Finding the Optimal Rank for LSI Models

Sudarsun Santhiappan, *IEEE Member*, Venkatesh Prabhu Gopalan

Checktronix India Pvt Ltd
Kilpauk, Chennai 600010, India
sudarsun@burning-glass.com
vprabhu@burning-glass.com

***Abstract*— Latent Semantic Indexing is a powerful linear algebraic method for dimension reduction. It is also very useful in solving synonymy problem of textual corpora. A corpora of several documents representing as a bag of features is represented as a Term-Document matrix (TDM), where a Term represents a feature. A TDM can also be a visualization of an experiment repeated several times on an unknown system, where a term of the TDM represents an unknown variable of the system and a document of the TDM represents experiment iteration. Using LSI, a large hyper-space of a corpora or a system could be decomposed into three smaller matrices (Left Singular matrix 'U', Right Singular Matrix 'V', Diagonal matrix of Singular values 'S') as a function of rank 'K', a scalar value. The rank is expected to be optimally smaller, with which the hyperspace could be represented in a sub-space without much of data loss. The choice of Rank 'K' is critical because if the value is chosen to be smaller than optimal, the derived subspace representation is rendered useless as the data loss could become high. We propose a method to mathematically derive the optimal rank, which ensures the best subspace representation of a large hyper-space TDM in reduced dimension. We prove the efficiency of our method by comparing the accuracy values of synonymy measurements made on reduced dimension subspaces that are cut at different 'K' values.**

***Keywords*— Accuracy Measurement, Diagonal Matrix, Dimension Reduction, Hyperspace, LSI, Optimal Rank, Singular Matrix, Singular values, Sub-space, Synonymy, Term Document Matrix**

I. Introduction

Latent Semantic Indexing (LSI) model is the application of Single Value Decomposition (SVD) on a hyperspace called Term Document Matrix (TDM) constructed from a corpus, which is represented using a fixed vocabulary of terms. LSI is based on the assumption that there is always an underlying 'hidden' semantic structure in the pattern of word-usage across documents, rather than just surface level choice of words. LSI attempts to identify this hidden semantic structure through statistical techniques and uses it to represent and retrieve information. This is done by modelling the association (co-occurrence patterns) that exist amongst terms and documents. LSI transforms the term-document vector space (hyperspace) into a more compact latent semantic space. Each dimension in the reduced space corresponds to an 'artificial concept'. These concepts loosely correspond to a set of terms. It is believed that in the vector space of reduced dimensionality, the words referring to related concepts, i.e., words that co-occur, are collapsed into the same dimension. Latent semantic space is thus able to capture similarities that go beyond term similarity. In the latent semantic space, a query and a document can have high similarity even if the document does not contain a query term, provided the terms are semantically related.

In this paper, we discuss about the importance of an optimal rank while building LSI models, which directly controls the accuracy of Information Retrieval capacity of the LSI model. We also discuss the issues in estimating the optimal rank by the traditional trial-and-error method. We present our systematic approach to estimation of optimal rank by analysing the singular values of a LSI model.

II. Related Work

Choosing an optimal dimensionality reduction parameter K, the rank of the SVD problem, for every document collection remains empirical. Traditionally, the optimal K has been chosen by running several sets of queries with known relevant document sets for different values of K. The K that results in the best retrieval performance is chosen as the optimal K for the chosen document collection. The optimal K value falls typically in the range of 100-300 dimensions. This has been an important topic of research for several years.

