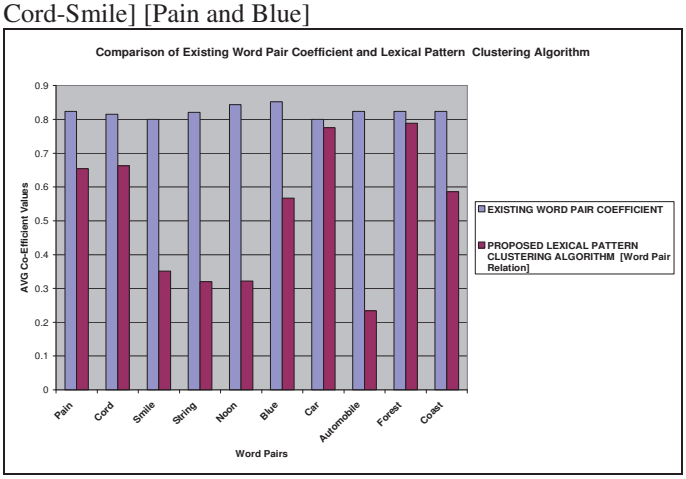


Fig 5.4 Comparison of Existing Word Pair Coefficient and Lexical Pattern Clustering Algorithm

The Table 5.13 is shows the bug word count term matrix in existing system process. The table contains bugs word and source code entity file is to finding co occurrence bugs word count details are shown below.

Table 5.13 Term Matrix

BUGS REPORT	SOURCE CODE ENTITY			
	1.html	1_2.html	1_3.html	1_4.html
ANTICMOS	3	2	5	4
ELIZA	2	3	5	4
GRAYBRID	3	3	5	4
COMMWAR RRIOR	3	0	5	4
ACMS	3	3	5	4
ABRAXAS	3	3	5	0
ACTIFED	3	3	5	4
BOMBER	2	3	1	4
AGENA	1	0	0	1
BUG	2	0	0	1

The Table 5.14 is shows the cosine similarity between bugs report entity and source code entity in existing system process. The table contains bugs report file and source code file in threshold value < 0.8 for vectors base model details are shown below.

The Table 5.15 is shows the cosine similarity between bugs report entity and source code entity in existing system process. The table contains bugs report file and source code file in threshold value < 0.5 for vectors base model details are shown below.

Table 5.14 cosine similarity value < 0.8

Entity	1. html	1_2. html	1_3. html	1_4. html	1_4_2. html
1.html	-	0.86	0.86	0.85	-
1_2 html	-	0.87	0.87	0.89	-
1_3.html	-	0.95	0.95	0.89	-
1_4.htnl	-	0.89	0.89	0.84	-

Table 5.15 cosine similarity value <0.5

Entity	1. html	1_2. html	1_3. html	1_4. html	1_4_2. html
1.html	0.76	0.86	0.86	0.85	0.72
1_2 html	0.75	0.87	0.87	0.89	0.72
1_3.html	0.73	0.95	0.95	0.89	0.73
1_4.htnl	0.71	0.89	0.89	0.84	0.65

The Fig 5.5 is shows the cosine similarity between bugs report entity and source code entity in existing system process. The fig contains bugs report file and source code file in threshold value < 0.8 for vectors base model details are shown below.

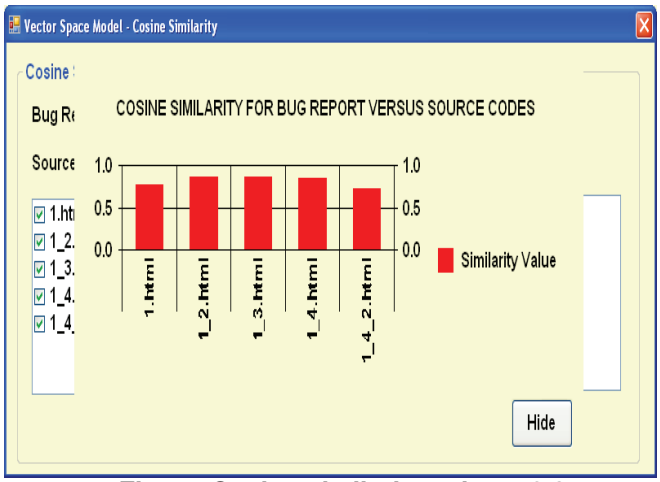


Fig 5.5 Cosine similarity value < 0.8

The Fig 5.6 is shows the cosine similarity between bugs report entity and source code entity in existing system process. The figure contains bugs report file and source code file in threshold value < 0.5 for vectors base model details are shown below.

