Fast Dimension Reduction for Document Classification Based on Imprecise Spectrum Analysis

Hu Guan† Bin Xiao‡ Jingyu Zhou† Minyi Guo† Tao Yang§
†Shanghai Jiao Tong Univ.
Shanghai, P. R. China
{guanhu,zhou-jy,guo-my}@sjtu.edu.cn
‡HK Polytechnic Univ.
Kowloon, Hong Kong
csbxiao@polyu.edu.hk
§Univ. of California
Santa Barbara, U.S.A.
tyang@cs.ucsb.edu

ABSTRACT

This paper proposes an algorithm called Imprecise Spectrum Analysis (ISA) to carry out fast dimension reduction for document classification. ISA is designed based on the one-sided Jacobi method for Singular Value Decomposition (SVD). To speedup dimension reduction, it simplifies the orthogonalization process of Jacobi computation and introduces a new mapping formula for transforming original DOCument-term vectors. To improve classification accuracy using ISA, a feature selection method is further developed to make inter-class feature vectors more orthogonal in building the initial weighted term-document matrix. Our experimental results show that ISA is extremely fast in handling large term-document matrices and delivers better or competitive classification accuracy compared to SVD-based LSI.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithms, Experimentation, Performance

Keywords

Feature Selection, LSI, SVD, Dimension Reduction

1. INTRODUCTION

Latent Semantic Indexing (LSI) with SVD is an effective dimension reduction method for document classification and other information analysis tasks. The computational overhead of SVD is known to be a bottleneck in dealing with large data sets. Even there are various advancements of the computational methods for SVD (e.g. [12, 7, 10, 1, 5]), SVD computation is still very expensive for larger matrices [11].

In this paper, we propose a fast dimension reduction algorithm called imprecise spectrum analysis (ISA) for document classification with training data. There are three key optimization strategies in ISA to speedup computation while sustaining a good accuracy.

First, we start with the one-sided Jacobi method which is more accurate numerically than the QR method [4]. Given a weighted term-matrix matrix H, the most time consuming part of the Jacobi method is an orthogonalization process expressed as $HV = B$ where V contains right-singular vectors and B is further decomposed as $B = U\Lambda$ where U contains left-singular vectors and Λ is a diagonal matrix containing singular values of H. We simplify the orthogonalization process with an approximation.

Second, to minimize the negative impact of imprecise computation, we consider a weighting design in constructing the initial matrix H. We make column vectors of H more orthogonal by utilizing terms' global scores and class-oriented characteristics.

Finally, we use a fast mapping formula in transforming original feature vectors into a reduced space. Given an original document-term vector d, the new dimension-reduced vector d_{lsi} is computed as $d^T H$. We show this formula has the functional equivalence to $d_{lsi} = d^T U\Lambda$ and has competitive or better accuracy compared to the other existing methods.

2. BACKGROUND

A document-term vector d can be transformed into a low dimension vector d_{lsi} in an LSI space with the following standard mapping formula [3]: $d_{lsi} = d^T U\Lambda^{-1}$ where d is the original document vector $d_{m\times 1}$, U is a column-orthogonal matrix $m \times r$, Λ is a diagonal matrix $r \times r$, and d_{lsi} is the target pseudo-document which has a much lower dimension k ($\leq r$) than the original space $H_{m\times n}$. Some work [10, 9] uses another mapping formula: $d_{lsi} = d^T U$.

The main cost of LSI in those mapping models is deriving the matrix U and Λ in decomposing the matrix H. Among those computing SVD methods, one-sided Jacobi method has the form $HV = B$ for a given matrix $H(m \times n)$ of rank r. Namely the Jacobi method starts from matrix H and performs a sequence of sweeps which essentially post-multiplies H with V and yields matrix B whose columns are orthogonal. The above iterative computation of the Jacobi method performs the following sequence of sweeps to make columns in H orthogonal. Each sweep uses the following plane rotation:

$$[(H_{k+1})_i, (H_{k+1})_j] = [(H_k)_i, (H_k)_j] \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}, \quad (1)$$

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.
Copyright 2010 ACM 978-1-4503-0099-5/10/10 ...$10.00.

where $(H_k)_i$ and $(H_{k+1})_i$ are the i-th column in H before and after the k-th plane rotation, and θ is a rotation angle that is designed to produce two orthogonal columns [6]. When all pair vectors are orthogonal under a precision value, by normalizing columns of B by the norm of vectors, we can get $B = U\Lambda$ and then $H = U\Lambda V^T$.

