ACCENTS Transactions on Information Security, Vol 1(4)
ISSN (Online): 2455-7196

Research Article

Comparison of LSI algorithms without and with pre-processing: using text document based search

Sheikh Muhammad Saqib*, Khalid Mahmood and Tariq Naeem
Institute of Computing and Information Technology, Gomal University, Dera Ismail Khan, Pakistan

Abstract
Searching of document/text is the most important need of each student or computer user. Searching through particular index or term is the old fashion, now a day's user want to search documents according to some phrase, query or requirement i.e. extraction of meaningful information from large collection according to some textual query. Different methods such as iterative residual rescaling (IRR), term frequency (TF), inverse document frequency (IDF), multi words are using to handle such issues. Latent semantic indexing (LSI) is an important method for current literature of information retrieval. LSI can find similar documents on particular textual phrase. Here author has implemented two algorithms (without and with pre-processing) of LSI for text documents. As a result, both algorithms can obtain the similar results but their processing time will be different.

Keywords
Iterative residual rescaling, Term frequency, Inverse document frequency, Latent semantic indexing, Pre-processing.

1.Introduction
Mining of selected document is the key element of any study. Suppose you need some documents related to sentiment analysis, there are lot of PDF files available online or offline. You can download hundreds related file in large scope i.e. documents of sentiment analysis. But at the time of study, you need only those documents which are related to phrase "sentiment analysis using supervised learning". Instead of finding related document manually, there should be a method which can automatically find those documents which are related to this text. LSI can provide a help for such issues. LSI is very easy to understand, implement and use. Results of LSI are very decent and faster compared to other methods. It aims to find the most representative features for document representation rather than the most discriminative ones [1, 2]. Major purpose of cosine similarity measure in LSI is to rank the data with respect to query, where data means stored documents and query is user requirement in text format. Here author has proposed and implemented an latent semantic indexing (LSI) approach using pre-processing & without pre-processing and concluded that both algorithm will work correctly but algorithm with pre-processing have less processing time and greater manual work (finding stop words, how to remove them, how to convert in lower case etc.).

*Author for correspondence

The algorithm without pre-processing is the reverse of previous one. Purpose of both algorithms with pre-processing and without pre-processing is same. Both rank the documents in descending order with respect to query. Here we are supposing following table to represent such concept. In *Table 1*, D is document, A is algorithm with pre-processing, and B is algorithm without pre-processing.

Table 1 Supposing values of D based on A and B

D	A	B	Ranking on A	Ranking on B
D1	0.8	0.7	D2	D2
D2	0.9	0.8	D1	D1
D3	0.4	0.2	D4	D4
D4	0.6	0.5	D3	D3

In *Table 1*, column-1 contains documents, column-2 contains similarity cosine values using algorithm-1, and column-3 contains similarity cosine values using algorithm-2. Resultant columns column-4 and column-5 representing the ranking of documents i.e. ranking of documents through both algorithms is same. Now, here we prove this supposition.

2.Related work
In cluster tree [3], hybrid similarity has been measured by using LSI and LSI is used to cluster clinical document [25]. Authors in [4] have applied LSI to find representation of concept by mapping the terms and phrases with document and then clustering

them. LSI and independent component analysis (ICA) [5, 6] have been used to find latent semantic structures in dataset each structure is a linear combination of the original features i.e. words. Using LSI approach, information retrieval methods has been proposed by the authors [7] using text documents. Sprinkling [9] which is the extension of LSI to supervised classification tasks and generating revised document representations that can be used by any technique founded on the vector space model. As LSI ignores class labels of training documents, sprinkling can handle such issue. Real world applications of topic modelling are limited due to issues of scalability.

Regularized latent sematic indexing (RLSI) is designed for parallelization and can handle large dataset without reduction of input vocabulary [11]. Term frequency-inverse document frequency (TF-IDF), LSI and multi-word is used for extraction of feature which is helpful for identification of important words in a text document [12]. Main goal of probabilistic latent semantic analysis (PLSA) is to model co-occurrence information under a probabilistic framework in order to discover the underlying semantic structure of the data [13].

Multilevel Latent Semantic Association method grouped the words in aspect expression for aspect expression of latent topic structure [14]. General text parser (GTP) based on LSI, parse a huge collection of documents and create a vector space information retrieval model for subsequent concept-based query processing [15]. Sentiment analysis means analyzing the people opinion as positive or negative [17]. The research on sentiments and opinions appeared in 2001 [18] and 2002 [19]. LSI and Machine learning has been used for multi-lingual sentiment analysis [16].