The Table 5.16 is shows the bug word skip count in lexical pattern clustering process in proposed system. The table contains source code entity, word skip count value and group of phrase bugs word count details are shown below.

For Example: Bugs Word: ANTICMOS ****** ABRAXAS

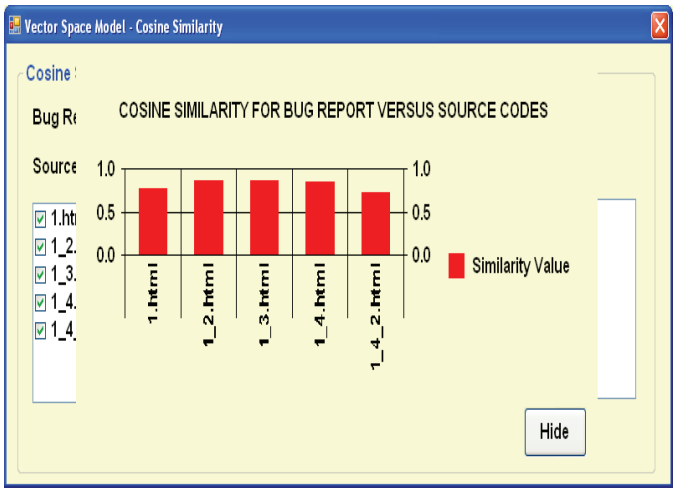


Fig 5.6 Cosine similarity value <0.5

Table 5.16 Skip Count in Lexical Pattern Clustering

S.No	Source Entity File	Bugs Skip Count	Bugs Phrase Count
1	10.html	1	24
2	10.html	2	24
3	2.html	3	2
4	2.html	5	3
5	2.html	7	3

The Table 5.17 is shows the bug word cluster size in lexical pattern clustering algorithm in proposed system. The table contains threshold values and cluster size details are shown below

Table 5.17 Cluster size in Lexical Pattern Clustering

S.NO	Threshold values	Cluster Size
1	0.5	1
2	0.6	2
3	0.8	2
4	0.9	4
5	1	5

The Fig 5.7 is shows the bug word skip count in lexical pattern clustering process in proposed system. The figure contains source code entity, word skip count value and group of phrase bugs word count details are shown below.

The Fig 5.8 is shows the bug word cluster size in lexical pattern clustering algorithm in proposed system. The figure contains threshold values and cluster size details are shown below.

HTML tags are removed from bug report and source code entity file before taken for similarity identification. Syntax words are removed from source code entity file before taken for similarity identification.

The comparison table is used to represent the status of analyzing patterns and pattern clustering in both system analyses such as existing system and proposed system with processing details of stop word, stem word, synonym word. In the existing system, the pattern is not used in the application and pattern clustering were not applied. The proposed system is processed as patterns are used and pattern clustering are applied in the paper.

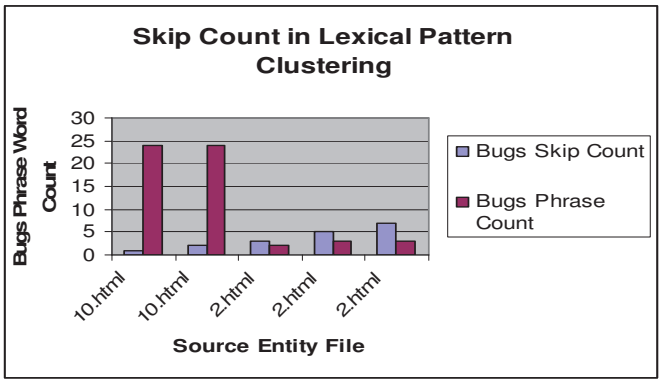


Fig 5.7 Skip Count in Lexical Pattern Clustering

The results are discussed under the performance of novel cluster base lexical pattern scheme. The result of proposed model bugs finding is discussed and compared with the existing finding bugs. To measure the performance of the proposed works are throughput threshold values are evaluated.

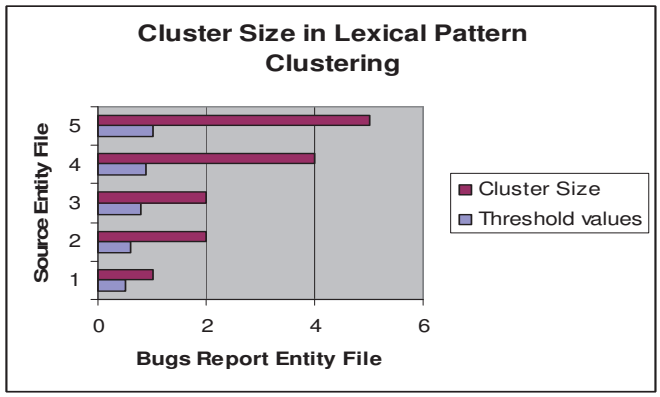


Fig 5.8 Cluster size in Lexical Pattern Clustering

6. CONCLUSION

In this paper presents a novel similarity measure between two documents. Several desirable properties are embedded in this measure. For example, the similarity measure is symmetric. The presence or absence of a feature is considered more essential than the difference between the values associated with a present feature. The similarity degree increases when the number of presence-absence features pair decreases.

