ISSN:0975-9646

# Text Search Optimization Using Latent Semantic Indexing

**Navneet kaur Waraich**
*Computer Science Department*
*Punjab Technical University*
*Punjab, India*

**Hardeep Singh Sindher**
*Computer Science Department*
*Punjab Technical University*
*Punjab, India*

***Abstract*— LSI is a powerful, generic practice which is able to index any document collection. It can be used in combination with an ordinary keyword search, or on behalf of one, with good quality results. Latent Semantic indexing for information retrieval (IR) and text mining operation on large data sets makes the optimized search result which is based on content rather than simple search. This helps search engine to more optimize in term retrieval processes. The proposed algorithm offered an approach combining clustering algorithm by means of latent semantic indexing (LSI) to retrieve the information.**

## I. Introduction

Standard keyword searches approach a document collection which contains a certain word or it doesn't. Prior to the search algorithm - the interdependency of any kind was not there between documents which are estimated on their contents. Latent semantic indexing adds a crucial step to the document indexing process. Additionally for recording which keywords a document contains, the procedure examines the document collection all together, to see which other documents include some of those similar words. LSI considers documents that have numerous words in general to be semantically close and ones with few words in general to be semantically isolated. The simple way correlates how a human being looking at content classifies a document collection.

Even though the latent semantic indexing algorithm does not know anything about what the words denote, the patterns it notices can make it appear surprisingly bright. When LSI indexed database is explored than the search engine crossways the similarity significance it has calculated for every content word and hence return the documents that fits the best query. Latent semantic indexing often returns relevant documents that do not contain the keyword in any way. For illustration: If the words manifold, n-dimensional and topology appear together in enough piece of writing, the search algorithm notices that the three terms are semantically close. This hunt for n dimensional manifolds will therefore return a set of items containing that phrase, but as well articles that contain just the topology word. This search engine understands zero about mathematics, but examining enough amount of documents train it that the three words are linked. It then uses that data to present a set of results with enhanced search than a basic keyword search.

## II. Related Work

Ahmed et al. [1] represented the evaluation of a summary on how close it is to the chief points in the source text. The salience, the frequency, and the relationship of the text with other texts in the collection were significant. Text categorisation using neural networks explicated the points and also has a practical impact. Hynek et al. [2] described the significance of automatic document summarization which increased with the threat of information overload we are facing. The proposed algorithm used inductive machine learning methods for automatic document classification. A novel text categorization method called Itemsets Classifier was developed. Besides other techniques, co-occurrence of terms to summarize text documents was also used. Mirzal [4] represented that the main trouble in using singular value decomposition is its retrieval performance depends strongly on the choosing of an appropriate decomposition rank. The proposed matrix completion algorithm produced a unique solution for each input. The proposed algorithm used similarity measures to find the related vertices, and then modified weights of the existing edges and/or created new edges based on the measures.

Radev et al. [5] described a large scale meta estimation of eight estimation measures for both single-document and multi-document summarizers. Together the qualitative and quantitative results were presented showing the strengths and draw-backs of all evaluation methods and how they ranked the different summarizers. Valle-Lisboa et al. [6] represented latent semantic analysis (LSA), a well-known method for information retrieval. It had also been applied as a model of cognitive processing and word-meaning acquisition. The proposed method detected an underlying block structure in the term-by-document matrix. In real cases this block structure is hidden. The correct explanation for LSA is searched in the structure of singular vectors rather than in the profile of singular values.

## III. Search For Content

Latent semantic indexing come across patterns of word allocation for a set of documents. Prior to we talk about the numerical underpinnings, look at the kind of words latent semantic indexing come across at.