It is very time-consuming to perform the above orthogonalization computation in the Jacobi method. For example, for a sparse document-term matrix H from the 20-newsgroup corpus, we found that before the first sweep, there were only 5% non-orthogonal pair vectors while this sparse matrix has 5% of non-zero elements. Then many sweeps of the Jacobi method are spent with a plane rotation on getting a 100%-orthogonal matrix B, and the entire process is slow. This uneconomic practice encouraged us to challenge the fundamental problem: can LSI for feature selection perform well when we relax the rigorously orthogonal constraint on matrix B?

3. DESIGN

An analysis on the LSI space [10] shows that feature vectors of those two documents from different topic classes should be nearly orthogonal. In this way, LSI does a particularly good job of classifying documents. Our approach is to relax the orthogonal constraint of B in the above one-sided Jacobi algorithm. Meanwhile, we make those pairs of vectors from different classes to be orthogonal as much as possible through feature scoring by extending a scoring formula called CFC [8] in constructing the initial matrix H. With this in mind, we can consider key pairs of vectors in H are near orthogonal, and thus we remove the time-consuming matrix rotation process in the Jacobi computation. Then the one-sided Jacobi computation can be simplified as the following steps:

- Construct matrix H and then use matrix H as an imprecise matrix B.
- Compute the l_2 norm of all column vectors in B and sort these vectors according to norm.
- Get an imprecise decomposition by $B = U\Lambda$.

To carry out dimension reduction which uses U_k and diagonal matrix Λ, we can select top k norms and take them as top k scalar values in the diagonal matrix Λ. and use corresponding normalized vectors as the matrix U_k. In the rest of this section, we present our weighting formula for initial matrix construction and an error upper-bound when using top k vectors to approximate. We will also present our design for mapping document vectors into the reduced space with analytic results that justify the choice of mapping.

3.1 Initial Matrix Construction and Error Bound

We construct an initial matrix H in a way such that two document vectors in different topic classes have an orthogonal trend. To do this, we extend CFC score [8] by exploiting summarized class information extracted from training data.

Table 1: Term definitions.

Term	Description
n	total # of document vectors
$\|C\|$	total # of classes
$c_{i,j}$	# of times that term t_i occurs in the j-th document
n_j	total # of terms in the j-th document
c_i	$= \sum_{j=1}^{n} c_{i,j}$, t_i's occurrences in corpus
CF_i	# of classes containing t_i in training data

The terms used in this paper is defined in Table 1. Our scoring formula to construct a term-document matrix H is called CFC-E, where weight $h_{i,j}$ for term t_i and document j in matrix H is defined as:

$$\begin{aligned} h_{i,j} =& e_i \times cf_i, \quad where \\ cf_i =& log(\frac{|C|}{CF_i}), \\ e_i =& 1 + \frac{1}{\log n} \sum_{j=1, c_{i,j} \neq 0}^{n} \frac{c_{i,j}}{c_i} \log \frac{c_{i,j}}{c_i}. \end{aligned} \tag{2}$$

After construction of the initial matrix H, we use this matrix H as the target matrix B. Then ISA uses the top k vectors with the maximum norm values from the normalized and sorted matrix H. This essentially means that we approximate the original matrix H with H_k which contains these top k vectors. We can derive the bound of errors raised from this approximation. Given an $m \times n$ matrix H, the relative error ratio of top k vectors to approximate H can be given as

$$\frac{||H - H_k||_F}{||H||_F} \leq \sqrt{1 - \frac{k}{n}} \tag{3}$$

where $||H||_F$ is the the Frobenius norm of H.

Using top k vectors is consistent with the strategy used in SVD-based matrix approximation. The more important a vector in matrix U or V is, the bigger the corresponding σ_i in matrix Λ will be and thus top k singular values with their corresponding singular vectors are selected in SVD-based matrix approximation.

3.2 Design of Mapping Formula

In general, we model the mapping formula as

$$d_{lsi} = d^T U \mathcal{F}(\Lambda).$$

With $\mathcal{F}(\Lambda) = \Lambda^{-1}$, it is the traditional formula $d_{lsi} = d^T U \Lambda^{-1}$. For mapping original vectors, the selected vector u_i is multiplied by a weight factor of $\frac{1}{\sigma_i}$. With $\mathcal{F}(\Lambda) = I$, we have $d_{lsi} = d^T U$. We find from our experiments that the accuracy of mapping model with $\mathcal{F}(\Lambda) = \Lambda^{-1}$ is not as good as the one with $\mathcal{F}(\Lambda) = I$ when the dimension of pseudo vectors k is large. This is because the singular value (Λ) becomes small for a large k value and inverse Λ^{-1} becomes too large in the LAS2 algorithm [2], which leads to a big numerical error.