To improve the efficiency of LSI, different researchers are working on different extensions of LSI i.e. singular value rescaling (SVR) based on LSI made experiments on text REtrieval conference (TREC) dataset showing the 5.9% best results than LSI [20], dynamic hybrid cut improves the effectiveness of the LSI approach for detecting concerns in source code [21] and a term-to-concept projection matrix has been developed to reduce dimension for decreasing the bottleneck of LSI [24].

Extended method based on LSI is able to filter the unwanted emails of Chinese and English [23]. In advanced search, human not only require index term information but also concept and ideas. Such concept based searching and automatic key extraction can be done through LSI [26]. After the comparative study of multi-words, TF-IDF and LSI on text classification, the experimental results are showing that LSI has best performance than other two techniques [22]. LSI can resolve the problem of lexical matching by using statistically derived conceptual indices [10]. In [8], it is observed that after the evaluation of documents, LSI performed 40% better compared to exact term matching techniques.

3.Documents ranking through LSI algorithm

LSI proposed by Deerwester in 1990 is an efficient information retrieval algorithm [7]. Basically in LSI, there is cosine similarity measure between coordinates of a document vector and coordinates of query vector. If this value is 1, means document is 100% closer to query, if it 0.5 means document is 50% closer and it is 0.9 means document is 90% closer with query.

Now the major point is that how we can find the coordinates of each document and query. Singular value decomposition (SVD) can determine the points or coordinates of documents and query. Through SVD, three matrices S, V and U can be determined by a matrix which will be used for further processing. To determine the values of such variables, SVD requires a matrix. Matrix consists of rows and columns containing integer values while here inputs are different text documents.

Feature matrix can be obtained by calculating the frequencies of each word. It means, first of all we will make feature matrix from all documents and then will calculate SVD as shown in Algorithm in *Table 2* from line 1-4. Line 5 and 6 will made a matrix for query. After this supporting variables S, V and U will be calculated by using numpy (Numeric Python). Now, from S, coordinates of all documents will be determined and these coordinates will be emerged with query to find query coordinates. At last, cosine similar function will be applied on these coordinates to fined closest documents to query.

3.1Algorithm for documents without Pre-processing

Table 2 Algorithm of LSI with stop words

1. **Input:** All Documents and Query
2. Tokenize All Documents
   Token=Token(All Documents)
3. Take The Union Set of Tokenized Documents
   Uniont=Union(Token)
4. Make Frequency Matrix From Uniont
   Fmat=Frequencymatrix(Uniont)
5. Make Query Matrix
6. Qmat=Querymatrix(Token(Query))
7. Decompose Frequency Matrix In U,S,V Using Svd From Usvt
8. Determine V From Vt
9. Find Uk,Vk And Sk
10. Uk = Extracting First Two Column of U
11. Vk = Extracting First Two Column of V
12. Sk= Extracting First Two Column and Row of S
13. Each Row of V Relates To Coordinates of Document
14. Find Coordinates of Query From **Q = Qtuksk-1**
15. First We Will Find Sk Inverse From Sk-→10
16. Second Q Transpose From Query Matrix →4
17. Uk Is Already Determined → 8
18. Now, Find **Q = Qtuksk-1**
19. Q Have Coordinates Of Query
20. Find Dot Product Of Q With Each Document Coordinates (→13)
21. Sort Dot Product Values In Descending Order
22. Output Ranking Of Documents With Respect To Query

We have checked above algorithm by taking three documents (d1="talcum powder has beautiful fragrance", d2=" talcum powder is white color", d3="black cat talcum powder") and a query (qry=" talcum powder is black cat") as input. In advance we know that d3 is very closest to query. *Table 3* is depicting the results of given inputs.