Two documents are least similar to each other if none of the features have non-zero values in both documents. Besides, it is desirable to consider the value distribution of a feature for its contribution to the similarity between two documents. The proposed scheme has also been extended to measure the similarity between two sets of documents.

In addition, the paper proposed a semantic similarity measure using both page counts and snippets retrieved from a web search engine for two words. Four word co-occurrence measures were computed using page counts. It proposed a skip lexical pattern extraction algorithm to extract numerous semantic relations that exist between two words. Moreover, a sequential pattern clustering algorithm was proposed to identify different skip lexical patterns that describe the same semantic relation.

The paper proposes an empirical method to estimate semantic similarity using page counts and text snippets retrieved from a web search engine for two words. Specifically, it defines various word co-occurrence measures using page counts and integrates those with skip lexical patterns extracted from text snippets. Both page counts-based co-occurrence measures and lexical pattern clusters were used to define features for a word pair.

7. FUTURE ENHANCEMENTS

The new system will implement the following future enhancements, if it is implemented the application will further improved so that the paper can work with the full efficiency.

- To ensure reliability in ever growing client count.
- The content can be searched from more than one search engine.
- In future, searching and comparing the video content based on the exact semantic words.
- Multitasking can be performed.
- If developed as web service, it can be accessed from anywhere
- The proposed system can be further developed by working in different operating system independently.

8. REFERENCES

[1] G. Amati and C. J. V. Rijsbergen, "Probabilistic models of information retrieval based on measuring the divergence from randomness," ACM Trans. Inform. Syst., vol. 20, no. 4, pp. 357–389, 2002.

[2] J. A. Aslam and M. Frost, "An information-theoretic measure for document similarity," in Proc. 26th SIGIR, Toronto, ON, Canada, 2003, pp. 449–450.

[3] D. Cai, X. He, and J. Han, "Document clustering using locality preserving indexing," IEEE Trans. Knowl. Data Eng., vol. 17, no. 12, pp. 1624–1637, Dec. 2005.

[4] S. Clinchant and E. Gaussier, "Information-based models for ad hoc IR," in Proc. 33rd SIGIR, Geneva, Switzerland, 2010, pp. 234–241.

[5] I.S.Dhillon, J. Kogan, and C. Nicholas, "Feature selection and document clustering," in A Comprehensive Survey of Text Mining, M. W. Berry, Ed. Heidelberg, Germany: Springer, 2003.

[6] I.S. Dhillon and D.S. Modha.Concept decompositions for large sparse text data using clustering.Machine Learning, 42(1):143–175, January 2001.Also appears as IBM Research Report RJ 10147, July 1999.

[7] I. S. Dhillon and D. S. Modha, "Concept decompositions for large sparse text data using clustering," Mach. Learn., vol. 42, no. 1, pp. 143–175, 2001.

[8] Broder, A. Z., S. C. Glassman, M. S. Manasse, and G. Zweig: 1997, `Syntactic clustering of the web'. Technical Report 1997-015, Digital Systems Research Center.

[9] Dhillon, I. S., D. S. Modha, and W. S. Spangler: 1998, `Visualizing Class Structure of Multidimensional Data'. In: S.Weisberg (ed.): Proceedings of the 30th Symposium on the Interface: Computing Science and Statistics, Vol. 30. Minneapolis, MN, pp. 488-493.

[10] H. Fang, T. Tao, and C. Zhai, "A formal study of heuristic retrieval constraints," in Proc. 27th SIGIR, Sheffield, South Yorkshire, U.K., 2004, pp. 49–56.

[11] G. Salton. Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer. Addison-Wesley, 1989.

[12] J. Lafferty and C. Zhai. Probabilistic relevance models based on document and query generation. In W. B. Croft and J. Laberty, editors, Language Modeling and Information Retrieval. Kluwer Academic Publishers, 2003.

[13] N. Fuhr. Language models and uncertain inference in information retrieval. In Proceedings of the Language Modeling and IR workshop.

[14] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In Proceedings of SIGIR'01, pages 334–342, Sept 2001.