From the above experimental finding, we infer that it is reasonable to let the mapping formula multiply weights to important vectors u_i proportionally according to values of σ_i. Thus we propose a mapping formula with $\mathcal{F}(\Lambda) = \Lambda$, namely, $d_{lsi} = d^T U \Lambda$. This gives each selected vector u_i a positive weight σ_i in the mapping process, which is consistent to the heuristic that the more important vector u_i is, the bigger weight σ_i is applied. Our experiments presented in 4.1 verify that classification performance of mapping with $\mathcal{F}(\Lambda) = \Lambda$ is close to that with $\mathcal{F}(\Lambda) = I$.

We will show that $d_{lsi} = d^T U \Lambda$ has the same effect as $d_{lsi} = d^T H$ for replacing the part $d_i d_j^T$ of Equation 4 shown below in SVM-based classification. This means that we can choose $d_{lsi} = d^T H$ to map vectors with an advantage that H can be used directly without any expensive computation other than a normalization process.

We analyze the functional equivalence of the above mapping formulas for classification tasks as follows. Given an

SVM-based classifier, for a dual Wolfe optimization problem L_D:

$$L_D = \sum \alpha_i - \frac{1}{2} \sum \sum \alpha_i \alpha_j y_i y_j d_i d_j^T, \quad (4)$$

where y_i is the label of the vector d_i and α_i is the Lagrange multipliers, if dimension reduction is carried out in LSI space, the part $d_i d_j^T$ will be replaced by $(d_{lsi})_i (d_{lsi})_j^T$. If we use the mapping formula $d_{lsi} = d^T U\Lambda$, the part $d_i d_j^T$ in Equation 4 can be substituted by $d_i^T U\Lambda (U\Lambda)^T d_j$. In the singular value decomposition $H = U\Lambda V^T$ for an initial matrix H, we have the transformation

$$HH^T = U\Lambda V^T (U\Lambda V^T)^T = U\Lambda V^T V \Lambda U^T.$$

We can get $HH^T = U\Lambda (U\Lambda)^T$ because $V^T V = I$. Consequently, the part $d_i d_j^T$ can be replaced in $d_i^T U\Lambda (U\Lambda)^T d_j = d_i^T HH^T d_j$. We get an equivalent mapping formula $d_{lsi} = d^T H$ from the mapping formula $d_{lsi} = d^T U\Lambda$.

4. EVALUATION

Table 2 lists different algorithms and approaches compared in our experiments and our evaluation has the following objectives:

- Demonstrate the rationality of $d_{lsi} = d^T U\Lambda$ and evaluate the effect between ISA and $d_{lsi} = d^T U\Lambda$.
- Demonstrate that ISA performs extremely fast.

Table 2: Different approaches.

Names	Descriptions
Base	classification without dimension reduction
SVD-1	standard reduction with $d_{lsi} = d^T U\Lambda^{-1}$
SVD	the same as SVD except that $d_{lsi} = d^T U$
SVD1	standard reduction with $d_{lsi} = d^T U\Lambda$
ISA	imprecise spectrum analysis

To assess the performance of different mapping formulas in classifying tasks, we use an SVM classifier to carry out classifying tasks, and use standard macro-averaging F1 (MacroF1) and micro-averaging F1 (MicroF1) as the accuracy metrics to evaluate the performance. As for SVD computation, we use LAS2 algorithm in the latest SVDLIBC library [1]. Three datasets used for experiments are summarized below.

WebKB. WebKB [2] contains seven categories and 8,203 pages (7,031 training and 1,172 testing). For the "Title","H1", and "URL" parts in the page, we give the weight 5 times more than those in "Body". We kept 28,473 unigram terms that occurred at least once in the training set.

20-newsgroup. This dataset from 20 Usenet newsgroups[3] consists of 19,899 messages (13,272 training and 6,627 testing) and 31,138 unigram terms. We only keep "Subject", "Keywords", and "Content" while words in "Subject" and "Keywords" are given the weight 5 times more than those in "Contents".

Reuters-21578. This dataset[4] contains 6,495 training texts, 2,557 testing texts and 11,430 unique unigram terms in this 52-category corpus. Words in titles are given the weight 5 times more than those in abstracts.