Table 3 Results of algorithm1

1. d1="talcum powder has beautiful fragrance" 2. d2=" talcum powder is white color" 3. d3="black cat talcum powder" qry=" talcum powder is black cat"		
Tokens ['telcome', 'powder', 'has', 'beautiful', 'fragrence'] ['telcome', 'powder', 'is', 'white', 'color'] ['black', 'cat', 'telcome', 'powder']		
Token Sets set(['beautiful', 'fragrence', 'has', 'telcome', 'powder']) set(['color', 'is', 'white', 'telcome', 'powder']) set(['telcome', 'black', 'powder', 'cat'])		
Union set(['beautiful', 'fragrence', 'color', 'is', 'cat', 'black', 'powder', 'white', 'has', 'telcome'])		
Feature Matrix [[1 1 0 0 0 0 1 0 1 1]] [[0 0 1 1 0 0 1 1 0 1]] [[0 0 0 0 1 1 1 0 0 1]]	Query Matrix [[0 0 0 1 1 1 1 0 0 1]]	
S [2.94984103 0. 0.] [0. 1.73205081 0.] [0. 0. 1.51605999]	V [[-0.605 0.707 -0.364]] [[-0.605 - 0.707 -0.364]] [[-5.15 9.697 8.568]]	U It is large matrix, we will display UK.
SK [[2.94984103 0.] [0. 1.73205081]]	VK [[-0.605 0.707] [-0.605 - 0707] [-0.515 9.697]]	UK [[-2.05405238e-01 4.08248290e-01] [-2.05405238e-01 4.08248290e-01] [-2.05405238e-01 - 4.08248290e-01] [-2.05405238e-01 - 4.08248290e-01] [-1.74754886e-01 6.56816799e-16] [-1.74754886e-01 6.56816799e-16] [-5.85565363e-01 2.46574729e-16] [-2.05405238e-01 - 4.08248290e-01] [-2.05405238e-01 4.08248290e-01] [-5.85565363e-01 2.46574729e-16]]
Coordinates of All Docs from VK [-0.605,0.707],[-0.605, -0.707], [-0.515, 9.697189]	SK Inverse [[0.33900, 0.0], [0.0, 0.57735]]	Coordinates of Resultant Query from Query Matrix UK and SK-1 [[-0.58513178 -0.23570226]]
Qry= telcome powder is black cat Results D1= telcome powder has beautiful fragrence= 0.319826412535 D2= telcome powder is white color= 0.887280361339 D3= black cat telcome powder= 0.927572256443		

From *Table 3*, it is clear that d3 (92%) is very close to query, d2 (88%) is close after d1 and d3 (31%) is close after d2.

3.2Validation of algorithm-1

To check the validity of algorithm, we can take a document similar to query. In above Algorithm1 when we have assigned another document d4 (d4=telcome powder is black cat) same as query (Qry= telcome powder is black cat) then result of d4 (100%) was 1.0 i.e. algorithm is working well, because query and d4 have same contents.

Table 4 Results of algorithm-1 by taking input document same as query

QRY= telcome powder is black cat Results
D1= telcome powder has beautiful fragrence= 0.319826412535
D2= telcome powder is white color= 0.887280361339
D3= black cat telcome powder= 0.927572256443
D4= telcome powder is black cat= 1.0

3.3Algorithm for documents with Pre-processing

In pre-processing there is lot of work i.e. converting each token in lower case, remove stop words, lemmatization, stemming, seeding etc. Here we are removing stop words and converting them into lower case. We have collected stop words (stopw) and special stop words (exStopw) from union result of algorithm-1 in *Table 4*. First we will remove all stop words from exStopw then from stopw, differ is shown in following *Table 5*.

stopw=["This","being", "as", "we", "have", "where", "been", "has", "had", "is","the",".","\n","on","in","of","from","to","I","we", "it","there","for","their","our","and","due","a","this", "that","about","through","or","may","be","an","by"," etc","can","also","these"]
exStopw=[".",",",":",":",";","?","/"]

Table 5 Steps for stop words removal

Document	Tokens	Order of removing stop words	Out put
Mining is a…. big field	Mining, is, a…, big, field	i) Remove stopw ii) Remove exStopw	i) Mining, a…, big, field **ii) Mining, a, big, field**
Mining is a…. big field	Mining, is, a…,big,field	i) Remove exStopw ii) Remove stopw	i) Mining, is, a, big, field **ii) Mining, big, field**

From above *Table 5*, if we choose first process then final output will be “mining, a, big, field” which contains stop word ‘a’ because in input document there is a word ‘a…’ while second process have output “mining, big, field” which have removed all stop words. Now adding following code in *Table 2* from line 3 to 5.

