Tensor Space Model for Document Analysis

Deng Cai
Dept. of Computer Science
UIUC
dengcai2@cs.uiuc.edu

Xiaofei He
Yahoo! Research Labs
hex@yahoo-inc.com

Jiawei Han
Dept. of Computer Science
UIUC
hanj@cs.uiuc.edu

ABSTRACT

Vector Space Model (VSM) has been at the core of information retrieval for the past decades. VSM considers the documents as vectors in high dimensional space. In such a vector space, techniques like Latent Semantic Indexing (LSI), Support Vector Machines (SVM), Naive Bayes, etc., can be then applied for indexing and classification. However, in some cases, the dimensionality of the document space might be extremely large, which makes these techniques infeasible due to the *curse of dimensionality*. In this paper, we propose a novel **Tensor Space Model** for document analysis. We represent documents as the second order tensors, or matrices. Correspondingly, a novel indexing algorithm called **Tensor Latent Semantic Indexing** (TensorLSI) is developed in the tensor space. Our theoretical analysis shows that TensorLSI is much more computationally efficient than the conventional Latent Semantic Indexing, which makes it applicable for extremely large scale data set. Several experimental results on standard document data sets demonstrate the efficiency and effectiveness of our algorithm.

Categories and Subject Descriptors:
H.3.1 [Information Storage and Retrieval]: Content Analysis and Indexing − *Indexing methods*

General Terms:
Algorithms, Theory, Experimentation, Performance

Keywords:
Tensor Space Model, Vector Space Model, Latent Semantic Indexing, Tensor Latent Semantic Indexing

1. INTRODUCTION

Document indexing and representation has been a fundamental problem in information retrieval for many years. Most of previous works are based on the Vector Space Model (VSM, [3]). The documents are represented as vectors, and each word corresponds to a dimension. Learning techniques such as Latent Semantic Indexing (LSI, [2]), Support Vector Machines, Naive Bayes, etc., can be then applied in such a vector space. The main reason of the popularity of VSM is probably due to the fact that most of the existing learning algorithms can only take vectors as their inputs, rather than tensors.

When VSM is applied, one if often confronted with a document space $\mathbb{R}^n$ with a extremely large n. Let $\mathbf{x} \in \mathbb{R}^n$ denotes the document vector. Let us consider $n = 1,000,000$ and learning a linear function $g(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. In most cases, learning g in such a space is infeasible in the sense of computability. For example, when LSI is applied in such a space, one needs to compute eigen-decomposition of a $1M \times 1M$ matrix [1].

Different from traditional Vector Space Model based document indexing and representation, in this paper, we consider documents as matrices, or the second order tensors. For a document set with n words, we represent the documents as the second order tensors (or, matrices) in $\mathbb{R}^{n_1} \otimes \mathbb{R}^{n_2}$, where $n_1 \times n_2 \approx n$. For examples, a $1,000,000$-dimensional vector can be converted into a 1000×1000 matrix. Let $X \in \mathbb{R}^{n_1} \otimes \mathbb{R}^{n_2}$ denotes the document matrix. Naturally, a linear function in the tensor space can be represented as $f(X) = \mathbf{u}^T X \mathbf{v}$, where $\mathbf{u} \in \mathbb{R}^{n_1}$ and $\mathbf{v} \in \mathbb{R}^{n_2}$. Clearly, $f(X)$ has only $n_1 + n_2$(=2000 in our case) parameters which is much less than $n(= 1,000,000)$ of $g(\mathbf{x})$.

Based on the tensor representation of documents, we propose a novel indexing algorithm called **Tensor Latent Semantic Indexing** (TensorLSI) operated in the tensor space rather than vector space. Sharing similar properties as the conventional Latent Semantic Indexing (LSI) [2], TensorLSI tries to find the principal components of the tensor space. Let $\{\mathbf{u}_i\}_{i=1}^{n_1}$ be a set of basis functions of $\mathbb{R}^{n_1}$ and $\{\mathbf{v}_j\}_{j=1}^{n_2}$ be a set of basis functions of $\mathbb{R}^{n_2}$. It is easy to show that $\{\mathbf{u}_i\mathbf{v}_j^T\}$ forms a basis of $\mathbb{R}^{n_1} \otimes \mathbb{R}^{n_2}$. Thus, TensorLSI aims at finding bases $\{\mathbf{u}_i\}$ and $\{\mathbf{v}_j\}$ such that the projections of the documents onto $\{\mathbf{u}_i\mathbf{v}_j^T\}$ can best represent the documents in the sense of reconstruction error.

It would be important to note that, while searching for the optimal bases $\{\mathbf{u}_i\}$ and $\{\mathbf{v}_j\}$, we need only to compute the eigen-decompositions of two $n_1 \times n_1$ and $n_2 \times n_2$ matrices. This property makes our algorithm particularly applicable for the case when the number of words is extremely large. This work is foundamentally motivated by [4]. For more detailed document analysis using TSM, please see [1].

2. THE ALGORITHM

TensorLSI is fundamentally based on LSI. It tries to project the data to the tensor subspace in which the reconstruction error is minimized. Given a set of document matrices $X_i \in \mathbb{R}^{n_1 \times n_2}$, $i = 1, \cdots, m$. Let $Y_i = U^T X_i V$ denote its projection in the tensor subspace $\mathcal{U} \otimes \mathcal{V}$. The reconstruction

[1]Note, we assume that the number of documents (m) is larger than n. When $m < n$, it suffices to compute the eigen-decomposition of a $m \times m$ matrix.

SIGIR'06, August 6–11, 2006, Seattle, Washington, USA.
ACM 1-59593-369-7/06/0008.