[1]http://tedlab.mit.edu/~dr/SVDLIBC/
[2]http://www.cs.cmu.edu/afs/cs/project/theo-20/www/data
[3]http://kdd.ics.uci.edu/databases/20newsgroups
[4]http://ronaldo.cs.tcd.ie/esslli07/sw/step01.tgz

For above three corpora, we used the tokenizer tool provided in the Trinity College sample. Stemming and word clustering were not applied. Entropy score for initial matrix H are extracted from the whole corpus. Original document vectors are constructed with TF-IDF score.

Parameter Settings. The SVMTorch package [5] and $SVM^{multiclass}$ package[6] are used in classifying tasks. The two SVM classifiers can perform multi-class tasks directly with one-*vs*-others decomposition and default parameter values. Experiments were performed on a machine with Intel Core2 Duo 2.8 GHz CPU, 4 GB memory.

4.1 Rationality of SVD1

On the spectrum of pseudo-document's dimension, the precision in classifying tasks is used to describe the performance of different mapping formulas. We construct H with $e \times cf$ in Reuters-21578, e in WebKB and $tf \times e$ in 20-newsgroup respectively because different scores for H can illustrate the stable performance of SVD1 more convincingly.

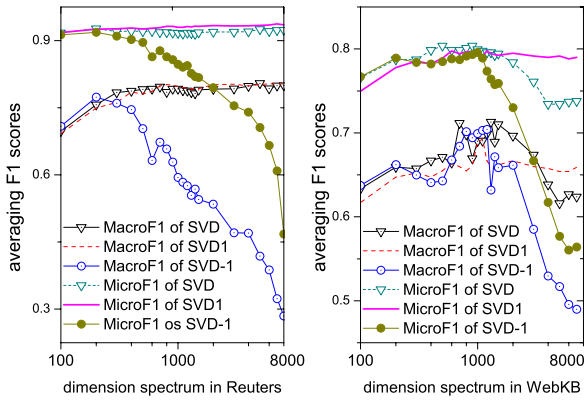

Figure 1: Performance of three mapping formulas for dimension reduction in Reuters and WebKB.

Fig. 1 shows that SVD and SVD1 have produced a comparable performance while they outperform SVD-1 significantly especially for a larger dimension size. For MicroF1, the difference between SVD and SVD1 is relatively small, and is getting bigger with larger dimension size. In WebKB, after dimension is bigger than 2000, SVD1 exhibits more stable performance while SVD-1's performance drops. For MacroF1, SVD is slightly better than SVD1, and SVD1 is sometime slightly better, especially for a larger dimension. Overall speaking, SVD1's performance is close to SVD and they perform better than SVD-1 for a higher dimension.

4.2 SVD1 vs. ISA

Fig. 2 shows that ISA can become competitive to SVD1 after dimension 5,000 in Reuters. If we use the full size of H, Fig. 2 shows that ISA can perform as well as SVD1 on high dimensions. ISA's performance is lower on low dimensions because training data is not sufficient, while SVD1 uses full training matrix and performs better.

Table 3 lists the results of comparison for the Reuters dataset. For example, we select the top k=1000 vectors in matrix H according to the norm of those vectors, then, LAS2 is used to carry out singular value decomposition for H_{1000}. ISA uses H_{1000} fully in its mapping formula $d_{lsi} = d^T H_{1000}$.

[5]http://www.idiap.ch/~bengio/projects/SVMTorch.html
[6]http://svmlight.joachims.org/svm_multiclass.html

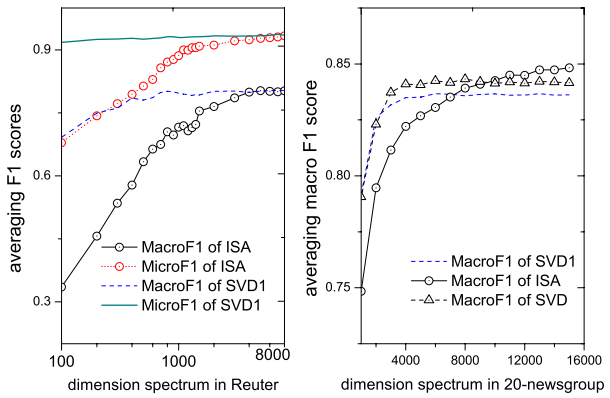

Figure 2: SVD1 vs. ISA in Reuters-21578 and 20-newsgroup.

Table 3: Performance comparison between SVD1 and ISA in Reuters-21578 after SVD is applied to top k vectors.