Berry [2, 3] describes the SVD process and interprets the resulting matrices in a geometric context. They show that the SVD, truncated to k dimensions, gives the optimal rank approximation to the original matrix. Wiemer-Hastings [4] shows that the power of LSI comes primarily from the SVD algorithm. Other researchers have proposed theoretical approaches to understanding LSI. Zha and Simon describe LSI in terms of a subspace model and propose a statistical test for choosing the optimal number of dimensions for a given collection [5]. Story discusses LSI's relationship to statistical regression and Bayesian methods [6]. Ding constructs a statistical model for LSI using the cosine similarity measure, showing that the term similarity and document similarity matrices are formed during the maximum likelihood estimation, and LSI is the optimal solution to this model [8]. Ding and He show the unsupervised dimension reduction is closely related to unsupervised learning, and use the top SVD dimensions to identify good centroid starting points for a K-means clustering algorithm [7]. Although other researchers have explored the SVD algorithm to provide an understanding of SVD-based information retrieval systems, to our knowledge Schutze was the first to study the values produced by LSI [9]. Kontostathis et al later expanded upon this work, showing that the SVD exploits higher order term co-occurrence in a collection, and showing the correlation between the values produced in the term-term matrix and the performance of LSI [10]. They further extended these results to determine the most critical values in an LSI system [11]. Sudarsun et al [17] described the trial and error method for optimal rank estimation. In the following sections, we show that the term relationship information can be found within the first few dimensions of the SVD and explained how to find that optimal dimension.

III. BACKGROUND

Latent Semantic Indexing on a text corpus represented by a Term-Document Matrix, A_{mn} is achieved by performing Singular Value Decomposition (SVD) on A, which gets decomposed into three matrices— a term-concept vector matrix, U_{ik}, a singular values matrix, S_{kk}, and a concept-document vector matrix, V_{kj}, with the parameter 'k', the rank of the SVD decomposition. Typically, the value of 'k' is far lesser in comparison with the 'm' and 'n', the dimensions of the TDM. The decomposed matrices comply with the following criteria:

- $A = USV^T$
- $U^TU = V^TV = I_K$
- $UU^T = I_M$
- $VV^T = I_N$
- $S_{11} > S_{22} > S_{33} > ... > S_{KK}$

LSI captures the co-occurrence patterns of the keywords and documents that were used to build the hyperspace TDM. LSI estimates the co-occurrence of features based on the frequency and association in the hyperspace.

A. Choice of Rank

The input matrix 'A_{mn}' could be reconstructed from the decomposed matrices by computing the product of U, S & V. The purpose of SVD is to reduce the dimension of the original TDM to a sub-space controlled by the parameter 'K'. LSI attempts to find the best sub-space representation of the input matrix 'A', in lesser dimension such that all the important information about 'A' are preserved and the statistically useless information are left behind. The choice of 'K' is critical because if the value of 'K' is too low, the decomposition may end up under-representing the hyperspace TDM. Alternately, if the choice of 'K' is too high, the decomposed sub-space may over-represent the hyperspace by adding in noise components.

B. Rank Vs Precision

If we consider plotting the change in Precision values while evaluating an LSI model against the change in Rank, K_i, it becomes clear that the precision values saturate after a particular rank $K_{optimal}$, followed by a slow decay. From Fig. 1, it is apparent that the precision saturates at rank (K=100), which we would call the optimal rank, $K_{optimal}$. To the left of $K_{optimal}$, the precision is low, because of under representation of prominent data from the hyperspace TDM. To the right of $K_{optimal}$, the precision appears to be decaying slowly because of the over representation of the sub-space with added in noise components.

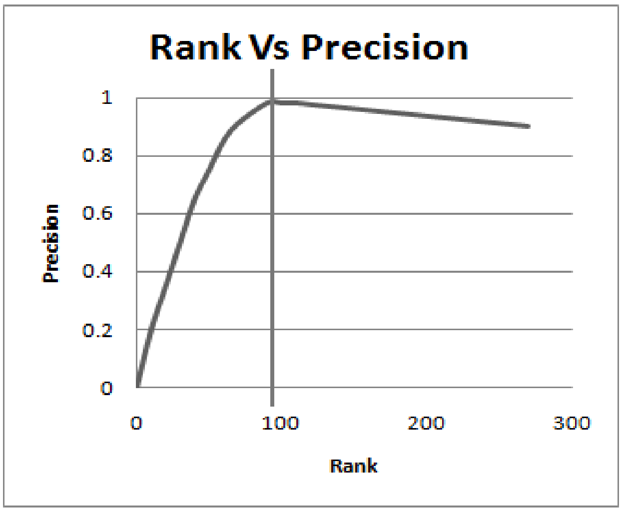

Fig. 1 Precision Vs Increasing Rank

IV. PROPOSED METHOD

We have attempted to identify the optimal K value by analysing the S matrix of the decomposed hyperspace TDM. Our consideration is to find the optimal value of k by using the singular values, which is sorted in the descending order during the decomposition process. When we plotted the singular values, we were able to witness the relationship between the slope of the S-value plot and the slope of the precision value plot.