Table 6 Algorithm-2 LSI with Pre-processing

```
Input: Different Documents and A Query
Output: Ranking Of Documents Related To Query
Stopw=["This","Being", "As", "We", "Have", "Where", "Been", "Has", "Had", "Is","The",".","\N","On","In","of","From","To","I","We","It","There","For","Their","Our","and","Due","A","This","That","About","Through","or","May","Be","An","By","Etc", "Can","Also","These"]
Exstopw=[".",",",":",":",";","?","/"]
Tokens = Union(Tokenize(All Documents))
## Tasks
## (I) Count Last Characters Related To Exstopw
## (Ii) Delete Last Counted Characters From Word
## (Iii) Count Last Characters Related To Stopw
## (Iv) Delete Last Counted Characters From Word
Remove Last Character of Word
For Word In Tokens
 If Word Ends With any Element of Exstopw
    Do Task (I) and (Ii)
 If Word Ends With any Element of Stopw
    Do Task (Iii) and (Iv)
```

Again we took similar inputs as in Algorithm-1 i.e. three documents (d1="talcum powder has beautiful fragrance", d2=" talcum powder is white color", d3="black cat talcum powder") and a query (qry=" talcum powder is black cat").-2. Following *Table 7* is showing the results of Algorithm-2.

Table 7 Results of algorithm 2

Qry= telcome powder is black cat results
D1= telcome powder has beautiful fragrence= 0.422535
D2= telcome powder is white color= 0.6180361339
D3= black cat telcome powder= 0.9989216

From *Table 7*, it is clear that d3 (99%) is very close to query, d2 (61%) is close after d1 and then d3 (42%) is close after d2.

Table 8 Similarity percentages of documents from both algorithms

D	A=Algorithm-1	B=Algorithm-1	RankinG on A	RankinG on B
D1	0.319826	0.422535	D3 (31%)	D3

D	A=Algorithm-1	B=Algorithm-1	RankinG on A	RankinG on B
				(42%)
D2	0.8872803	0.618036	D2 (88%)	D2 (61%)
D3	0.9275722	0.998921	D1 (92%)	D1 (99%)

Hence from *Table 8*, it is clear that similarity percentages of documents with query is different in both algorithm but the results on base of descending order is same i.e. D3>D2>D1. This result is same as we have supposed in *Table 1*.

3.4Comparison of both algorithm

Now we are taking abstract of 5 papers as (D2>D1>D3>D4>D5), manually we have checked, where D2 is very closely relevant to query while D5 is not related with query.

Table 9 Input documents as abstract of different papers

D1	This Paper Presents An Unsupervised Approach To Aspect-Based Opinion Polling From Raw Textual Reviews Without Explicit Ratings. The Key Contribution of This Paper Is Three-Fold. First, A Multi Aspect Bootstrapping Algorithm Is Proposed To Learn From Unlabeled Data Aspect-Related Terms of Each Aspect To Be Used For Aspect Identification. Second, An Unsupervised Segmentation Model Is Proposed To Address The Challenge of Identifying Multiple Single-Aspect Units In A Multi-Aspect Sentence. Finally, An Aspect Based Opinion Polling Algorithm Is Presented. Experiments On Real Chinese Restaurant Reviews Show That Our Opinion Polling Method Can Achieve 75.5% Precision Performance.
D2	In this paper, we propose a review selection approach towards accurate estimation of feature ratings for services on participatory websites where users write textual reviews for these services. Our approach selects reviews that comprehensively talk about a feature of a service by using information distance of the reviews on the feature. The rating estimation of the feature for these selected reviews using machine learning techniques provides more accurate results than that for other reviews. The average of these estimated feature ratings also better represents an accurate overall rating for the feature of the service, which provides useful feedback for other users to choose their satisfactory services.
D3	The “Aspect Based Sentiment Analysis” task focuses on the recognition of aspect term and category and classification of emotions (positive, negative, conflict, neutral) in restaurant reviews for the aspect. In this paper we propose the system for recognizing aspects and analyzing the sentiments using SVM for the restaurant review dataset. We compare the performance of the system with well-known KNN classifier.
D4	Spam Detection Consumers increasingly rate, review and research products online (Jansen, 2010; Litvin et al., 2008). Consequently, websites containing consumer reviews are becoming targets of opinion spam. While recent work has focused primarily on manually identifiable instances of opinion spam, in this work we study deceptive opinion spam—fictitious opinions that have been deliberately written to sound authentic.
D5	This research paper represents a multi-agent system, which have four Agents named as Knowledge Acquisition Agent, Attendance Agent,Decision Making Agent and Communication Agent that works together to that automatically gets inputs, manipulates the data, prepares timetable as well as keeps the record of students’ attendance and makes communication with its environment in an automatic fashion through sensors. All the agents work like human agents, which is one of the basic aims of computer technology. This work depicts an idea to integrate the Human Expertise, Information as well as the Biometric Technologies to solve real world problems. Feedback may be used as a learning element in the processing of the Multi-agent system. Snapshots (i.e., time table preparation, Attendance records, decision about absenteeism etc) depict how the various results are being provided by this multi-agent system to help human. This system can easily be implemented through adaptation of Biometric Technology and may also be used for employees’ attendance record as well as for security purposes, in future research.
Query	The rating estimation of the feature for these selected reviews using machine learning techniques and experiments on real Chinese restaurant reviews provides more accurate results than that for other reviews.