Ordinary language is occupied by redundancies and not each word that appears in a text carries semantic sense. In reality, English have numerous frequently used words which do not hold content at all: useful words, prepositions, auxiliary verbs, conjunctions and others. The initial move in latent semantic indexing is gathering all those inappropriate words from a document and put down only content words which enclose semantic importance. There are number of ways to describe a content word but one mode for creating a record of content words from a document collected works is as follows:

1. Formulate a whole list of all the words that emerge anywhere in the set
2. Remove articles, prepositions, and conjunctions
3. Discard familiar verbs (know, see, do, be)
4. Dispose of pronouns
5. Discard common adjectives (big, late, high)
6. Remove extra words (therefore, thus, however, although, etc.)
7. Discard any words that appear in each document
8. Remove any words that appear in only single document set

The method reduces the documents into sets of content words so that we can use it to index the collection afterwards. Generate a term document matrix using the list of content words and documents, among documents listed along the horizontal axis and content words along the vertical axis. We go across the appropriate row for each content word in the list and place an 'X' in the column for every document where that word appears. We leave that column empty if the word does not appear. Repeating the procedure for every phrase and text in our collection results mainly in the form of empty grid with a sparse scattering of X-es. The grid shows everything about the document collected works. By looking for X-es in the appropriate column we can list all the content words in any specified document, otherwise we can find all the documents containing a certain content word by looking across the suitable row in the matrix.

Our arrangement is binary - a square in the grid either contains an X or it does not. The counterpart of a common keyword search is the large grid which looks for exact matches between documents and keywords. By substituting blanks and X-es with zeroes and ones, we get a numerical matrix containing the same data.

The answer step in latent semantic indexing is decomposing the matrix using a method called singular value decomposition. For example: Imagine that we are interested about what people normally order for breakfast downwards at our local diner and we want this information to be displayed in visual form. We examine all the breakfast orders from a hectic weekend day and record the words bacon, coffee and eggs occur in each order for how many times.

The results of our study are presented in a graphical manner by locating up a chart with three orthogonal axes-one for each keyword described. The preference of direction is random - perhaps a bacon axis in the x direction, an eggs axis in the y direction and the coffee axis in the z direction. We calculate the occurrence of each keyword to plan a particular breakfast order and then capture the right number of steps all along the axis for that word. As soon as we are finished, we get a cloud of points in three-dimensional space, representing all of that day's breakfast orders.

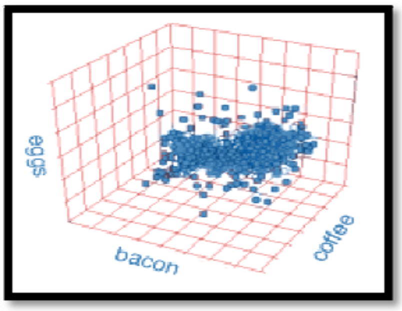

Figure 1: Displaying information in visual form

If a line is drawn from the starting point of the graph to each one of these points, we acquire a set of vectors in 'bacon-eggs-and-coffee' space. The key items were determined in any particular order through the size and direction of each vector and the set of all the vectors taken as one tells something about the type of breakfast populace favor on a Saturday morning.

Our graph illustrates a term space. Each breakfast order forms a vector in that space with its direction and magnitude determined by how many times the three keywords emerge in it. Every keyword corresponds to a separate spatial direction which is vertical to all the supplementary. Resulting term space has three dimensions because our instance uses three keywords, making it possible for us to imagine it. It is simple to see that this space could have any number of dimensions which depends on how many keywords we chose to use. Suppose we were to go reverse in the course of the orders and also trace occurrences of sausage, muffin, and bagel, than there would be a six dimensional term space with six dimensional document vectors.

Applying this method to a true document collected works, it return results in a term space with several number of aspects. The file in our set is a vector with various components as there are content words. Documents in such a space that have numerous words in general will have vectors that are close to each one, despite the fact that documents with little shared words will have vectors that are extremely away from each other.

Latent semantic indexing works by analyzing this large and multidimensional space down into a lesser number of dimensions. Keywords that are semantically alike will get compressed collectively and will no longer be completely distinct. The smudging of boundaries is what allows latent semantic indexing to go ahead of straight keyword matching.

