Journal of Engineering and Applied Sciences 13 (6): 1499-1505, 2018
ISSN: 1816-949X

Web Documents Similarity Using K-Shingle Tokens and MinHash Technique

Mehdi Ebady Manaa and Ghufran Abdulameer
College of Information Technology, University of Babylon, 51001 Babylon, Iraq

Abstract: Nowadays, web search engine plays an integral role in discarding similar documents from the web search engine using one of the effective data mining techniques. Document similarity techniques in a massive data mining is such important technique in order to detect the mirror pages and the similarity of the articles in a large web repository. This will lead to avoid showing two web pages which are near identical at the top of search results. One of the document similarity approach is based on K-shingle which is a unique sequence of consecutive K words that can be used to find the similarity between two documents (K is a positive integer). The large web documents can be represented in a sets of long bit vectors 0 and 1. Here, 0 means not found while 1 means found in that document. The two documents that are near identical should have many shingles in common. The similarity ratio is calculated by using one of the distance metrics such as Jaccard similarity between two documents. Jaccard similarity is working well in the comparison between a pair of set values in a small dataset and to find the similarity score. Whereas in the large data set, MinHash and Locality-Sensitive Hashing (LSH) techniques come to solve this problem by providing a small signature matrix for the fast approximation to the truly Jaccard similarity in less time. In this study, we apply the Jaccard similarity, MinHash and LSH techniques based on K-shingles for a different number of the documents. The results show that the MinHash and LSH techniques produce more accuracy in results with less time for large documents. The experimental results show that the chosen K-shingle is applied into different documents number of ranges from 100, 200, 300-1000 documents. The hash functions are applied in different number from 10, 20 and 30. The average similarity time is <5 sec. The false positive and false negative were minimum to truly clustering of the documents.

Key words: K-shingle, Jaccard similarity, MinHash, LSH, documents similarity, data mining

INTRODUCTION

The improvement of information technology has created the extensive measure of databases and colossal information in different territories. The exploration in databases and data innovation has offered to ascend in a way to deal with the store and control this valuable information for further basic leadership. A search engine such as Google or Yahoo have accessed the information on the web using the web crawlers and to provide a search over billions of documents stored on millions of computers (Peshave and Dezhgosha, 2005). Information in the web is retrieved by using one of many TCP/IP suite protocols such as HTTP, HTTPs and FTP. Knowledge retrieval methods have a significant role to retrieve the essential information from the internet. For example, people withdraw money from ATM machine using credit card number (Parziale *et al*., 2006).

The data on the Internet includes data that organized in an unstructured way. The "unstructured data" means not clear semantically or not easy to computer structure. In opposite, the "structured data" consists of structure data that organize in a systematic way such as a relational database which consists of records and attributes. In reality, the most texts on the internet are structured if we count the latent linguistic structure of human languages. The most texts are structure because it includes headings, paragraphs and footnotes which are commonalty represented in documents by explicit the underlying code for these web text (Manning *et al*., 2008).

Data mining is a procedure of extraction of valuable data and examples from immense information. It is called knowledge discovery process, information mining or information extraction (Han and Kamber, 2006; Bhaya and Manaa, 2014). In other words, data mining is a consistent procedure that is utilized to look through an extensive measure of information to discover helpful information. Many algorithms and techniques are used for knowledge discovery from databases. Figure 1 illustrates the main techniques in data mining.

Data mining comes into many steps: from the selection of data to preprocessing step that removes

Corresponding Author: Mehdi Ebady Manaa, College of Information Technology, University of Babylon, 51001 Babylon, Iraq

