Using Random Indexing to improve Singular Value Decomposition for Latent Semantic Analysis

Linus Sellberg, Arne Jönsson

Department of Computer and Information Science, Santa Anna IT Research Institute AB
Linköping University, SE-581 83, LINKÖPING, SWEDEN
x07linse@ida.liu.se, arnjo@ida.liu.se

Abstract

We present results from using Random Indexing for Latent Semantic Analysis to handle Singular Value Decomposition tractability issues. We compare Latent Semantic Analysis, Random Indexing and Latent Semantic Analysis on Random Indexing reduced matrices. In this study we use a corpus comprising 1003 documents from the MEDLINE-corpus. Our results show that Latent Semantic Analysis on Random Indexing reduced matrices provide better results on Precision and Recall than Random Indexing only. Furthermore, computation time for Singular Value Decomposition on a Random Indexing reduced matrix is almost halved compared to Latent Semantic Analysis.

1. Introduction

We are developing an FAQ-system called ACC (Automatic Contact Center) were clients requests for guidance, help or contact information are handled without human intervention, c.f. (Åberg, 2002). We use a two-step approach for our development of ACC. First, we collect human-human call center conversations and build a database that can be searched in various ways. This database will be used to suggest responses to the call center agent continuously. The database will be extended when new requests arise. Once the database is large enough we will use it to automatically answer requests from the clients.

In this paper we present improvements to techniques for identifying "similar" requests from a collection of FAQ-items. Such techniques differ from techniques utilised in traditional QA systems, such as FALCON (Harabagiu et al., 2000), where a collection of documents are used to retrieve an answer. Typically QA-systems first utilise information retrieval techniques to select documents corresponding to the question type and keywords from the question. Then the answer is extracted from this subset of documents, for instance, through pattern matching.

FAQ-systems are instead based on previously recorded FAQ-items and utilise various techniques to identify the FAQ-item(s) that best resembles the current question and present a matching answer. For instance, the FAQFinder system (Mlynarczyk and Lytinen, 2005), which uses existing FAQ knowledge bases to retrieve answers to natural language questions. FAQFinder utilises a mixture of semantic and statistical methods for determining question similarities. Another technique is to use a frequency-based analysis from an ordinary FAQ with given/static questions and answers (Ng'Ambi, 2002). Linguistic based automatic FAQ systems often starts with finding the question word, keywords, keyword heuristics, named entity recognition, and so forth (Moldovan et al., 1999). Knowledge based systems can also utilise dialogue to ask the user to guide the search (Kurohashi and Higasa, 2000). Their knowledge base has a dictionary-like structure based on domain concepts.

One issue for automatic help-desk systems is that we have often many-to-many mappings between requests and responses. A question is stated in many ways and, as humans answer the requests, the response to a question can be stated in many ways. The propositional content can also vary, although operators re-use sentences, at least in e-mail help desk-systems (Zukerman and Marom, 2006).

Help-desk e-mail conversations are further characterised by: (1) having many requests raising multiple issues, (2) having high language variability and (3) with many answers utilising non-technical terms not matching technical terms in the requests (Marom and Zukerman, 2007). In this paper we will not consider (1), i.e. we will not present answers comprising multiple issues.

2. Vector space models

To handle (2) and (3) above, it is natural to investigate techniques that in various ways reduce the linguistic variability and better capture semantically related concepts. One prominent such technique is vector space models (Eldén, 2007). The basic idea is to formulate the problem in a matrix terminology, usually by constructing a matrix using text units as columns and letting the words in all text units each have a row in the matrix. Then a certain entry in the matrix is nonzero iff the word corresponding to the row exists in the text unit represented by the column. There are also other ways of representing text in a matrix but those will not be discussed further here.

2.1. Latent Semantic Analysis

Latent Semantic Analysis (LSA) (Landauer and Dumais, 1997) is one of the most well-known vector space models and has, among other things, been used on a limited help-desk data set with promising results (Caron, 2000). When using questions in an FAQ to compute a vector space model, LSA uses Singular Value Decomposition(SVD) to find a reduced vector space that fits the original as well as possible using a lower ranked matrix.