After taking above documents and query as inputs for Algorithm-1 shown in *Table 2* and Algorithm-2 shown in *Table 6*, we have obtained the following results from both as shown in *Table 9.*

Table 10 Results from both algorithms

Documents	A=coordinates-algo1	B=coordinates-algo2	Similarity values based on A	Similarity values based on B
D1	[-0.26306995581813425, 0.17706615964420416]	[-0.26306995581813425, 0.17706615964420416]	0.975432470662	0. 98282967271
D2	[-0.48315134771552765, 0.763917350444424]	[-0.48315134771552765, 0.763917350444424]	0.980494811666	0. 999986475172
D3	[-0.3200168157115258, 0.1295143355108754]	[-0.3200168157115258, 0.1295143355108754]	0. 929122463932	0. 898978659695
D4	[-0.09911747894964751, 0.053488425748337724]	[-0.09911747894964751, 0.053488425748337724]	0. 964179178316	0. 891008231649
D5	[-0.7649339426218684, -0.604518719389029]	[-0.7649339426218684, -0.604518719389029]	0. 13823609022	0. 0321949520955
Query Coordinates	[-0.30767381 0.29438224]	[-0.30767381 0.29438224]	1.0	1.0

Table 10 is representing the results of both algorithms on five documents. Manually we have selected D2 is very close to query and D5 is very far from query and ranking was D2>D1>D3>D4>D5. From results of both algorithms D2 (98% from Algorithme-1, 99% from Algorithm-2) is very close to query and D5 (13% from Algorithme-1, 32% from Algorithm-2) is very far from query. Hence from both algorithms ranking is D2>D1>D3>D4>D5. Now to obtain processing time, we have find size of each matrix as shown in *Table 11*

Table 11 Size of matrices in algorithm-1 and algorithm-2

Algorithm	U	V	S	UK	Query matrix	Feature matrix
Algorithm-1	86436	25	5	588	294	1470
Algorithm-2	53824	25	5	464	232	1160

Hence it is clear that U, UK, Query Matrix and Feature Matrix of Algorithm-2 have less size then that of Algorithm-1. But for Algorithm-2, we will consider some time for Pre-Processing.

4.Conclusion

Text based intelligent information processing is the requirement of each internet user. They use search-engine for retrieving information on the bases of sentence not on bases of particular word. Also users of computers want to search the documents from existing thousand stored documents. It is very hard to search out required documents from stored document manually. There is lot of work is related to such issue. After exploring the all techniques, LSI is a best method for retrieving the information. LSI has better semantic and statistically quality [22] and text retrieval is the current literature of LSI. We have implemented two algorithms (without Pre-Processing and with Pre-Processing) of LSI and found that both the results are same with respect to ranking of documents. Here in *Table 12* we have made different attempts on these algorithms to find out the maturity of these algorithms.

Here we took five samples of documents. Each sample consist of three (D1, D2, D3) text documents.

Sample-1: Manually we have considered D3 is very close to Query. After applying both algorithms on sample-1, we have obtained percentages of D3 from Algorithm-1 99% and also 99% from Algorithm-2.

Sample-2: Manually we have considered D2 is very close to Query. After applying both algorithms on sample-2, we have obtained percentages of D2 from Algorithm-1 85% and 99% from Algorithm-2.