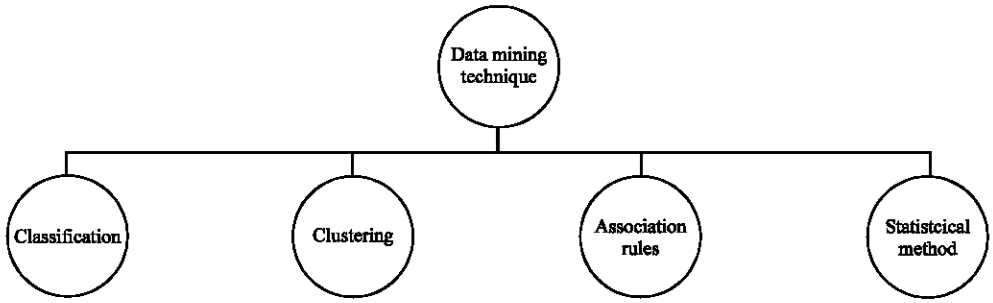

Fig. 1: Data mining techniques

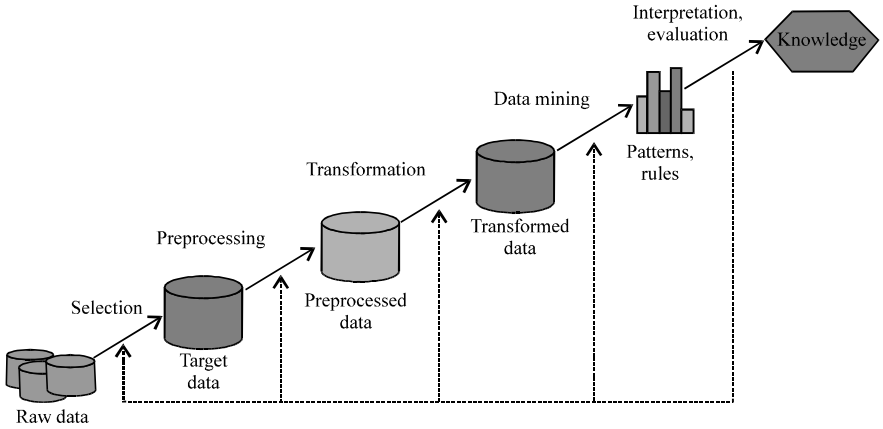

Fig. 2: Knowledge discovery for data

noise data and fills the missing values. Transformation step to put the data in the suitable mode. Model construction or data mining approaches such as clustering or classification and then evaluation step. Figure 2 shows the main step of data mining lifecycle steps (Hamilton *et al*., 2013).

Text clustering is an imperative utilization of data mining. It is interested in gathering similar content for documents data. document clustering has been examined for use in various diverse zones. clustering has been proposed for use in perusing a gathering of archives returned by a web search tool because of a client's question. Web search engines browse big documents collections for further processing. Document similarity is a fundamental task in data mining techniques and can be used in many applications such as clustering, classification and ranking (Wang *et al*., 2015) . Document clustering is implemented by converting words of large documents into a set based on K-shingles and then find similarity between sets using Jaccard similarity such as clustering social media documents, articles news and finding the plagiarism between documents.

Web search engine such as Google avoid the duplicate documents to be appeared in the top result of the search engine using many techniques. K-shingle is one method to convert document words into a set of words. Each shingle has many words depend on the number of K. It represents documents by sets and then facilitates the process of similarity using Jaccard coefficient. The K-shingle with Jaccard similarity is working well for the small documents but it is not effective to find similarity for large documents (Phan *et al*., 2015). MinHash technique is a method to use numbers of hash functions for shingles instead of substring and find the probability of pair documents to be nearly identical using signature matrix. Signature matrix is constructed by choosing random n row permutation from characteristic matrix M. Then, call the MinHash function by these permutation h_1, h_2, …, h_n. The signature matrix S will replace the ith column of M by the MinHash signature for the ith column.

Literature review: This study illustrates the related research for the proposed system. Many researchers adopted different techniques for document similarities. The similarity between object using self-join problem is adopted by Baraglia *et al*. (2010). They showed in the proposed method to discover all those pairs of objects

whose similarity above predefined user threshold. The map-reduce technique is applied effectively to leverage the Distributed File System (DFS) to support the communication patterns and to achieve the scalability. The obtained results show the parallel work outperform the state-of-art serial algorithm by factor 4.5. The clustering of books relationship regarding book itself and page level is adopted by Spasojevic and Poncin (2011). The classification of the dataset that consists of 15 M (5B pages) is run using the hashing technique with the pre-defined threshold. The margin SVM classifier is obtained with two performance measurement recall and precisions to evaluate the system. The feature selection of the pair documents is implemented using Character Error Ratio (CER) and word length. The MapReduce parallel technique is adopted to reduce the time of pairs comparisons.