The SVD factorizes a matrix A into $A = USV^T$ where U is the new orthonormal basis for A, S is a diagonal matrix denoting how prevalent each column is in the basis while V^T is the coordinates of the original documents using the new basis.

It turns out that results are often improved when dimensions that correspond to the lower values of S are removed from the basis. This removes the orthogonality restraint between the different word vectors, and words that appear in similar contexts will have a scalar product $\neq 0$. The product of the l biggest values of S, S_l and the corresponding columns and rows of U_l and V_l^T makes the closest possible approximation of A of rank l (Eldén, 2007).

One way to look at this is that the dimensionality reduction is reducing the amount of noise in the data. An additional bonus of reducing the rank of the SVD is that the result is faster to compare with queries, due to fewer elements to multiply. It is also less computationally demanding to compute a partial SVD consisting of only the l biggest singular values and the corresponding singular vectors than it takes to compute a complete SVD factorization.

Answers to questions are retrieved based on how similar they are to previous questions in the vector space, by changing the basis of the query to U and then compare likeness with the coordinates in V^T, often based on the cosine between the two (Caron, 2000).

For larger data sets SVD can be computationally demanding. One way to improve scalability of LSA is by using Generalised Hebbian Learning (GHA) for SVD (Gorrell, 2006). This allows for incremental SVD and handles very large data sets. When applied to a situation with continuously added FAQs, the GHA approach may well be preferred. For our current application the frequency of novel requests is small, and simpler methods may suffice. One such possibility is to reduce the dimensionality of the matrix on which SVD is calculated using Random Indexing (Gorrell, 2006; Kanerva et al., 2000).

2.2. Random Indexing

Random Indexing is an incremental vector space model that is computationally less demanding (Karlgren and Sahlgren, 2001). The Random Indexing model reduces dimensionality by, instead of giving each word a whole dimension, it gives them a random vector by a much lesser dimensionality than the total number of words in the text.

Random Indexing differs from the basic vector space model in that it doesn't give each word an orthogonal unit vector. Instead each word is given a vector of length 1 in a random direction. The dimension of this randomized vector will be chosen to be smaller than the amount of words in the document, with the end result that not all words will be orthogonal to each other since the rank of the matrix won't be high enough. This can be formulated as $AT = \tilde{A}$ where A is the original matrix representation of the $d \times w$ word-document matrix as in the basic vector space model, T is the random vectors as a $w \times k$ matrix representing the mapping between each word w_i and the k-dimensional random vectors, $\tilde{A}$ is A projected down into $d \times k$ dimensions. A query is then matched by first multiplying the query vector with T, and then find the column in $\tilde{A}$ that gave the best match.

T is constructed by, for each column in T, each corresponding to a row in A, selecting n different rows. $n/2$ of these are assigned the value $1/\sqrt{(n)}$, and the rest are assigned $-1/\sqrt{(n)}$. This ensures unit length, and that the vectors are distributed evenly in the unit sphere of dimension k (Sahlgren, 2005). An even distribution will ensure that every pair of vectors have a high probability to be orthogonal.

Information is lost during this process (pigeonhole principle, the fact that the rank of the reduced matrix is lower). However, if used on a matrix with very few nonzero elements, the induced error will decrease as the likelihood of a conflict in each document, and between documents, will decrease.

Using Random Indexing on a matrix will introduce a certain error to the results. These errors will be introduced by words that match with other words, i.e. the scalar product between the corresponding vectors will be $\neq 0$. In the matrix this will show either that false positive matches are created for every word that have a nonzero scalar product of any vector in the vector room of the matrix. False negatives can also be created by words that have corresponding vectors that cancel each other out.