Sample-3: Manually we have considered D3 is very close to Query. After applying both algorithms on sample-3, we have obtained percentages of D3 from Algorithm-1 99 and also 99% from Algorithm-2.

Sample-4: Manually we have considered D2 and D3 is very close to Query. After applying both algorithms on sample-4, we have obtained percentages of D2 & D3 from both algorithms 93% & 97%.

Sample-5: Manually we have considered all irrelevant documents. After applying both algorithms on sample-5, we have obtained percentages of 0% or less than 0%.

Table 12 is representing that both algorithms are working well while processing time of algorithm-1 is greater than algorithm-2 because matrices' size of algorithm-1 is greater than algorith-2. Following *Figure 1* is representing that algorithm-1 has greater processing time than algorithm-2 with respect to 5-samples of documents.

Table 12 Different attempts on both algorithms

Samples	Algos	D1	D2	D3	Automatic identified similar doc with query	Manually identified similar doc with query	Size of feature matrix	Size of u Matrix
Sample-1	Algo-1	0.00731759340968	0.09664019845	0.99404749508	D3	D3	789	69169
	Algo-2	0.202726610236	0.0277404903936	0.999587952509	D3	D3	648	46656
Sample-2	Algo-1	0.561377445816	0.853152864627	0.791933831548	D2	D2	840	78400
	Algo-2	0.989564559362	0.998464409368	-0.0567068171327	D2	D2	684	51984
Sample-3	Algo-1	-0.0424976035434	0.780859494526	0.998660969105	D3	D3	780	67600
	Algo-2	0.113754028079	0.741611395428	0.99218523483	D3	D3	609	41209
Sample-4	Algo-1	-0.343125747237	0.939289476989	0.939289476989	D2,D3	D2,D3	780	67600
	Algo-2	0.212232168853	0.977219272479	0.977219272479	D2,D3	D2,D3	609	41209
Sample-5	Algo-1	0.784547360818	-0.617292208796	-0.627806233247	No	No	426	20164
	Algo-2	0	0	0	No	No	318	11236

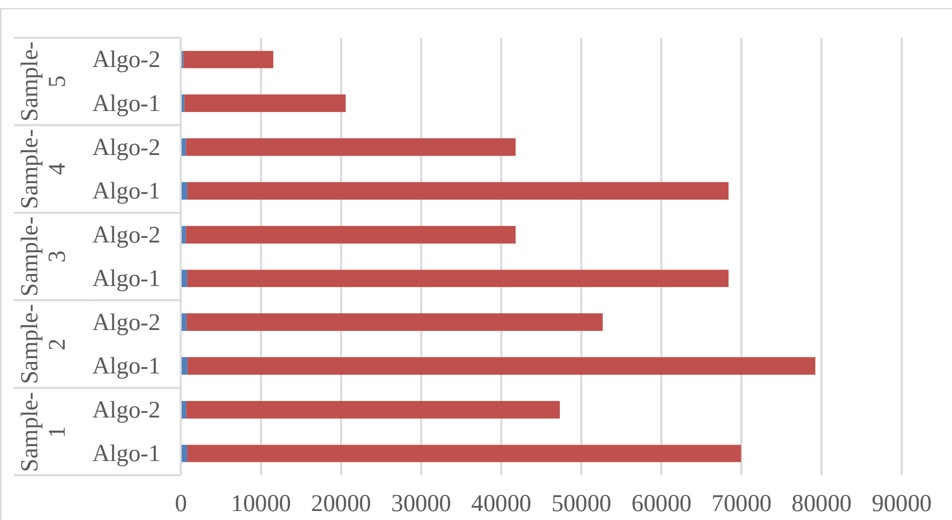

Figure 1 comparison of both algorithms on five samples

Acknowledgment

None.

Conflicts of interest

The authors have no conflicts of interest to declare.

References

[1] Sun JT, Chen Z, Zeng HJ, Lu YC, Shi CY, Ma WY. Supervised latent semantic indexing for document categorization. In fourth IEEE international conference on data mining 2004 (pp. 535-8). IEEE.

