Clustered Sub-matrix Singular Value Decomposition

Fang Huang
School of Computing
Robert Gordon University
Aberdeen, AB25 1HG, UK
f.huang@rgu.ac.uk

Yorick Wilks
Department of Computer Science
University of Sheffield
Sheffield, S1 4DP, UK
y.wilks@dcs.shef.ac.uk

Abstract

This paper presents an alternative algorithm based on the singular value decomposition (SVD) that creates vector representation for linguistic units with reduced dimensionality. The work was motivated by an application aimed to represent text segments for further processing in a multi-document summarization system. The algorithm tries to compensate for SVD's bias towards dominant-topic documents. Our experiments on measuring document similarities have shown that the algorithm achieves higher average precision with lower number of dimensions than the baseline algorithms - the SVD and the vector space model.

1 Introduction

We present, in this paper, an alternative algorithm called Clustered Sub-matrix Singular Value Decomposition(CSSVD) algorithm, which applied clustering techniques before basis vector calculation in SVD (Golub and Loan, 1996). The work was motivated by an application aimed to provide vector representation for terms and text segments in a document collection. These vector representations were then used for further preprocessing in a multi-document summarization system.

The SVD is an orthogonal decomposition technique closely related to eigenvector decomposition and factor analysis. It is commonly used in information retrieval as well as language analysis applications. In SVD, a real m-by-n matrix A is decomposed into three matrices, $A = U \sum V^T$. $\sum$ is an m-by-n matrix such that the singular value σ_i=$\sum_{ii}$ is the square root of the i^{th} largest eigenvalue of AA^T, and $\sum_{ij} = 0$ for $i \neq j$. Columns of orthogonal matrices U and V define the orthonormal eigenvectors associated with eigenvalues of AA^T and $A^T A$, respectively. Zeroing out all but the k, $k < rank(A)$, largest singular values yields $A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$, which is the closest rank-k matrix to A. Let A be a term-document matrix. Applications such as latent semantic indexing (Deerwester et al., 1990) apply the rank-k approximation A_k to the original matrix A, which corresponds to projecting A onto the k-dimension subspace spanned by $u_1, u_2, ..., u_k$. Because $k \ll m$, in this k-dimension space, minor terms are ignored, so that terms are not independent as they are in the traditional vector space model. This allows semantically related documents to be related to each other even though they may not share terms.

However, SVD tends to wipe out outlier (minority-class) documents as well as minor terms (Ando, 2000). Consequently, topics underlying outlier documents tend to be lost. In applications such as multi-document summarization, a set of related documents are used as the information source. Typically, the documents describe one broad topic from several different view points or sub-topics. It is important for each of the sub-topics underlying the document collection to be represented well.

Based on the above consideration, we propose the CSSVD algorithm with the intention of compensat-

Proceedings of NAACL HLT 2007, Companion Volume, pages 69–72,
Rochester, NY, April 2007. ©2007 Association for Computational Linguistics

ing for SVD's tendency to wipe out minor topics. The basic idea is to group the documents into a set of clusters using clustering algorithms. The SVD is then applied on each of the document clusters. The algorithm thus selects basis vectors by treating equally each of the topics. Our experiments on measuring document similarities have shown that the algorithm achieves higher average precision with lower number of dimensions than the SVD.

2 the Algorithm

The input to the CSSVD algorithm is an m×n term-document matrix A. Documents in matrix A are grouped into a set of document clusters. Here, we adopt single-link algorithm to develop the initial clusters, then use K-means method to refine the clusters. After clustering, columns in matrix A are partitioned and regrouped into a set of sub-matrices A_1,A_2,...,A_q. Each of these matrices represents a document cluster. Assume A_i, $1 \leq i \leq q$, is an $m \times n_i$ matrix, these sub-matrices are ranked in decreasing order of their sizes, i.e., $n_1 \geq n_2 \geq ... \geq n_q$, then $n_1 + n_2 + ... + n_q = n$.

The algorithm computes basis vectors as follows: the first basis vector u_1 is computed from A_1, i.e., the first left singular vector of A_1 is selected. In order to ensure that the basis vectors are orthogonal, singular vectors are actually computed on residual matrices. R_{ij}, the residual matrix of A_i after the selection of basis vectors u_1, u_2,..., u_j, is defined as

$$R_{ij} = \begin{cases} A_i & j = 0 \\ A_i - proj(A_{ij}) & otherwise \end{cases}$$

where, $proj(A_{ij})$ is the orthogonal projection of the document vectors in A_i onto the span of u_1,u_2,...,u_j, i.e.,

$$proj(A_{ij}) = \sum_{k=1}^{j} u_k u_k^T A_i$$

the residual matrix of A_i describes how much the document vectors in A_i are excluded from the proposed basis vectors u_1, u_2,..., u_j. For the first basis vector computation, residual matrices are initialized as original sub-matrices. The computation of the residual matrix makes the remaining vectors perpendicular to the previous basis vectors, thus ensures that the basis vectors are orthogonal, as the eigenvector computed next is a linear combination of the remaining vectors.