Two efficient algorithms are applied using Refined Dj-Index based on Jaccard similarity (Deng *et al*., 2012). The real data from Computer Science Bibliography database is applied that contain more than 1.2 million biographic records. These records are extracted based on publication year and venue. The similar objects are grouped in one database and different objects which are non-similar are generated with similarity 0. SampleDJ and TrackDJ are two affecting algorithms for fast approximation and computing the average pairwise Jaccard coefficient. The evaluation metrics RDJ are error are used to evaluate the proposed system.

Multi-view based similarity approach is proposed by Thella and Sridevi (2013). They supposed that the two points Pi and Pj in the cluster clustering C is viewed from the third reference point Pk. The similarity between two documents is calculated using cosine angle between (Pi, Pj) with the regarding of the Euclidean distance. Locality-Sensitive Hashing (LSH) is proposed to detect the social spammer who copies information from other and causes duplicate Tweets (Zhang *et al*., 2013). They use real-life microblog dataset crawled from the Sina Weibo. The results for the potential spammers and their behavior are analyzed. They set the number of characters in shingles to 2-4 and they set 200 MinHash functions to generate the signature matrix. MapReduce paradigm is implemented using an open source of Hadoop to reduce the time complexity for large tweets.

K-shingle with Jaccard similarity is used to build the characteristic matrix. Since, the time complexity is still high when the number of shingle is high, the MinHash technique is used to reduce the size of characteristics matrix by generating the signature matrix with size hash number (hash#) in row and Documents number (D#) for the column. Bloom Filter (BF) data structure is implemented in the research. The BF reduces the compassion of n columns from O(n2) into O(n) because the signature matrix is still a big problem for some large compassions. The results are tested using four text files containing news for different newspapers (Chauhan and Batra, 2015).

Sumayia, Hind and Israa proposed similar documents using three cluster analysis k-mean, k-medoid and k-means fast. The data set was collected from the capstone project. The preprocessing step is used to tokenize and remove the unnecessary words with three lengths in these documents. The proposed method is presented by given a set of documents and divided them into multiple groups using the three-cluster analysis such that the documents in the same group are nearly similar to each other than the documents in the other groups. They concluded that the k-medoid show better performance than the k-means and k-means fast when Davies-Bouldin Index (DB) performance measure is used with cosine similarity. The ability cosine of cosine similarity measures ignores the document lengths to achieve (Al-Anazi *et al*., 2016).

The approach by Zamora *et al*. (2016) is used Local-Sensitive Hashing (LSH) which enables Jaccard similarity approximation and SimHash which approximates cosine similarity for high dimensional data to cluster a large number of real applications. Hamming distance with low dimensions is adopted to approximate the pairwise similarity which doesn't require data storage. the clustering bisecting method is applied to the similarity matrix. Experiential results show better running times and lower use of memory.

MATERIALS AND METHODS

Key components of the proposed method: This research is implemented using the following key concepts.

Jaccard similarity: The Jaccard similarity between two sets M and N is $|M \cap N|/M \cup N|$, that is the ratio of the size of the intersection of M and N to the size of their union (Rajaraman and Ullman, 2011):

$$JSim\ (M, N) = \frac{|M \bigcap N|}{|M \bigcup N|} \quad (1)$$

Jaccard similarity satisfies the following: JSim (M, N = 1) if and only if M = N; JSim (M, N) = 0 if and only if M≠N and JSim is symmetric. The most important use of the Jaccard similarity is finding the textual similar documents of large corpus such as the web (Niwattanakul *et al*., 2013). Similarity documents based on Jaccard similarity used in different applications such as mirror pages, plagiarism, articles from the same source, on-line purchase and movies rating. For the following two sets defined as the JSim = 4/6.

Table 1: Shingles of documents

Shingles	D_1	D_2	D_3
ab	1	1	1
bc	1	1	0
cd	1	1	0
bd	0	0	1
de	0	1	1

Set A = new set (["Java", "Programming", "Pure", "Object", "Oriented", "Language"])
Set B = new set (["Programming", "Java", "Object", "Language"])

K-shingles of documents: The most effective way to represent the documents as set to find the textual similarity is shingling the document into tokens of short strings that appear within it. In this case, the documents that have equivalent items between them will share common shingles. The (K) refers to substring of length (K) from the document. The choose of (K) should be picked large enough that the probability of any given shingle appearing in any given documents is low (Niwattanakul *et al*., 2013). To represent documents as set using K-shingle, we need to preprocess the documents by removing any punctuation mark, transform all letters to lower case, make a single space between words and remove white space up front and at the ends. Partitioning the text into separate words. For example, suppose the Documents D_1 has the following text {"abcdab"}. if the k = 2, the possible consecutive substring of length 2 found within Document D_2: {"ab", "bc", "cd", "da"}. If the document D_2 has the following text D2 = {"abcde"} and D_3 = {"abde"}. The characteristic matrix for three documents can be represented in Table 1.