Random Indexing has been used and compared to LSA on the TOEFL test giving similar results on the tests, but with a much reduced matrix (Kanerva et al., 2000).

3. Experiment

We will investigate the performance of these three techniques, i.e. (1) standard Latent Semantic Analysis, (2) Random Indexing and (3) Latent Semantic Analysis on a matrix where Random Indexing is used to reduce the dimensionality of the matrix before singular value decomposition. Performance will be measured as calculation time as well as precision and recall.

We have used a subset of the MEDLINE-corpus[1]. The subset comprise 1003 documents and 30 test queries. The corpus and test queries are parsed, stemmed and have stop words removed. Stemming and removal of stop words are performed since the information contained in them doesn't matter for the problem area and these actions will be very likely to increase performance. The results are inserted into a word-document matrix with words representing rows and documents representing columns, resulting in a matrix with 3020 rows and 1063 columns.

MEDLINE is, admittedly, a small corpus, but it is publicly available and sufficient to show if Random Indexing applied prior to computing SVD is more efficient. It should also be sufficient to show what effects the usage of Random Indexing have on precision and recall compared to LSA with SVD on a Random Indexing reduced matrix.

To investigate how Random Indexing affects the computation time of the SVD we make two Random Indexing transformations, one with half the size of the original matrix, i.e. 1500 rows, and one with a quarter of the original size, 750 rows. After this we run SVD on the original matrix and also on both matrices reduced with Random Indexing. These reduced matrices contain a lot more nonzero values than the original matrix, and our aim is to see how this trade off between more non zero values versus fewer rows behave performance wise.

We do this by first performing SVD for matrices with just a few columns for all three alternatives, and then increasing

[1] www.dcs.gla.ac.uk/idom/ir_resources/test_collections

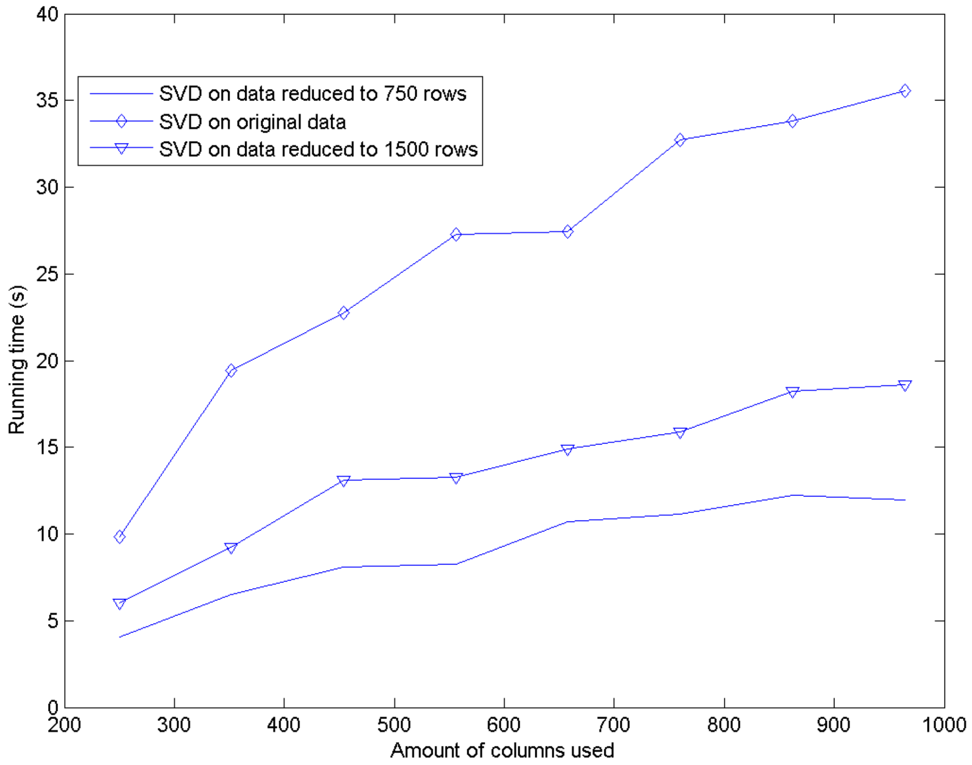

Figure 1: Scaling of SVD and of SVD used on Random Indexing reduced matrices

the amount of columns incrementally while measuring the time it takes to perform the SVD.

