Document Clustering Using Locality Preserving Indexing

Deng Cai, Xiaofei He, and Jiawei Han, *Senior Member*, *IEEE*

Abstract—We propose a novel document clustering method which aims to cluster the documents into different semantic classes. The document space is generally of high dimensionality and clustering in such a high dimensional space is often infeasible due to the curse of dimensionality. By using Locality Preserving Indexing (LPI), the documents can be projected into a lower-dimensional semantic space in which the documents related to the same semantics are close to each other. Different from previous document clustering methods based on Latent Semantic Indexing (LSI) or Nonnegative Matrix Factorization (NMF), our method tries to discover both the geometric and discriminating structures of the document space. Theoretical analysis of our method shows that LPI is an unsupervised approximation of the supervised Linear Discriminant Analysis (LDA) method, which gives the intuitive motivation of our method. Extensive experimental evaluations are performed on the Reuters-21578 and TDT2 data sets.

Index Terms—Document clustering, locality preserving indexing, dimensionality reduction, semantics.

✦

1 INTRODUCTION

DOCUMENT clustering is one of the most crucial techniques to organize the documents in an unsupervised manner. It has received a lot of attention in recent years [18], [28], [27], [17], [24].

Many clustering methods have been proposed, such as k-means [20], naive Bayes or Gaussian mixture model [16], [1], [18], single-link [16], and DBSCAN [11]. From different perspectives, these clustering methods can be classified into agglomerative or divisive, hard or fuzzy, deterministic or stochastic. The typical data clustering tasks are directly performed in the data space. However, the document space is always of very high dimensionality, ranging from several hundreds to thousands. Due to the consideration of the *curse of dimensionality*, it is desirable to first project the documents into a lower-dimensional subspace in which the semantic structure of the document space becomes clear. In the low-dimensional semantic space, the traditional clustering algorithms can be then applied. To this end, spectral clustering [23], [21], clustering using LSI [29], and clustering based on nonnegative matrix factorization [28], [27] are the most well-known techniques.

Some recent work on spectral clustering shows its capability to handle highly nonlinear data (the data space has high curvature at every local area). Also, its strong connections to differential geometry make it capable of discovering the manifold structure of the document space. The spectral clustering usually clusters the data points using the top eigenvectors of *graph Laplacian* [5], which is defined on the affinity matrix of data points. From the graph partitioning perspective, the spectral clustering tries to find the best cut of the graph so that the predefined criterion function can be optimized. Many criterion functions, such as the ratio cut [4], average association [23], normalized cut [23], and min-max cut [8] have been proposed along with the corresponding eigen-problem for finding their optimal solutions. From the perspective of dimensionality reduction, spectral clustering embeds the data points into a low dimensional space where the traditional clustering algorithm (e.g., k-means) is then applied. One major drawback of these spectral clustering algorithms might be that they use the nonlinear embedding (dimensionality reduction) which is only defined on "training" data. They have to use all the data points to learn the embedding. When the data set is very large, to learn such an embedding is computational expensive, which restricts the application of spectral clustering on large data sets.

Latent Semantic Indexing (LSI) [7] is one of the most popular linear document indexing methods which produces low dimensional representations. LSI aims to find the best subspace approximation to the original document space in the sense of minimizing the *global* reconstruction error. In other words, LSI seeks to uncover the most representative features rather the most discriminative features for document representation. Therefore, LSI might not be optimal in discriminating documents with different semantics which is the ultimate goal of clustering.

Recently, Xu et al. applied the Nonnegative Matrix Factorization (NMF) algorithm for document clustering [28], [27]. They model each cluster as a linear combination of the data points, and each data point as a linear combination of the clusters. They also compute the linear coefficients by minimizing the *global* reconstruction error of the data points using Nonnegative Matrix Factorization. Thus, the NMF method still focuses on the global geometrical structure of document space. Moreover, the iterative update method for solving NMF problem is computationally expensive.

In this paper, we propose a novel document clustering algorithm by using Locality Preserving Indexing (LPI).

- *D. Cai and J. Han are with the Department of Computer Science, University of Illinois at Urbana Champaign, 201 N. Goodwin Ave., Urbana, IL 61801. E-mail: {dengcai2, hanj}@cs.uiuc.edu.*
- *X. He is with the Department of Computer Science, The University of Chicago, 1100 East 58th Street, Chicago, IL 60637. E-mail: xiaofei@cs.uchicago.edu.*

Different from LSI, which aims to discover the global Euclidean structure, LPI aims to discover the *local* geometrical structure. LPI can have more discriminating power. Thus, the documents related to the same semantics are close to each other in the low-dimensional representation space. Also, LPI is derived by finding the optimal linear approximations to the eigenfunctions of the Laplace Beltrami operator on the document manifold. The Laplace Beltrami operator takes the second order derivatives of the functions on the manifolds. It evaluates the smoothness of the functions. Therefore, it can discover the nonlinear manifold structure to some extent. Some theoretical justifications can be traced back to He et al. [15], [14]. The original LPI is not optimal in the sense of computation in that the obtained basis functions might contain a trivial solution. The trivial solution contains no information and is thus useless for document indexing. A modified LPI is proposed to obtain better document representations. In this low-dimensional space, we then apply traditional clustering algorithms such as k-means to cluster the documents into semantically different classes.