[15] C. Bird, A. Bachmann, E. Aune, J. Duffy, A. Bernstein, V. Filkov, and P. Devanbu, "Fair and Balanced?: Bias in Bug-Fix Data Sets," Proc. Seventh European Software Eng. Conf. and Symp. Foundations of Software Eng., pp. 121-130, 2009.

[16] J. Chang and D.M. Blei, "Relational Topic Models for Document Networks," Proc. 12th Int'l Conf. Artificial Intelligence and Statistics, pp. 81-88, 2009.

[17] G.V. Cormack, C.L. Clarke, and S. Buettcher, "Reciprocal Rank Fusion Outperforms Condorcet and Individual Rank Learning Methods," Proc. 32nd Int'l Conf. Research and Development in Information Retrieval, pp. 758-759, 2009.

[18] R. Moser, W. Pedrycz, and G. Succi, "A Comparative Analysis of the Efficiency of Change Metrics and Static Code Attributes for Defect Prediction," Proc. 30th Int'l Conf. Software Eng., pp. 181-190, 2008.

[19] F. Rahman, D. Posnett, A. Hindle, E. Barr, and P. Devanbu, "BugCache for Inspections: Hit or Miss?" Proc. 19th Symp. and 13th European Conf. Foundations of Software Eng., pp. 322-331, 2011.

[20] S. Rao and A. Kak, "Retrieval from Software Libraries for Bug Localization: A Comparative Study of Generic and Composite Text Models," Proc. Eighth Working Conf. Mining Software Repositories, pp. 43-52, 2011.

[21] Y. S. Maarek, D. M. Berry, and G. E. Kaiser. An Information Retrieval Approach for Automatically Constructing Software Libraries. IEEE Trans. Softw. Eng., 17(8):800–813, 2006.

[22] E. Kocaguneli, T. Menzies, and J.W. Keung, "On the Value of Ensemble Effort Estimation," IEEE Trans. Software Eng., vol. 38, no. 6 pp. 1403-1416, Nov./Dec. 2012.

[23] S.K. Lukins, N.A. Kraft, and L.H. Etzkorn, "Bug Localization Using Latent Dirichlet Allocation," Information and Software Technology, vol. 52, no. 9, pp. 972-990, 2010.

[24] A.T. Nguyen, T.T. Nguyen, J. Al-Kofahi, H.V. Nguyen, and T.N. Nguyen, "A Topic-Based Approach for Narrowing the Search Space of Buggy Files from a Bug Report," Proc. 26th Int'l Conf. Automated Software Eng., pp. 263-272, 2011

[25] S. Rao and A. Kak, "Retrieval from Software Libraries for Bug Localization: A Comparative Study of Generic and Composite Text Models," Proc. Eighth Working Conf. Mining Software Repositories, pp. 43-52, 2011.

[26] R.L. Glass, Facts and Fallacies of Software Engineering. Addison-Wesley Professional, 2003.

[27] R.W. Selby, "Enabling Reuse-Based Software Development of Large-Scale Systems," IEEE Trans. Software Eng., vol. 31, no. 6 pp. 495-510, June 2005.

[28] S.K. Lukins, N.A. Kraft, and L.H. Etzkorn, "Bug Localization Using Latent Dirichlet Allocation," Information and Software Technology, vol. 52, no. 9, pp. 972-990, 2010.

[29] http://web.ist.utl.pt/~acardoso/datasets/[Online].

[30] http://www.cs.technion.ac.il/~ronb/thesis.html

[31] http://www.daviddlewis.com/resources/testcollections /reuters21578/

[32] C. G. González, W. Bonventi, Jr., and A. L. V. Rodrigues, "Density of closed balls in real-valued and autometrized boolean spaces for clustering applications," in Proc. 19th Brazilian Symp. Artif. Intell., Savador, Brazil, 2008, pp. 8–22.

[33] R. W. Hamming, "Error detecting and error orrecting codes," Bell Syst. Tech. J., vol. 29, no. 2, pp. 147–160, 1950.

[34] K. M. Hammouda and M. S. Kamel, "Efficient phrase-based document indexing for web document clustering," IEEE Trans. Knowl. Data Eng., vol. 16, no. 10, pp. 1279–1296, Oct. 2004.

[35] K. M. Hammouda and M. S. Kamel, "Hierarchically distributed peer-to-peer document clustering and cluster summarization," IEEE Trans. Knowl. Data Eng., vol. 21, no. 5, pp. 681–698, May 2009.

[36] J. Han and M. Kamber, Data Mining: Concepts and Techniques, 2nd ed. San Francisco, CA, USA: Morgan Kaufmann; Boston, MA, USA: Elsevier, 2006.