[2] What are the advantages and disadvantages of Latent Semantic Analysis? http://www.quora.com/What-are-

the-advantages-and-disadvantages-of-Latent-Semantic-Analysis. Accessed 20 October 2015.
[3] Devi MV. Cluster tree based hybrid document similarity measure. Compusoft.2014; 3(1):494-8.
[4] Huang A, Milne D, Frank E, Witten IH. Clustering documents using a Wikipedia-based concept representation. In pacific-Asia conference on knowledge discovery and data mining 2009 (pp. 628-36). Springer Berlin Heidelberg.
[5] Deerwester S, Dumais ST, Furnas GW, Landauer TK, Harshman R. Indexing by latent semantic analysis. Journal of the American Society for Information Science. 1990; 41(6):391-407.
[6] Kolenda T, Hansen LK, Sigurdsson S. Independent components in text. In advances in independent component analysis 2000 (pp. 235-56). Springer London.
[7] Phadnis N, Gadge J. Framework for document retrieval using latent semantic indexing. International Journal of Computer Applications. 2014; 94(14).
[8] Sadjirin R, Rahman NA. Efficient retrieval of Malay language documents using latent semantic indexing. In international symposium on information technology 2010 (pp. 1410-5). IEEE.
[9] Chakraborti S, Mukras R, Lothian R, Wiratunga N, Watt SN, Harper DJ. Supervised latent semantic indexing using adaptive sprinkling. In IJCAI 2007 (pp. 1582-7).
[10] Barbara Rosario, Latent Semantic Indexing: An overview, INFOSYS 240 Spring 2000.
[11] Wang Q, Xu J, Li H, Craswell N. Regularized latent semantic indexing. In proceedings of the 34th international ACM SIGIR conference on research and development in information retrieval 2011 (pp. 685-94). ACM.
[12] Govindarajan M. Sentiment analysis of movie reviews using hybrid method of naive Bayes and genetic algorithm. International Journal of Advanced Computer Research. 2013; 3(4):139-45.
[13] Oneata D. Probabilistic latent semantic analysis. 2011.
[14] Guo H, Zhu H, Guo Z, Zhang X, Su Z. Product feature categorization with multilevel latent semantic association. In proceedings of the 18th ACM conference on information and knowledge management 2009 (pp. 1087-96). ACM.
[15] Mahesh TR, Suresh MB, Vinayababu M. Text mining: advancements, challenges and future directions. International Journal of Reviews in Computing. 2010; 3:61-5.
[16] Bader BW, Kegelmeyer WP, Chew PA. Multilingual sentiment analysis using latent semantic indexing and machine learning. In IEEE 11th international conference on data mining workshops 2011 (pp. 45-52). IEEE.
[17] Liu B. Sentiment analysis and opinion mining. Synthesis lectures on human language technologies. 2012; 5(1):1-67.
[18] Das S, Chen M. Yahoo! for Amazon: extracting market sentiment from stock message boards. In proceedings of the Asia pacific finance association annual conference (APFA) 2001 (p. 43).
[19] Morinaga S, Yamanishi K, Tateishi K, Fukushima T. Mining product reputations on the web. In proceedings of the eighth ACM SIGKDD international conference on knowledge discovery and data mining 2002 (pp. 341-9). ACM.
[20] Yan H, Grosky WI, Fotouhi F. Augmenting the power of LSI in text retrieval: singular value rescaling. Data & Knowledge Engineering. 2008; 65(1):108-25.
[21] Van der Spek P, Klusener S. Applying a dynamic threshold to improve cluster detection of LSI. Science of Computer Programming. 2011; 76(12):1261-74.
[22] Zhang W, Yoshida T, Tang X. A comparative study of TF* IDF, LSI and multi-words for text classification. Expert Systems with Applications. 2011; 38(3):2758-65.
[23] Yang Q, Li FM. Support vector machine for customized email filtering based on improving latent semantic indexing. In international conference on machine learning and cybernetics 2005 (pp. 3787-91). IEEE.
[24] Hao J, Liao L, Dong X. Improving latent semantic indexing with concepts mapping based on domain ontology. In international conference on natural language processing and knowledge engineering 2008 (pp. 1-6). IEEE.
[25] Han C, Choi J. Effect of latent semantic indexing for clustering clinical documents. In IEEE/ACIS 9th international conference on computer and information science 2010 (pp. 561-6). IEEE.
[26] Rodrigues R, Asnani K. Concept based search using LSI and automatic keyphrase extraction. In 3rd international conference on emerging trends in engineering and technology 2010 (pp. 573-7). IEEE.