H_k	ISA		SVD1	
	MicroF1	MacroF1	MicroF1	MacroF1
k=1000	0.8868	0.7162	0.8858	0.7164
k=2000	0.9116	0.7651	0.9116	0.7653
k=3000	0.9214	0.7858	0.9221	0.7866
k=4000	0.9245	0.7996	0.9245	0.7996
k=5000	0.9280	0.8028	0.9280	0.8001
k=6000	0.9292	0.8016	0.9292	0.8010

Table 3 shows that ISA can still perform as well as SVD1 on all dimensions when SVD also only uses the same set of top vectors.

4.3 Processing time cost

For LSI-based classification, two key steps are dimension reduction computation and training using SVM. For LAS2, Table 4 illustrates the cost of a few choices of targeted dimensions and processing time is extremely slow for larger dimensions. ISA only needs to sort the initial matrix H after normalization, so ISA took less than one second to perform matrix decomposition for three corpora.

Table 4: Elapsing seconds of LAS2 and ISA for matrix computation with different targeted dimensions.

Dim	Reuters		20-Newsgroups	
	LAS2	ISA	LAS2	ISA
1000	270	0.29	298	0.37
3000	3340	0.29	4266	0.37
6000	7757	0.29	32059	0.37

In Table 5, for each sample dimension, the training process had been carry out 10 times with $SVM^{multiclass}$ in 20-newsgroup. We extracted the smallest time-cost for LAS2 while the biggest time-cost for ISA. Table 5 shows that SVD's vectors with dimension 3,000 need 9.77 seconds in training while ISA's results with dimension 12,000 need only 5.22 seconds. The pseudo-document vectors mapped using SVD are dense while the pseudo-document vectors produced by ISA are sparse. For example, over 92% of elements in transformed vectors are zero in ISA for 20-newsgroup and the SVM code has taken this advantage.

Table 5: SVM's training cost of CPU Time.

LAS2		ISA	
dim	Cost	dim	Cost
1000	3.70	3000	1.42
2000	6.28	6000	1.81
3000	9.77	10000	4.43
4000	13.35	12000	5.22
5000	16.69	15000	6.28
6000	20.03	19897	8.30

5. CONCLUSIONS

Three tested benchmarks show that our approach can perform extremely fast dimension reduction while have an accuracy competitive to SVD-based LSI in document classification. ISA's complexity is proportional to the number of non-zero elements in H. Though ISA has a bigger suitable dimension (≈0.6n in the tested cases), vectors transformed by ISA are highly sparse and a classification learning algorithm such as SVM can exploit sparse computation to achieve low training time. Our future work is to study ISA for other applications.

Acknowledgment

This work is support in part by NSFC 60811130528. We thank anonymous referees for their valuable comments.

6. REFERENCES

[1] N. Ailon and B. Chazelle. Faster dimension reduction. *Communications of the ACM*, 53(2):97–104, 2010.

[2] M.W. Berry. *Multiprocessor sparse SVD algorithms and applications*. PhD thesis, UIUC, 1991.

[3] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[4] James Demmel and Kresimir Veselic. Jacobi's method is more accurate than qr. *SIAM J. Matrix Anal. Appl*, 13:1204–1245, 1992.

[5] Zlatko Drmač and Krešimir Veselić. New fast and accurate jacobi svd algorithm. i. *SIAM J. Matrix Anal. Appl.*, 29(4):1322–1342, Nov. 2007.

[6] G.R. Gao and S.J. Thomas. An optimal parallel Jacobi-like solution method for the singular value decomposition. In *Proc. of the International Conference on Parallel Processing*, pages 47–53, 1988.

[7] Jing Gao and Jun Zhang. Sparsification strategies in latent semantic indexing. In *Proceedings of the 2003 Text Mining Workshop*, pages 93–103, 2003.

[8] H. Guan, J. Zhou, and M. Guo. A class-feature-centroid classifier for text categorization. In *Proceedings of the 18th international conference on World Wide Web*, pages 201–210, Madrid, Spain, 2009.

[9] H. Kim, P. Howland, and H. Park. Dimension reduction in text classification with support vector machines. *The Journal of Machine Learning Research*, 6:37–53, 2005.

[10] C.H. Papadimitriou, H. Tamaki, P. Raghavan, and S. Vempala. Latent semantic indexing: A probabilistic analysis. In *Proc. of ACM PODS*, pages 159–168, 1998.

[11] Xiaoguang Qi and Brian D. Davison. Web page classification: Features and algorithms. *ACM Comput. Surv.*, 41(2):1–31, 2009.

[12] Llyod N. Trefethen and David Bau, III. *Numerical Linear Algebra*. SIAM, 1997.