SHORT PAPER

International Journal of Recent Trends in Engineering ,Vol 1, No. 3, May 2009

# Evaluation of SVD and NMF Methods for Latent Semantic Analysis

Rakesh Peter, Shivapratap G, Divya G, Soman KP
Amrita University/CEN, Coimbatore, India
Email: {p_rakesh, g_shivapratap, g_divya, kp_soman}@ettimadai.amrita.edu

***Abstract* -Different mathematical techniques are being developed to reduce the dimensionality of data within large datasets, for robust retrieval of required information. Latent Semantic Analysis (LSA), a modified low rank approximation form of Vector Space Model, can be used for detecting underlying semantic relationships within text corpora. LSA performs a low-rank approximation on term-document matrix, which is generated by transforming textual data into a vector representation, thereby bringing out the semantic connectedness present among the documents of the corpus. Singular Value Decomposition (SVD) is the traditional approximation method used for LSA, wherein lower dimensional components from the decomposition are truncated. On truncation, the linguistic noise present in the vector representation is removed, and the semantic connectedness is made visible. One of the pitfalls of using SVD is that the truncated matrix will have negative components, which is not natural for interpreting the textual representation. Nonnegative Matrix Factorization (NMF) addresses this issue by generating non-negative parts-based representation as the low rank approximation for performing LSA. The paper provides an in-depth overview of how both methods are being used for the purpose of Information Retrieval. Performance evaluation of the methods has been performed using standard test datasets.**

## I. INTRODUCTION

Since information explosion has been taking hold over the Internet, handling of large datasets has become an active research area. A huge amount of data is being generated everyday from a variety of sources, and techniques for structuring the same data are being explored, so that they can be stored and retrieved in a convenient fashion. Information Retrieval has come a long way, evolving itself through different classic models such as Boolean model, Probabilistic Model and finally, Vector Space Model (VSM) [1].

VSM relies on the fact that the meaning of the document can be derived from the document's constituent terms. VSM deals with transforming the text corpus into a term-document matrix,

$$\boldsymbol{A}_{mxn} = \begin{bmatrix} a_{11} & .. & a_{1n} \\ .. & .. & .. \\ a_{m1} & .. & a_{mn} \end{bmatrix}$$

where $m$ is the number of unique terms and $n$ is the number of individual documents in the corpus. $a_{ij}$ corresponds to the frequency of each term element from the respective document. The query to be retrieved is represented as a vector,

$$\boldsymbol{q}_{1xm} = [q_1 \quad ... \quad q_m]$$

where $q_i$ is the frequency of the term element, corresponding to its position in the column of the term-document matrix. In order to obtain a similarity measure of the query vector with each document in the corpus, the cosine measure between $\boldsymbol{q}$ and each column vector of $\boldsymbol{A}$ is computed, the reason being the most relevant documents are expected to be represented as vectors closest to the query.

$$sim(\boldsymbol{q}, a_j) = \frac{\boldsymbol{q}\, a_j}{\sqrt{\|\boldsymbol{q}\|\|a_j\|}}$$

The issue with VSM is that the term-document matrix is usually very sparse and also high dimensional, making it susceptible to linguistic noise and also, it is difficult to capture the semantic information.

## II. DIMENSIONALITY REDUCTION METHODS

Latent Semantic Analysis (LSA) [2] comes as an extension to VSM to overcome these issues, wherein a low-rank approximation is performed on the term-document matrix by means of dimensionality reduction methods. Rather than simply trying to match the words of the queries, it tries to retrieve results on the basis of the conceptual content of the documents. In this paper, we evaluate the performance of two dimensionality reduction methods used in LSA – SVD[3] and NMF[4].

## III. IMPLEMENTATION

As a part of evaluating the dimensionality reduction methods mentioned above, we have done an implementation of a Java-based Information Retrieval system – *iDrishti*, which

performs the evaluation of test datasets using standard measures such recall and precision.

The *iDrishti* system makes use of Lucene [8], a Java-based robust full-text indexing and search library, for generating the term-document matrix from the text corpus. Lucene can be configured to internally store the term-frequency vector for each document, which can be retrieved from the generated Lucene index [9]. Lucene has got inbuilt methods for performing stop-word filtering and stemming, which are helpful to remove the redundancies in the term-document matrix. Lucene also supports plug-ins for various file formats such PDF, DOC, RTF, HTML, XML, etc. using which a variety of corpus can easily be indexed.