After calculating a basis vector, the algorithm judges whether the sub-matrices have been well represented by the derived basis vectors. The residual ratio was defined as a criterion for this judgement,

$$rr_{ij} = \frac{||R_{ij}||_F^2}{n_i \times (k_i + 1)}$$

where R_{ij} is the residual matrix of A_i after j basis vectors have been selected[1]; n_i is the number of the documents in matrix A_i; k_i is the number of singular vectors that have been selected from matrix A_i. Residual ratios of each sub-matrix are calculated. The sub-matrix with the largest residual ratio is assumed to be the one that contains the most information that has not been represented by the previous chosen basis vectors. The first left singular vector of this sub-matrix is computed and selected as the next basis vector. As described above, the computation of a basis vector uses the corresponding residual matrix. Once a basis vector is selected, its influence from each sub-matrix is subtracted. The procedure is repeated until an expected number of basis vectors have been chosen. The pseudo-code of the algorithm for semantic space construction is shown as follows:

```
1. Partition A into matrices $A_1$,...,$A_q$ corresponding to document clusters, where $A_i$ , $1 \leq i \leq q$, is an $m \times n_i$ ($n_1 \geq n_2 \geq ... \geq n_q$) matrix.
2. For i=1,2,...,q    {$R_i$= $A_i$; k[i]=0;}
3. j=1; r=1;
4. $u_r$= the first unit eigenvector of $R_j R_j^T$;
5. For i=1,2,...,q    $R_i$= $R_i$ - $u_r u_r^T R_i$;
6. k[r]=k[r]+1; r=r+1;
7. For i=1,2,...,q    $rr_i$= $\frac{||R_i||_F^2}{(n_i \times (k[i]+1))}$;
8. j=t if $rr_t > rr_p$ for p=1,2,...,q and $p \neq t$;
9. If $rr_j \leq threshold$ then stop else goto step 4.
```

For the single-link algorithm used in the CSSVD, we use a threshold 0.2 and cosine measure to calculate the similarity between two clusters in our experiments. The performance of the CSSVD is also relative to the number of dimensions of the created

[1] $||A||_F = \sqrt{\sum_{i,j} A_{ij}^2}$

subspace. As described above, the algorithm uses the residual ratio as a stopping criterion for the basis vector computation. In each iteration, after a basis vector is created, the residual ratio is compared to a threshold. Once the residual ratio of each sub-matrix fell below a certain threshold, the process of basis-vector selection is finished. In our experiments, the threshold was trained on corpus.

After all the k basis vectors are chosen, a term-document vector d_i can be converted to d_i^k, a vector in the k-dimensional space, by multiplying the matrix of basis vectors following the standard method of orthogonal transformation,i.e., $d_i^k = [u_1, u_2, ..., u_k]^T d_i$.

3 Evaluation

3.1 Experimental Setup

For the evaluation of the algorithm, 38 topics from the Text REtrieval Conference (TREC) collections were used in our experiments. These topics include foreign minorities, behavioral genetics, steel production, etc. We deleted documents relevant to more than one topic so that each document is related only to one topic. The total number of documents used was 2962. These documents were split into two disjoint groups, called 'pool 1' and 'pool 2'. The number of documents in 'pool 1' and 'pool 2' were 1453 and 1509, respectively. Each of the two groups used 19 topics.

We generated training and testing data by simulating the result obtained by a query search. This simulation is further simplified by selecting documents containing same keywords from each document group. Thirty document sets were generated from each of the two document groups, i.e. 60 document sets in total. The number of documents for each set ranges from 51 to 582 with an average of 128; the number of topics ranges from 5 to 19 with an average of 12. Due to the limited number of the document sets we created, these sets were used both for training and evaluation. For the evaluation of the documents sets from 'pool 1', 'pool 2' was used for training, and vice versa.

To construct the original term-document matrix, the following operations were performed on each of the documents: 1) filtering out all non-text tags in the documents; 2) converting all the characters into lower case; 3) removing stop words - a stoplist containing 319 words was used; and 4) term indexing - the tf.idf scheme was used to calculate a term's weight in a document. Finally, a document set is represented as a matrix $A = [a_{ij}]$, where a_{ij} denotes the normalized weight assigned to term i in document j.