The JSim is calculated between the documents in the Table 1. JSim (D_1, D_2) = 3/4, Jsim (D_1, D_3) = 1/5, Jsim (D_2, D_3) = 2/5. This way is infeasible for large documents such as (10^6 documents) because we need compassion (10^6/2) ≅ 6 days. In this study, we use K-shingle in different size.

Hashing of shingles: The space complexity for the shingling takes O(NK) for document with N words. Instead, the hash function can hash shingle with size (K) into number of buckets. The bucket number can be used instead of the substring. In this study, we use many of the hash function in the form in Eq. 2:

$$f(x) = ax+b \bmod p \quad (2)$$

Where:

x = An input shingle from the original matrix
a and b = Two randoms numbers
p = A prime number that is greater than the largest value of x

Suppose Table 1 shingles have been hashed to the bucket number 0-4 in Table 2 which is called characteristic matrix. This way is still infeasible for large documents to compare every pair of documents because the characteristic matrix M is sparse for large documents.

Table 2: Characteristic Matrix (M) of hashing shingles

Shingles	D_1	D_2	D_3
0	1	1	1
1	1	1	0
2	1	1	0
3	0	0	1
4	0	1	1

Table 3: Hash function for characteristic Matrix M of Table 2

Shingles	D_1	D_2	D_3	h_1= 2x+1 mod 7	h_2 = 3x +2 mod 7
0	1	1	1	1	2
1	1	1	0	3	5
2	1	1	0	5	1
3	0	0	1	1	4
4	0	1	1	2	7

MinHashing signature and Locality-Sensitive Hashing (LSH): MinHash comes to convert the large set into small signature while preserving the similarity between two pairs of the matrix. The goal is to find smaller columns from small signatures. The generated matrix is called signature matrix with row for hash functions and columns for documents. The main steps of MinHashing used in this study are illustrated in Algorithm 1.

Algorithm 1; MinHashing steps:

Input: Characteristic Matrix M, hash functions $h_1, h_2, h_3, \ldots, h_n$.
Output: Signature Matrix (S)
Begin
1. Picking n randomly hashing functions $h_1, h_2, h_3, \ldots, h_n$
2. Construct the signature Matrix S from characteristic Matrix M, where each row (i) is a hash function and each column (c) is a document. Then, set SIG(i,c) as signature matrix element for the ith hash h_i (r) function and column c
   2.1: Convert the long bit vector into short signatures. For each column c in documents do the following
      a. if c has 0 in both documents rows r, do nothing
      b. if row has 1, then for each I = 1,2 ,…, n set SIG(i,c) to the smaller value of the current value of SIG (I, c) and h_i(r)
   2.2 Then Pr[h_8(c1) = h_8(c2)] = sim(c1,c2)
3. If we choose the (h_1 = 2x+1 mod 7) and (h_2 = 3x +2 mod 7), Table 3 shows the signature matrix generated from the characteristic matrix in Table 3
End

Table 3 shows the hashing of the K-shingles using twso hash functions: h_1 = 2x+1 mod 7 and (h1) and h_2 = 3x+2 mod 7.

The signature matrix that is generated from Table 3 is illustrated in Table 4. The comparison results for Jaccard similarity JSim is illustrated in Table 5.

The signature matrix converts the bit vectors in M into a short signature in S. The comparisons for large documents is still high and need time. The last step is to find the most candidate pairs for comparison. LSH focus on choosing the candidate pairs of the signatures that are

hashing to the same buckets. The general idea is to divide the signature matric M into (b) bands of (r) rows using the Eq. 3:

$$n = br \quad (3)$$

Where:

n = A number of rows in S

b = A number of bands and is a number of rows in each band

The number of comparisons will reduce where each pair of documents hash to the same buckets is a candidate pair to be compared. The Similarity threshold St is set to choose the candidate pairs for performing the Jaccard similarity. The overall steps of this research are illustrated in Fig. 3. The overall step of this researcher is illustrated in Algorithm 2.