We use precision and recall to compare the techniques (Eldén, 2007). Precision measures how exact the retrieval process is and is defined as the number of documents correctly retrieved divided by the total number of documents Recall is a measure of how complete the retrieval process is and it is defined as the number of documents correctly retrieved divided by the total number of correct documents.

To measure precision and recall we run tests with the 30 MEDLINE query entries and then compare the results to a handmade set of correct answers, also publicly available.

We do this for the pure vector model sans LSA, for pure LSA, for pure Random Indexing and lastly for LSA applied on a matrix that have been reduced with Random Indexing.

4. Results

Our results show that SVD on a Random Indexing reduced matrix with 1500 rows is about twice as fast as SVD on our non-reduced matrix. Furthermore, the difference is more or less independent on the number of columns used, see Figure 1. Using the more reduced matrix on 750 rows provides even shorter running time, but the reduction is not as dramatic as the reduction from the original 3000 rows matrix to the 1500 rows matrix.

Precision and Recall for LSA on Random Indexing reduced matrices compared to Random Indexing and LSA show that LSA on Random Indexing reduced matrices provides far better results, for precision above 0,2, than Random Indexing, but slightly worse results than pure LSA, Figure 2. We also see a loss in Precision and Recall for Random Indexing compared to not only our two versions of LSA, but also to the pure vector space model. This seemingly contradict (Karlgren and Sahlgren, 2001) who report similar performance as for LSA (although for a different problem).

5. Summary

In this paper we have presented results from using Random Indexing to reduce the dimensionality of the matrix used to perform Singular Value Decomposition for Latent Semantic Analysis. Our results show that Precision and Recall are far better than using only Random Indexing and that computation time for the Singular Value Decomposition is almost halved compared to non-reduced matrices.

The presented results used a rather small data set, 1003 documents from the MEDLINE corpus, as it is readily available. Consequently our original matrix has few rows. We believe, however, that the gain is most likely to increase significantly if this test would have been conducted on a larger matrix. Here we only used reductions of a half and a quarter of the size, while on a real big problem it would be more likely to be somewhere along the lines of two to three magnitudes of change (Kanerva et al., 2000).

Acknowledgment

This research is financed by Santa Anna IT Research Institute AB.