A. Singular Values

When a scatter plot of the singular values 'S_i' against the rank 'K_i' is made, it becomes apparent that the values saturate after a particular 'K' value, which we call as '$K_{critical}$'.

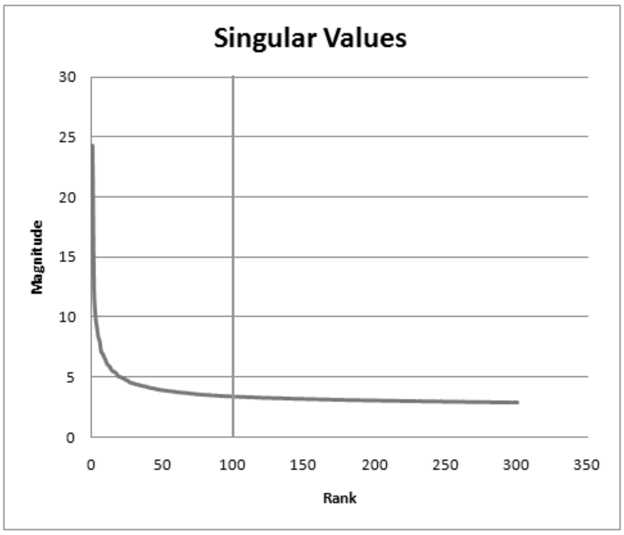

Fig.2 Singular values Vs Rank

$K_{critical}$ is defined as a chosen K value where the slope of the singular value plot (alternately, dS/dK) becomes less than a pre-determined threshold value, which is typically in the range of 1E-3. It is interesting to note that the $K_{critical}$ estimated using the singular value plot and the $K_{optimal}$ estimated using the precision plot is in the likelihood. So, it should be possible to estimate $K_{optimal}$ from just the slope information of the Singular values of a LSI model.

B. Experiment

LSI model could be used to identify synonymous keywords and documents given an input query, which could be a keyword or a document. We chose to test the model using the query keywords and validated the precision of the synonymous keywords returned as results from the input query. We considered testing the LSI model built using technical documents from several industries. We carefully chose evenly distributed keywords from every industrial domain such that the query space is uniformly populated. A evaluation set is created using the chosen uniformly distributed keyword queries along with the probable synonymous keywords handpicked from the chosen technical document corpus. The chosen keywords are queried against the model and the results generated by the model are validated against our evaluation set to compute the precision score for every query. The plot of precision against rank would appear like Fig.1.

The steps are listed below—

a) Select a corpus of several thousand technical text documents that are uniformly distributed across several industrial domains.
b) Select several keywords from each industrial domain that could represent the breadth of the industry
c) Build an LSI model using the corpus
d) Normalize the singular values
e) Plot the singular values of the LSI model
    a. Setup a threshold value (typically 1E-3)
    b. Find the cut in point, where the slope of the magnitude is less than the set threshold
    c. If the cut in point is too low or too high than expected, the threshold may have to be tuned.
f) Cut the LSI model at different K values
g) For each K value
    a. Run the query set against the model
    b. Evaluate the results generated by the model
    c. Compute the precision on a industry basis
    d. Compute the average precision
h) Plot the average precision against every K value
i) Compare the plots and conclude.