Table 4: Signature matrix (S) of Table 3

Hash functions	D_1	D_2	D_3
h_1	1	1	1
h_2	1	1	2

Table 5: Signature matrix of Table 5

Columns vs. signature values	1-2 columns	1-3 columns	2-3 columns
Column/column from (M)	3/4	1/4	2/4
Signature/signature from (S)	1	1/2	1/2

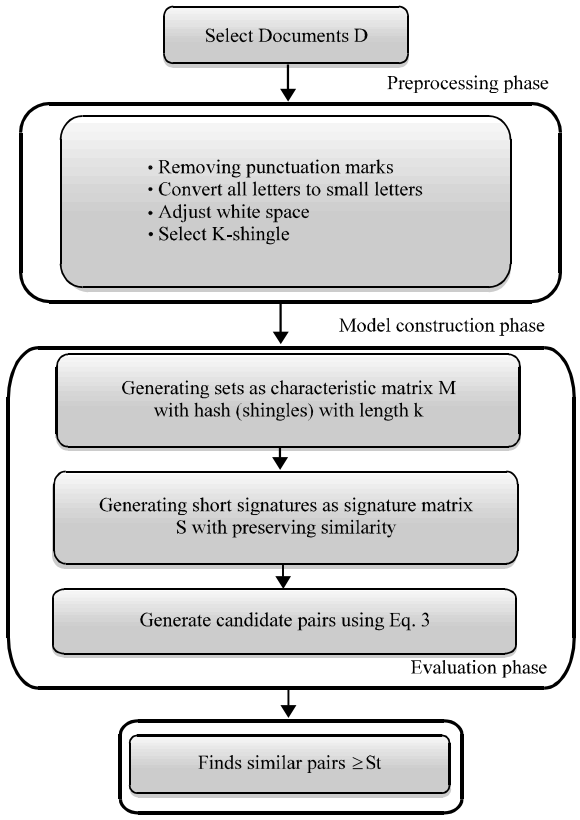

Fig. 3: The proposed system of documents similarity

Algorithm 2; The proposed system implementation algorithm:

Input: Documents $D_1, D_2, D_3, \ldots, D_n$
Output: Clustering similar documents
Begin
1. Set K = integer, then construct from each document a set of K-shingle
2. Generate characteristic Matrix M by hashing K-shingle into some buckets
3. Generate Signature matrix S from Algorithm 1, if the signature matrix short then stop, otherwise go to step 5
4. Set a threshold St. Pick a number of band (b) and a number or rows ® such that in Eq. 3. The St can be selected approximately as $(1/b)^{1/r}$ to adjust the false positive and false negative
5. Applying LSH to select a candidate pair and check similarity
6. Optionally, go to documents and check the similarity ratio

End

RESULTS AND DISCUSSION

Dataset collection: This research is implemented using the C# language with laptop specification: CPU Intel(R) Core ™ i5-2.50 GHz, 4 GB memory size and Windows 8.1 as operating system. The documents data is collected from different web sites and mainly from the websites (19, 20). There are different size of documents 100, 1000, 10000 on the available datasets.

Signature matrix: The system was implemented based on the different size of K-shingles. Table 6 shows the time using K-shingles in size from 3-10 for different numbers of documents 100, 200, …, 1000. The average time was 3 sec.

The time is increased when the number of documents increase too. The best K-shingle is 5 to find the similarity between documents.

The results show example of the signature matrix using different numbers of hash functions. Table 7 shows the output of the signature matrix. Figure 4 shows the number of documents versus time.