Table 1: Complexity Comparison of TensorLSI and LSI

	Time complexity	Minimum memory	Storage size
TLSI	$O(mn^{1.5})$	$O(n)$	$O(2n)$
LSI	$O((m+n)q^2)$	$O(q^2)$	$O(kn)$

n is the number of features and m is the number of documents
$q = \min(m, n)$
k is usually around several hundreds

error for X_i can be written as $\|X_i - UY_iV^T\|$. Thus, the objective function of TensorLSI can be described as follows:

$$\min_{U,V} \sum_{i=1}^{m} \|X_i - UY_iV^T\|^2 \quad (1)$$

which is equivalent to the following:

$$\min_{U,V} \sum_{i=1}^{m} \|X_i - UU^TX_iVV^T\|^2 \quad (2)$$

With some algebraic steps [1], we can show that, the optimal U and V are given by solving the following two eigenvector problems:

$$\left(\sum_{i=1}^{m} X_iX_i^T\right)\mathbf{u} = \lambda^u\mathbf{u} \quad , \quad \left(\sum_{i=1}^{m} X_i^TX_i\right)\mathbf{v} = \lambda^v\mathbf{v} \quad (3)$$

After obtaining the basis vectors $\{\mathbf{u}_i\}$ $(i = 1, \cdots, n_1)$ and $\{\mathbf{v}_j\}$ $(i = 1, \cdots, n_2)$, each $\mathbf{u}_i\mathbf{v}_j^T$ is a basis of the transformed tensor space, and $\mathbf{u}_i^TX_t\mathbf{v}_j$ $(t = 1, \cdots, m)$ is the coordinate of X_t corresponding to $\mathbf{u}_i\mathbf{v}_j^T$ in this tensor space. When we want to keep the first k principle component of document in the transformed tensor space, we use function

$$f(\mathbf{u}_i, \mathbf{v}_j) = \sum_{t=1}^{m} tr(\mathbf{u}_i^TX_t\mathbf{v}_j\mathbf{v}_i^TX_t^T\mathbf{u}_j) = \sum_{t=1}^{m}(\mathbf{u}_i^TX_t\mathbf{v}_j)^2 \quad (4)$$

to evaluate the importance of $\mathbf{u}_i\mathbf{v}_j^T$ with respect to i and j. $f(\mathbf{u}_i, \mathbf{v}_j)$ reflects the importance of the tensor basis $\mathbf{u}_i\mathbf{v}_j^T$ in terms of reconstruction error. When we want to keep the first k principle component in the transformed tensor space, we sort $f(\mathbf{u}_i, \mathbf{v}_j)$ for all the i and j in decreasing order and choose the first k pairs.

Table 1 lists the computational complexity comparison of TensorLSI and LSI, more detailed analysis can be found in [1].

3. EXPERIMENTS

In this section, document clustering on Reuters-21578 corpus[2] is used to show the effectiveness of our proposed algorithm. We chose k-means as our clustering algorithm and compared three methods. These three methods are listed below:

- k-means on original term-document matrix (Baseline)
- k-means after LSI (LSI)
- k-means after TensorLSI (TLSI)

The clustering result is evaluated by comparing the obtained label of each document with that provided by the document corpus. The accuracy (AC) is used to measure the clustering performance [1].

[2]Reuters-21578 corpus is at `http://www.daviddlewis.com/resources/testcollections/reuters21578/`

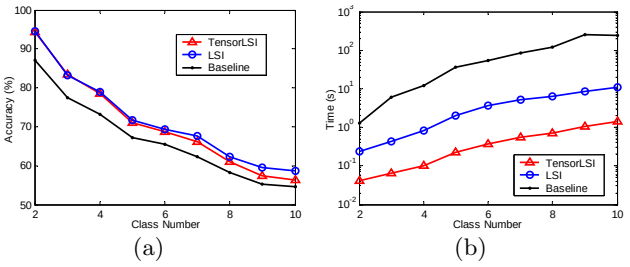

Figure 1: (a)Clustering accuracy with respect to the number of classes. (b) Computation time (time on dimension reduction plus time on k-means) with respect to the number of classes. As can be seen, TensorLSI achieves comparable accuracy with LSI while much faster than LSI.

The evaluations were conducted with different number of clusters, ranging from 2 to 10. For each given cluster number k, 50 tests were conducted on different randomly chosen categories, and the average performance was computed over these 50 tests.

The average accuracy and the computation time (time on dimension reduction plus time on k-means) of three methods are shown on fiure 1. Both LSI and TensorLSI are significantly better than baseline with respect to both clustering accuracy and computation time. For $k = (2, 3, 4, 5, 6)$, LSI and TensorLSI achieved almost same clustering accuracy. For $k = (8, 9, 10)$. Clustering using LSI is slightly better than clustering using TensorLSI. However, in all cases, the computation time (time on dimension reduction plus time on k-means) of TensorLSI is extremely shorter than that of LSI.

4. CONCLUSIONS

A novel document representation and indexing method has been proposed in this paper, called Tensor Latent Semantic Indexing. Different from conventional LSI which considers documents as vectors, TensorLSI considers documents as the second order tensors, or matrices. Based on the tensor representation, TensorLSI tries to find an optimal basis for the tensor subspace in terms of reconstruction error. Also, our theoretical analysis shows that TensorLSI can be much more efficient than LSI in time, memory and storage especially when the number of documents is larger than the number of words.

5. REFERENCES

[1] D. Cai, X. He, and J. Han. Tensor space model for document analysis. Technical report, Computer Science Department, UIUC, UIUCDCS-R-2006-2715, April 2006.

[2] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[3] G. Salton, A. Wong, and C. S. Yang. A vector space model for information retrieval. *Communications of the ACM*, 18(11):613–620, 1975.

[4] J. Ye. Generalized low rank approximations of matrices. *Machine Learning*, 61(1-3):167–191, Nov. 2005.