The *iDrishti* system retrieves the term-frequency vectors for each document from Lucene index and the term-frequency matrix is generated. Local and global term weighting schemes are then applied to the matrix, in order to improve the scoring for the important terms. The system supports various term weighting schemes [10] such as Term Frequency (TF), Binary and Logarithmic for local weighting and Inverse Document Frequency (IDF), Probabilistic Inverse and Entropy for global weighting. The column vectors of the matrix corresponding to each document are then normalized using the Frobenius vector norm. The query for retrieval is processed and the query vector is also generated similarly.

The *iDrishti* system provides a plug-in interface for performing dimensionality reduction, so that the user can program and evaluate algorithms by overriding the methods of the interface, thereby abstracting other procedures involved in the evaluation process. For carrying out the SVD operation, we make use of Matrix Toolkit for Java (MTJ) [11], which has a built-in "SingularvalueComputer". The Multiplicative Update [4] algorithm for performing NMF was implemented as afore mentioned plug-in to the system.

$W = \text{abs}(\text{randn}(m,k))$;
$H = \text{abs}(\text{randn}(k,n))$;
for i = 1 : *maxiter*
$H = H.*(W^T A)./(W^T H + 10^{-9})$
$W = W.*(AH^T)./(WHH^T + 10^{-9})$
end

Figure 1. Multiplicative Update algorithm for NMF

Once the low rank approximation is carried out on the term-document matrix using the dimensionality reduction methods, the query vector can be projected into the reduced dimensional space, and the cosine similarity between column vectors and the query vector can be evaluated.

## IV. Performance Evaluation

The performance evaluation in the system is being carried out by using standard measures of recall and precision [12]. For each query, 11-point interpolated average precision [13] is computed.

The test dataset used for this evaluation is the MEDLINE [14] dataset, which contains 1033 documents and 30 queries. The MEDLINE dataset was chosen since (1) it provides document relevance listing for each query, (2) it has been used in previous related evaluations [13] [15]. After stop word elimination and stemming the corpus, 7014 terms were obtained. In the term document matrix, logarithmic local weighting and inverse document frequency is used for global weighting.

In the following figures, we present the results of our evaluation. In Figures 2 and 3, we provide the average precision with varying recall levels, for different *k* values of SVD and NMF respectively, along with the VSM baseline for comparison.

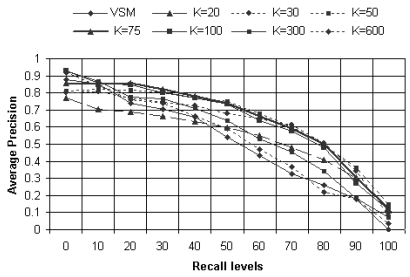


Figure 2. Average Precision at varying recall levels and *k* values for SVD

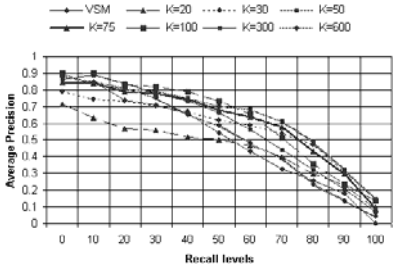


Figure 3. Average Precision at varying recall levels and *k* values for NMF

The interpolated precision values at the 11 recall levels returned from both methods suggest similar performance figures. Varying the value of *k* between 50 and 100 gives the best results for both methods. This is made more visible in Figure 4, the average precision vs. *k* values, where in we can clearly see that both methods have a similar curve. SVD has got the highest value of average precision (p = 0.644) on the whole run, at *k* value 75.

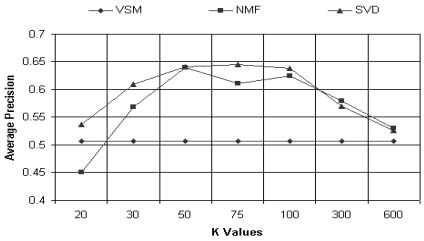


Figure 4. Comparison of Interpolated Average Precision for different *k* values for VSM, SVD and NMF

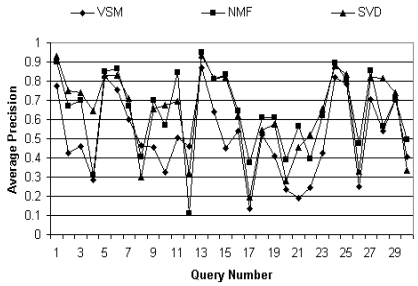


Figure 5. Comparison of Interpolated Average Precision of MEDLINE queries for VSM, SVD and NMF ($k$ = 50).