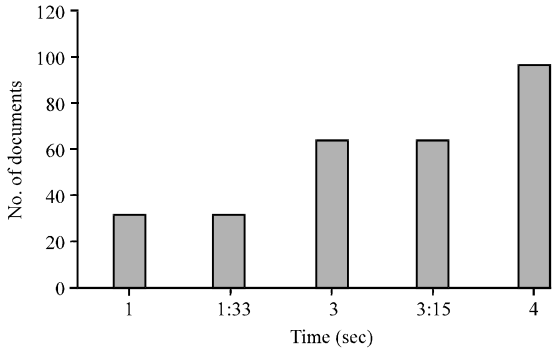

Fig. 4: Time of signature matrix with number of documents

Table 6: Time of variable K-shingle and documents

	Time (sec)						
No. of documents	Shingle k = 4	Shingle k = 5	Shingle k = 6	Shingle k = 7	Shingle k = 8	Shingle k = 9	Shingle = 10
100	1	1	1	1	1	1	1
200	1	1	1	1	1	1	1
300	1	1	1	1	1	2	2
400	2	3	2	2	2	2	2
500	2	2	2	3	2	3	2
600	3	3	3	3	3	3	3
700	4	4	4	3	3	4	4
800	5	4	5	4	4	4	5
900	5	4	5	5	4	4	5
1000	5	5	5	6	5	5	5

Table 7: Time versus number of documents, number of hash function and words length

Items	No. of hash functions	No. of documents	No. of words in documents	Average number of words	Time (sec)
1	10	33	8186	240	1
2	20	33	8186	240	1:33
3	10	66	25105	380	3
4	20	66	25105	380	3:15
6	20	100	105005	560	4

Table 8: The probability of one band

Similarity S between pairs of documents	Probability of sharing same buckets $1-(1-S^r)^b$	Candidate pairs (%)	$FP^*=1-(1-S^r)^b$, $FN^{**}=(1-S^n)^b$
0.2	0.006380581	0.64	FP
0.3	0.047494259	4.75	FP
0.4	0.186049552	18.61	FP
0.5	0.470050715	47.01	FP
0.6	0.801902454	19.81	FN
0.7	0.974780544	2.53	FN
0.8	0.999643942	0.01	FN
0.9	0.999999982	0.01	FN

*FP: False Positive that is dissimilar pairs hash to the same bucket; **FN: False Negative that is similar pairs do not hash to the same bucket

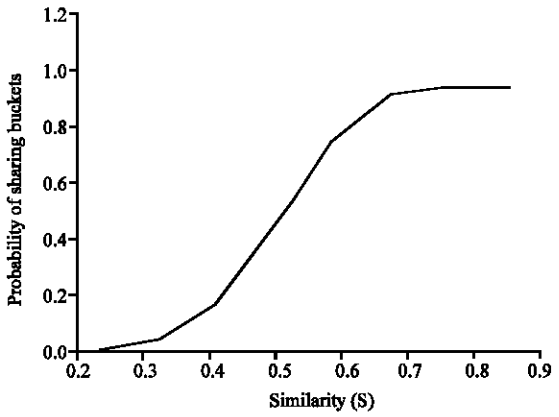

Fig. 5: Probability of sharing the same buckets versus Similarity of documents (S)

Finally, the Similarity threshold St to select the candidate pairs was 0.5. Table 8 shows the probability of at least one band is identical where b = 20 and n = 100 from Eq. 3, we could find 5 rows in each documents. Figure 5 shows the probability of sharing same buckets with the Similarity (S) as S-curve.

CONCLUSION

Documents similarity has become more interesting with the development of modern technology for the large numbers of users when access to the electronic sites for discarding the duplicate documents in the top results by the search engine. In this study, mining of documents similarity based on K-shingle is time consuming when the number of document increase suddenly. MinHash is a technique converting the large characteristics Matrix (M) into small Signature matrix (S) by choosing number of hash functions h_n. In addition, the work is proposed some of hash functions for compassions each pairs of documents. This way is still time consuming when the signature matrix become large. LSH technique comes to play a significant role to find a candidate pairs for different number of documents 100, 200, …, 1000. The candidate pairs is selected when the probability of two documents hash to the same bucket. Set Similarity threshold St to certain value is effective to choose the similarity distance between pairs of documents that is greater than the Similarity threshold St. The results show the false negative and false positive rates is minimum.

RECOMMENDATION

The future research will combine this research with one of the clustering technique using hybrid method to evaluate the research.

REFERENCES

Al-Anazi, S., H. AlMahmoud and I. Al-Turaiki, 2016. Finding similar documents using different clustering techniques. Procedia Comput. Sci., 82: 28-34.