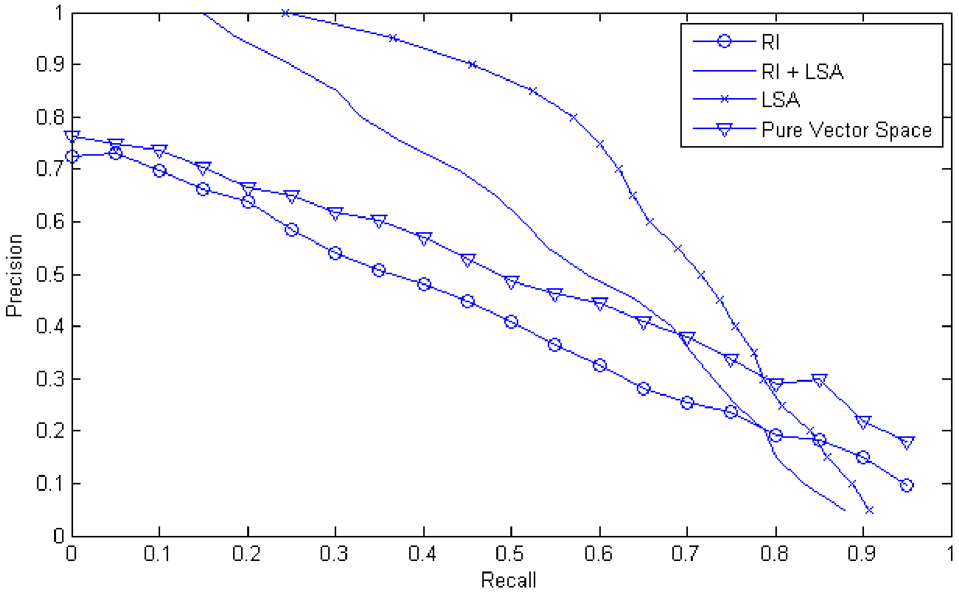

Figure 2: Precision as a function of Recall for LSA, Random Indexing and LSA with Random Indexing reduced matrices

6. References

John Caron. 2000. Applying lsa to online customer support: A trial study. Master's thesis, University of Colorado, Boulder.

Lars Eldén. 2007. *Matrix Methods in Data Mining and Pattern Recognition*. Society for Industrial & Applied Mathematics (SIAM).

Genevieve Gorrell. 2006. *Generalized Hebbian Algorithm for Dimensionality Reduction in Natural Language Processing*. Ph.D. thesis, Linköping University.

S. Harabagiu, D. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Girju, V. Rus, and P. Morarescu. 2000. FALCON: Boosting Knowledge for Answer Engines. In *Proceedings of Text Retrieval Conference (TREC-9)*.

Pentti Kanerva, Jan Kristofersson, and Anders Holst. 2000. Random indexing of text samples for latent semantic analysis. In *Proceedings of the 22nd Annual Conference of the Cognitive Science Society. Erlbaum, 2000.*, page 1036.

Jussi Karlgren and Magnus Sahlgren. 2001. From words to understanding. In Y. Uesaka, P.Kanerva, and H. Asoh, editors, *Foundations of Real-World Intelligence*, chapter 26, pages 294–308. Stanford: CSLI Publications.

Sadao Kurohashi and Wataru Higasa. 2000. Dialogue helpsystem based on flexible matching of user query with natural language knowledge base. In *Proceedings of the 1st SIGdial workshop on Discourse and dialogue*, pages 141–149, Morristown, NJ, USA. Association for Computational Linguistics.

Thomas K. Landauer and Susan T. Dumais. 1997. A solution to Plato's problem: The latent semantic analysis theory of the acquisition, induction, and representation of knowledge. *Psychological Review*, 104:211–240.

Yuval Marom and Ingrid Zukerman. 2007. A predictive approach to help-desk response generation. In Manuela M. Veloso, editor, *Proceedings of IJCAI 2007, Hyderabad, India*, pages 1665–1670.

S. Mlynarczyk and S. Lytinen. 2005. Faqfinder question answering improvements using question/answer matching. In *Proceedings of L&T-2005 - Human Language Technologies as a Challenge for Computer Science and Linguistics*.

D. Moldovan, S. Harabagiu, M. Pasca, R. Mihalcea, R. Goodrum, R. irji, and V. Rus. 1999. Lasso: A tool for surfing the answer net.

D Ng'Ambi. 2002. Pre-empting user questions through anticipation: data mining faq lists. In *Proceedings of the 2002 Annual Research Conference of the South African institute of Computer Scientists and information Technologists on Enablement Through Technology*. ACM International Conference Proceeding Series.

M. Sahlgren. 2005. An Introduction to Random Indexing. *Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*.

Ingrid Zukerman and Yuval Marom. 2006. A comparative study of information-gathering approaches for answering help-desk email inquiries. In *Proceedings of 19th Australian Joint Conference on Artificial Intelligence, Hobart, Australia*.

Johan Åberg. 2002. *Live Help Systems: An Approach to Intelligent Help for Web Information Systems*. Ph.D. thesis, Linköpings universitet, Thesis No 745. `http://www.ida.liu.se/~johab/articles/phd.pdf`.