C. Algorithm

The algorithm for the estimation of optimal K from the singular values is based on the slope 'dS/dK', where dS/dK is the differentiation of scalar value 'S', with respect to Rank 'K'. The threshold value of 1E-3 is empirically found for our chosen corpus. The threshold could be tuned, if needed, for suiting a different corpus.

```
Algorithm 1: S Values Normalization
Input: Skk

       Let Si:= Diagonal elements of Skk
       Let Total:= Sum of all elements in Si
       Let Sum:= 0
       Let P:= -1
       For i in 1, K
           Let Sum:= Sum+S[i]
           If Sum/Total > 0.5
               P:= i
               Break
           Endif
       Endfor
       If P eq -1
           Report Error
       Endif

       Let Total:=0
       For i in P, K
           Let Total:= Total+S[i]
       Endfor
       For i in P, K
           Let S[i]:=S[i]/Total
       Endfor

       Remove elements 1,P-1 from S
       Return S,P
```

```
Algorithm 2: Find Critical K
Input: S, P, Threshold

       For i in P+1, K
           If S[i]-S[i-1] < Threshold
               Return i
           Endif
       Endfor
```

While normalizing the Singular values, we chose to eliminate the first few elements because of their high magnitude. When high magnitude values are used in the normalization, the average value gets skewed toward the high magnitude side. To avoid this problem, we chose to eliminate the elements that contribute to just about 50% of the total magnitude of the estimated subspace representation. Algorithm 1 describes the procedure of eliminating the high magnitude components of S matrix. It would be interesting to see that 50% of the magnitude is covered roughly within the first 20 odd singular values.

The normalized and truncated S values are used to measure the first-order differential to find the slope. When the slope becomes smaller than the set threshold for a particular 'k' value, it is declared that the current 'k' value becomes the critical 'K'. Algorithm 2 describes the procedure in finding the 'K'.

D. Illustration

We performed an experiment as per the procedure described in section IV (B) using a corpus of 100,000 technical text documents. We have evaluated the model for ranks ranging from 50 to 500 in steps of 50 and plotted the precision values. Similarly, we did a plot of the Singular values of the model. The plots are shown in Fig.3. It is estimated that the model is performing best at rank K=300, which is confirmed by the minimum slope of the singular values at K=300.

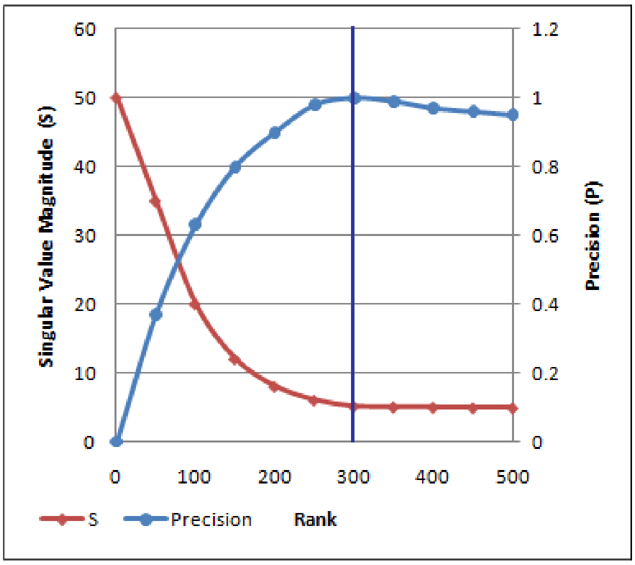

Fig.3 Singular values (S) and Precision (P) plotted against Rank (K) for a LSI model built with a technical text corpus of 100,000 documents

V. Conclusion

There are no standard methods for choosing the optimal rank of the reduced dimension while building SVD or LSI models. Usually the choice is made through repeated trial and error. We have presented a method to estimate the optimal rank using a simple and systematic procedure.

From the experiments, we made an important discovery about the co-occurrence and correlation estimation of the LSI model, which is; the singular values of the LSI model can also be visualized as the topic proportions in an aspect model. While doing so, the optimal rank could be understood as the number of topics that are distributed in the hyperspace. We could classify these topics into three categories, viz. seeding or strong topics, secondary or supportive topics and the junk or noise topics. The proportion of these topics is the main reason for the precision or accuracy of the model. Absence of primary topics will directly affect the accuracy, and the inclusion of the unnecessary junk topics degrades the performance of the model, which was made evident in Fig.1. The optimal rank is the cut-off point which ensures the inclusion of only the useful topics and thus achieving higher accuracy.