Baraglia, R., G.D.F. Morales and C. Lucchese, 2010. Document similarity self-join with MapReduce. Proceedings of the 2010 IEEE 10th International Conference on Data Mining (ICDM), December 13-17, 2010, IEEE, Sydney, NSW, Australia, ISBN:978-1-4244-9131-5, pp: 731-736.

Bhaya, W. and M.E. Manaa, 2014. Review clustering mechanisms of distributed denial of service attacks. J. Comput. Sci., 10: 2037-2046.

Chauhan, S.S. and S. Batra, 2014. Finding similar items using LSH and bloom filter. Proceedings of the International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), May 8-10, 2014, IEEE, Ramanathapuram, India, ISBN:978-1-4799-3915-2, pp: 1662-1666.

Deng, F., S. Siersdorfer and S. Zerr, 2012. Efficient jaccard-based diversity analysis of large document collections. Proceedings of the 21st ACM International Conference on Information and Knowledge Management, October 29-November 02, 2012, ACM, Maui, Hawaii, USA., ISBN:978-1-4503-1156-4, pp: 1402-1411.

Hamilton, H., E. Gurak, L. Findlater and W. Olive, 2013. Computer science 831: Knowledge discovery in databases. Master Thesis, Department of Computer Science, University of Oxford, Oxford, England, UK.

Han, J. and M. Kamber, 2006. Data Mining: Concepts and Techniques. 2nd Edn., Morgan Kaufmann Publisher, San Fransisco, USA., ISBN-13: 978-1558609013, Pages: 800.

Manning, C.D., P. Raghavan and H. Schutze, 2008. An Introduction to Information Retrievel. Cambridge University Press, Cambridge, UK., ISBN:9780521865715, Pages: 482.

Niwattanakul, S., J. Singthongchai, E. Naenudorn and S. Wanapu, 2013. Using of jaccard coefficient for keywords similarity. Proceedings of the International MultiConference on Engineers and Computer Scientists (IMECS 2013) Vol. 1, March 13-15, 2013, MECS Publisher, Hong Kong, ISBN:978-988-19251-8-3, pp: 1-5.

Parziale, L., W. Liu, C. Matthews, N. Rosselot and C. Davis *et al*., 2006. TCP/IP Tutorial and Technical Overview. 8th Edn., IBM Redbooks, India, Pages: 963.

Peshave, M. and K. Dezhgosha, 2005. How search engines work: And a web crawler application. Ph.D Thesis, University of Illinois Springfield, Springfield, Illinois.

Phan, T.N., M. Jager, S. Nadschlager, J. Kung and T.K. Dang, 2015. An efficient document indexing-based similarity search in large datasets. Proceedings of the 2nd International Conference on Future Data and Security Engineering (FDSE 2015), November 23-25, 2015, Springer, Ho Chi Minh City, Vietnam, pp: 16-31.

Rajaraman, A. and J.D. Ullman, 2011. Mining of Massive Datasets. Cambridge University Press, UK., ISBN-13: 978-1107015357, Pages: 326.

Spasojevic, N. and G. Poncin, 2011. Large scale page-based book similarity clustering. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), September 18-21, 2011, IEEE, Beijing, China, ISBN:978-1-4577-1350-7, pp: 119-125.

Thella, P.P. and G. Sridevi, 2013.. A novel clustering method for similarity measuring in text documents. Intl. J. Modern Eng. Res., 3: 2823-2826.

Wang, C., Y. Song, H. Li, M. Zhang and J. Han, 2015. Knowsim: A document similarity measure on structured heterogeneous information networks. Proceedings of the 2015 IEEE International Conference on Data Mining (ICDM), November 14-17, 2015, IEEE, Atlantic City, New Jersey, USA., ISBN:978-1-4673-9504-5, pp: 1015-1020.

Zamora, J., M. Mendoza and H. Allende, 2016. Hashing-based clustering in high dimensional data. Expert Syst. Appl., 62: 202-211.

Zhang, Q., H. Ma, W. Qian and A. Zhou, 2013. Duplicate detection for identifying social spam in microblogs. Proceedings of the 2013 IEEE International Congress on Big Data (BigData Congress), June 27-July 2, 2013, IEEE, Santa Clara, California, USA., ISBN:978-1-4799-0182-1, pp: 141-148.