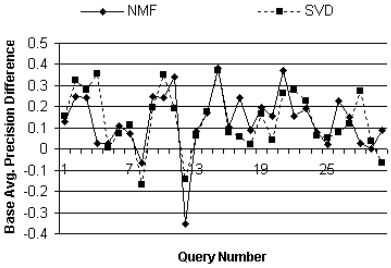


Figure 6. Comparison of Average Precision of queries for SVD and NMF ($k$ = 50) with VSM as baseline.

In Figures 5 and 6, we have presented the results on evaluation of interpolated average precision for each of the 30 queries in the MEDLINE database. Figure 5 represents the variation of precision for each query with respect to VSM, SVD and NMF at $k$ value of 50, where we have recorded the maximum average precision in Figure 4.

We have also presented the Base Average Precision Difference metric in Figure 6, with which the performance difference with respect to precision of NMF and SVD can be clearly illustrated. Base Average Precision Difference is computed by taking the average precision values of VSM as the baseline, and calculating the difference for NMF and SVD from the baseline. The visual analysis of Figure 6 clearly shows how each query has performed for a method with respect to the other. Amongst the 30 queries, NMF has performed better than SVD in 17 queries.

## V. CONCLUSION

A study on the performance of these dimensionality reduction methods on the MEDLINE dataset was carried out and the results are presented. Both the methods are performing well above the conventional Vector Space Model, although significant performance difference between each other could not be detected for this particular dataset, as far as the parameters of precision and recall are considered. NMF has the advantage over SVD that it produces a natural "additive parts-based" representation of data, owing to its non-negativity, which can be helpful in document clustering based on topics.

## REFERENCES

[1] Baeza-Yates, R., Riberio-Neto. *Modern Information Retrieval.* Addison Wesley, 1999.

[2] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman.. Indexing by Latent Semantic Analysis, In *Journal of the American Society of Information Science*, 1990.

[3] Golub, G. H. and Van Loan, C. F. *Matrix Computations, 3rd ed.*, John Hopkins University Press, 1996.

[4] Daniel D. Lee & H. Sebastian Seung. Algorithms for Non-negative Matrix Factorization. In *Proceedings of NIPS'00, volume 13*. MIT Press, 2000.

[5] M. Chu, F. Diele, R. Plemmons, S. Ragni, Optimality, computation, and interpretations of nonnegative matrix factorizations. Available from the web at http://www.wfu.edu/~plemmons, 2004.

[6] P. Paatero, U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. In *Environmetrics 5*, 1994.

[7] F Shahnaz, M Berry, P Pauca, R Plemmons. Document Clustering using Non-negative Matrix Factorization. In *Journal on Information Processing and Management*, 2005.

[8] Lucene: A high performance full text indexing and searching library for Java. Available from the web at http://lucene.apache.org/, 2005.

[9] Grant Ingersoll, Ozgur Yilmazel, Niranjan Balasubramanian, Steve Rowe, Svetlana Smoyenko. Advanced Lucene. *Presented at ApacheCon 2005.* Available from web at http://www.cnlp.org/apach econ2005/AdvancedLucene.pdf.

[10] Salton, G., Buckley C. Term weighting approaches in automatic text retrieval. In *Readings in Informatrion Retrieval*, Morgan Kaufmann Publishers, 1997.

[11] Matrix Toolkit for Java: Linear Algebra package for Java, available from web at http://www.mi.uib.no / ~bjornoh/mtj/.

[12] R. Baeza-Yates, W.B. Frakes, *Information Retrieval: Data structures and algorithms.* Prentice Hall, 1992.

[13] T. G. Kolda and D. P. O'Leary, Latent semantic indexing via a semi-discrete matrix decomposition. In *The Mathematics of Information Coding, Extraction and Distribution, vol. 107 of IMA Volumes in Mathematics and Its Applications*, Springer-Verlag, 1999.

[14] MEDLINE Dataset, available from ftp://ftp.cs.cornell.edu/pub/smart/med/.

[15] P. Husbands, H. Simon, C. Ding. On the use of singular value decomposition for text retrieval. In *Computational Information Retrieval*, 2001.