Acknowledgment

The authors wish to acknowledge their colleagues for their expert advices and assistance in executing the experiments.

References

[1] M. W. Berry, “Large-scale sparse singular value computations”, *The International Journal of Supercomputer Applications*, 6(1):13–49, Spring 1992.

[2] M. W. Berry, Z. Drmac, and E. R. Jessup, “Matrices, vector spaces, and information retrieval”, *SIAM Review*, 41(2):335–362, 1999.

[3] M.W. Berry, S. T. Dumais, and G.W. O’Brien, “Using linear algebra for intelligent information retrieval”, *SIAM Review*, 37(4):575–595, 1995

[4] P. Wiemer-Hastings, “How latent is latent semantic analysis?” *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pp 932–937, 1999.

[5] H. Zha and H. Simon, “A subspace-based model for Latent Semantic Indexing in information retrieval”, *In Proceedings of the Thirteenth Symposium on the Interface*, pp 315–320, 1998.

[6] R. E. Story, “An explanation of the effectiveness of Latent Semantic Indexing by means of a Bayesian regression model”, Information Processing and Management, 32(3):329–344,1996.

[7] C. Ding and X. He, “K-means clustering via principal component analysis”. In ICML ’04: *Proceedings of the twentyfirst international conference on Machine learning*, pp 29, 2004. ACM Press.

[8] C. H. Q. Ding, “A similarity-based probability model for latent semantic indexing”, In *Proceedings of the Twenty-second Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pp 59–65, 1999.

[9] H. Sch¨utze, “Dimensions of meaning” In *Proceedings of Supercomputing*, pp 787–796, 1992.

[10] A. Kontostathis and W. M. Pottenger, “A framework for understanding Latent Semantic Indexing (LSI) performance”, Information Processing and Management, 42(1):56–73, 2006.

[11] A. Kontostathis, W. M. Pottenger, and B. D. Davison, “Identification of critical values in latent semantic indexing” In *T. Lin, S. Ohsuga, C. Liau, X. Hu, and S. Tsumoto, editors, Foundations of Data Mining and Knowledge Discovery*, pp 333–346. Spring-Verlag, 2005.

[12] E. R. Jessup and J. H. Martin, “Taking a new look at the latent semantic analysis approach to information retrieval”, In Computational Information Retrieval, pp 121–144, 2001.

[13] T. A. Letsche and M. W. Berry, “Large-scale information retrieval with latent semantic indexing”, Information Sciences, 100(1-4):105–137, 1997.

[14] Kontostathis, “Essential Dimensions of Latent Semantic Indexing (LSI)”, *Proceedings of the 40th Hawaii International Conference on System Sciences*, 2007

[15] AswaniKumar Ch. and Srinivas S. (2006), “On the effect of rank approximation on information retrieval”, *Proc. Int. Conf. Systemics Cybernetics and Informatics*, India, pp. 876–880.

[16] Balinski J. and Danilowicz C. (2005), “Ranking method based on inter document distances”, Information Process Management., Vol. 41, No. 4, pp. 759–775.

[17] Sudarsun Santhiappan, Venkatesh Prabhu Gopalan, and Sathish Kumar Veeraswamy, “Role of Weighting on TDM in Improving Performance of LSA on Text Data", *Proceedings of IEEE INDICON* 2006.

About the Authors

Sudarsun S (M’ 2002) is the Director – R & D at Checktronix India Pvt Ltd, Chennai. He holds an M.Tech in Computer Science Engg from IIT Madras. He received a BE degree in Electronics and Instrumentation Engg from Madras University with a Gold Medal. His research interests include SNLP, IR, IE, Search Engineering, Text Mining, Machine Learning and Distributed Computing

Venkatesh Prabhu G is the Applied Research Scientist at Checktronix India Pvt Ltd, Chennai. He holds an ME in Computer Science from Anna University, Chennai. He received his BE in Computer Science from MKU, Madurai. His research interests include AI, Data / Text / Web Mining, Pattern Classification, Knowledge Based Neural Networks, Machine Learning, and Soft Computing.