## IV. Singular Value Decomposition

Singular Value Decomposition (SVD) is an orthogonal decomposition and is used to decrease the number of dimensions used to signify the documents. For example: Imagine we keep tropical fish, and are proud of our prize aquarium - so proud that we want to give in an image of it to a magazine for the benefit. We desire a good angle from where photo can be taken so that we can get the greatest possible image. We make sure that as countless of the fish as probable are noticeable in our photo, without being unseen by other fish in the foreground. We also don't want the fish all group together in a cluster, but slightly shot from an angle that demonstrate them nicely scattered in the water. Since our tank is transparent on all sides, we can take a variety of photographs from the top of, beneath, and from all in the region of the aquarium, and select the greatest one. In arithmetic terms, we are looking for the best possible mapping of points from a 3-space (the fish) onto a plane (the screen in our camera). 'Optimal' means 'aesthetically pleasing'. Now imagine that our goal is to preserve the relative space between the fish as greatly as feasible, so that fish on the odd sides of the tank do not get superimposed in the snap to appear like they are exactly next to each other. We do exactly what the SVD algorithm does with a much higher-dimensional space for richness.

The SVD algorithm conserves as much data as feasible about the relative space between the documents, whilst collapsing them downwards into a much less significant set of dimensions. Data is mislaid and content words are placed over one another in this fall down. Data loss sounds like a terrible thing, but here it is a blessing. We are losing noise from the original term document matrix, enlightening similarities that were latent in the document collected works. Related things become more interrelated, while dissimilar things remain different. This reductive mapping grants latent semantic indexing its seemingly intelligent behaviour of being able to correlate semantically related expressions.

## V. Proposed Algorithm

This paper has presented an approach combining clustering algorithm with latent semantic indexing (LSI) to retrieve the information. The clustering algorithm that we used is based on the k-means clustering algorithm. The standard k-mean algorithm seems to be a greedy algorithm. We therefore proposed a new 2-step k-means algorithm aiming to get better the standard algorithm.

The main benefit of our technique is that it forces the centroid vector in the direction of the extremities and consequently gets a completely dissimilar starting point as compared to the standard k-means method. This also makes the method less greedy than the standard one, which in itself is a development.

Despite the improvements, there are still some issues that we need to address. First, although the starting point is completely different from the one with the standard k-mean method, the global step is similar to the standard one, and thus the effect on the last result is not big. Second, if a cluster does not get any documents, it will remain empty when the clustering process is finished, which is not good. However, if this happens, it is straightforward to correct it by letting a cluster to keep its old centroid vector. A manner to avoid a cluster being empty is to limit the number of documents to be clustered in each iteration and set best possible value for the marginal value in the local mode. This determines how far from each other the centroids should be. Another side effect of retaining the old centroid vector is to make the 2-step method less sensitive to the initial location of the initial centroid vectors. We consider that our method can be used to retrieve appropriate documents from a document collected works.

## Conclusion

The proposed algorithm helps in making of clusters which in further improves the performance of the system. The indexing performed by latent semantic indexing technique enhances the search results and provide the most appropriate documents as is needed by the user. LSI along with clustering algorithm makes an efficient effort in bringing out the meaningful search.

## References

1. Ahmad, K., Vrusias, B. and Oliveira, P.C.F. (2003), "*Summary Evaluation and Text Categorization*", Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval, Toronto, Canada, pp. 443-444, 2003.
2. Hynek, J. and Jezek, K. (2003), *"Practical Approach to Automatic Text Summarization"*, Proceedings of the ELPUB '03 conference, Guimaraes, Portugal, pp. 378-388, 2003.
3. Information retrieval group, <*http://ir.dcs.gla.ac.uk/resources/test_collections/*>, accessed on 15 July 2014.
4. Mirzal, A. (2013), *"Similarity based matrix completion algorithm"*, International Conference on Control System, Computing and Engineering, Dec 2013.
5. Radev, R., Teufel, S., Saggion, H., Lam, W., Blitzer, J., Qi, H., Celebi, A., Liu, D. and Drabek, E. (2003), "*Evaluation Challenges in Large-scale Document Summarization*", Proceeding of the 41th annual meeting of the Association for Computational Linguistics, Sapporo, Japan, pp. 375-382, 2003.
6. Valle-Lisboa, J.C. and Mizraji, E. (2007), *"The uncovering of hidden structures by Latent Semantic Analysis"*, Information Sciences, 177(19), pp4122-4147, 2007.