3.2 Evaluation Measures

Our algorithm was motivated by a multi-document summarization application which is mainly based on measuring the similarities and differences among text segments. Therefore, the basic requisite is to accurately measure similarities among texts. Based on this consideration, we used the CSSVD algorithm to create the document vectors in a reduced space for each of the document sets; cosine similarities among these document vectors were computed; and the results were then compared with the TREC relevance judgments. As each of the TREC documents we used has one specific topic. Assume that similarity should be higher for any document pair relevant to the same topic than for any pair relevant to different topics. The algorithm's accuracy for measuring the similarities among documents was evaluated using average precision taken at various recall levels (Harman, 1995). Let p_i denote the document pair that has the i^{th} largest similarity value among all pairs of documents in the document set. The precision for an intra-topic pair p_k is calculated by

$$precision(p_k) = \frac{number\ of\ p_j\ where\ j \leq k}{k}$$

where p_j is an intra-topic pair. The average of the precision values over all intra-topic pairs is computed as the average precision.

3.3 Results

The algorithms are evaluated by the average precision over 60 document sets. In order to make a comparison, two baseline algorithms besides CSSVD are evaluated. One is the vector space model (VSM) without dimension reduction. The other is SVD taking the left singular vectors as the basis vectors.

To treat the selection of dimensions as a separate issue, we first evaluate the algorithms in terms of the best average precision. The 'best average precision' means the best over all the possible numbers of dimensions. The second row of Table 1 shows the best average precision of our algorithm, VSM, and SVD. The best average precision on average over 60 document sets of CSSVD is 69.6%, which is 11.5% higher than VSM and 6.1% higher than SVD.

measure	VSM	SVD	CSSVD
best average precision (%)	58.1	63.5	69.6
average DR (%)	N/A	54.4	32.1
average precision (%)	58.1	59.5	66.8

Table 1: the algorithm performance

In the experiments, we observed that the CSSVD algorithm obtained its best performance with the number of dimensions lower than that of SVD. The Dimensional Ratio (DR) is defined as the number of dimensions of the derived sub-space compared with the dimension number of the original space, i.e.,

$$DR = \frac{\#\ of\ dimensions\ in\ derived\ space}{\#\ of\ dimensions\ in\ original\ space}$$

The average dimensional ratio is calculated over all the 60 document sets. As the algorithms' computational efficiency is dependent on the number of dimensions computed, our interest is in getting good performance with an average dimensional ratio as low as possible. The third row of Table 1 shows the average dimensional ratio that yielded the best average precision. The average dimensional ratio that CSSVD yielded the best average precision is 32.1%, which is 22.3% lower than that of SVD. Thus, our algorithm has the advantage of being computationally inexpensive, assuming that we can find the optimal number of dimensions.

The bottom row of Table 1 shows the average precision of the algorithms. The threshold used in CSSVD algorithm was trained on corpus. Let p be the threshold on residual ratio that yielded the best average precision on the training data. The value of p is then used as the threshold on the evaluation data. For the SVD algorithm, the average dimensional ratio that yielded the best average precision on training data was used as the dimensional ratio to determine the subspace dimensionality in evaluation. The performance shown here are the average of average precision over 60 document sets. Again, the CSSVD achieves the best performance, which is 7.3% higher than the performance of SVD and 8.7% higher than VSM.

4 Conclusion

We have presented an alternative algorithm, the CSSVD, that creates vector representation for linguistic units with reduced dimensionality. The algorithm aims to compensate for SVD's bias towards dominant-topic documents by grouping documents into clusters and selecting basis vectors from each of the clusters. It introduces a threshold on the residual ratio of clusters as a stopping criterion of basis vector selection. It thus treats each topic underlying the document collection equally while focuses on the dominant documents in each topic. The preliminary experiments on measuring document similarities have shown that the CSSVD achieves higher average precision with lower number of dimensions than the baseline algorithms.

Motivated by a multi-document summarization application, the CSSVD algorithm's emphasis on topics and dominant information within each topic meets the general demand of summarization. We expect that the algorithm fits the task of summarization better than SVD. Our future work will focus on more thorough evaluation of the algorithm and integrating it into a summarization system.

5 Acknowledgments

We would like to thank Mark Sanderson, Horacio Saggion, and Robert Gaizauskas for helpful comments at the beginning of this research.

References

Ando R.K. 2000 Latent Sementic Space: Iterative Scaling Improves Precision of Inter-document Similarity Measurement. *Proceedings of ACM SIGIR 2000*, Athens, Greece.

Deerwester S., Dumais S., Furnas G., and Landauer T. 1990. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391-407.

Golub G. and Loan C.V. 1996. *Matrix Computations*. Johns-Hopkins University Press, Maryland, US.

Harman D.K. 1983. Overview of the second Text Retrieval Conference (TREC-2). *Information Processing Management*, 31(3):271-289.