The rest of this paper is organized as follows: In Section 2, we give a brief review of LSI and LPI. Section 3 introduces our proposed document clustering algorithm. Some theoretical analysis is provided in Section 4. The experimental results are shown in Section 5. Finally, we give concluding remarks and future work in Section 6.

2 A BRIEF REVIEW OF LSI AND LPI

LSI is one of the most popular algorithms for document indexing. It is fundamentally based on SVD (Singular Value Decomposition). Given a set of documents $\mathbf{x}_1, \cdots, \mathbf{x}_n \in \mathbb{R}^m$, they can be represented as a term-document matrix $X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n]$. Suppose the rank of X is r, LSI decomposes the X using SVD as follows:

$$X = U\Sigma V^T,$$

where $\Sigma = diag(\sigma_1, \cdots, \sigma_r)$ and $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r$ are the singular values of X, $U = [\mathbf{a}_1, \cdots, \mathbf{a}_r]$ and $\mathbf{a}_i$ is called the left singular vector, and $V = [\mathbf{v}_1, \cdots, \mathbf{v}_r]$ and $\mathbf{v}_i$ is called the right singular vector. LSI uses the first k vectors in U as the transformation matrix to embed the original documents into a k-dimensional subspace. It can be easily checked that the column vectors of U are the eigenvectors of XX^T. The basic idea of LSI is to extract the most representative features at the same time the reconstruction error can be minimized. Let $\mathbf{a}$ be the transformation vector. The objective function of LSI can be stated below:

$$\mathbf{a}_{opt} = \arg\min_{\mathbf{a}} \|X - \mathbf{a}\mathbf{a}^T X\|^2$$
$$= \arg\max_{\mathbf{a}} \mathbf{a}^T X X^T \mathbf{a},$$

with the constraint

$$\mathbf{a}^T \mathbf{a} = 1.$$

Since XX^T is symmetric, the basis functions of LSI are orthogonal. It would be important to note that XX^T becomes the data covariance matrix if the data points have a zero mean, i.e., $X\mathbf{e} = \mathbf{0}$, where $\mathbf{e} = [1, \cdots, 1]^T$. In such a case, LSI is identical to Principal Component Analysis [10].

For more details on theoretical interpretations of LSI using SVD, refer to [2], [9], [22].

Different from LSI, which aims to extract the most representative features, LPI aims to extract the most discriminative features. Given a similarity matrix S, LPI can be obtained by solving the following minimization problem:

$$\mathbf{a}_{opt} = \arg\min_{\mathbf{a}} \sum_{i=1}^{n} \left(\mathbf{a}^T \mathbf{x}_i - \mathbf{a}^T \mathbf{x}_j\right)^2 S_{ij}$$
$$= \arg\min_{\mathbf{a}} \mathbf{a}^T X L X^T \mathbf{a},$$

with the constraint

$$\mathbf{a}^T X D X^T \mathbf{a} = 1,$$

where $L = D - S$ is the *graph Laplacian* [5] and $D_{ii} = \sum_j S_{ij}$. D_{ii} measures the local density around $\mathbf{x}_i$. LPI constructs the similarity matrix S as:

$$S_{ij} = \begin{cases} \mathbf{x}_i^T \mathbf{x}_j, & \text{if } \mathbf{x}_i \text{ is among the } p \text{ nearest neighbors of } \mathbf{x}_j \\ & \text{or } \mathbf{x}_j \text{ is among the } p \text{ nearest neighbors of } \mathbf{x}_i \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the objective function in LPI incurs a heavy penalty if neighboring points $\mathbf{x}_i$ and $\mathbf{x}_j$ are mapped far apart. Therefore, minimizing it is an attempt to ensure that, if $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close," then $y_i (= \mathbf{a}^T \mathbf{x}_i)$ and $y_j (= \mathbf{a}^T \mathbf{x}_j)$ are close as well [13]. Finally, the basis functions of LPI are the eigenvectors associated with the smallest eigenvalues of the following generalized eigen-problem:

$$XLX^T \mathbf{a} = \lambda XDX^T \mathbf{a}.$$

3 THE PROPOSED METHOD

In this section, we describe our clustering algorithm, which can be thought of as a combination of subspace learning and k-means. We begin with the motivations of our work.

3.1 Motivation

In this section, we will provide some motivations about the reasoning of LPI followed by a traditional clustering algorithm like k-means.

Generally, the document space is of high dimensionality, typically ranging from several thousands to tens of thousands. Learning in such a high-dimensional space is extremely difficult due to the *curse of dimensionality*. Thus, document clustering necessitates some form of dimensionality reduction. One of the basic assumptions behind data clustering is that, if two data points are close to each other in the high-dimensional space, they tend to be grouped into the same cluster. Therefore, the optimal document indexing method should be able to discover the local geometrical structure of the document space. To this end, the LPI algorithm is of particular interest. LSI is optimal in the sense of reconstruction. It respects the global Euclidean structure while failing to discover the intrinsic geometrical structure, especially when the document space is nonlinear (see [14] for details).

Another consideration is due to the discriminating power. One can expect that the documents should be projected into the subspace in which the documents with different semantics can be well separated, while the documents with common semantics can be clustered. As indicated in [14], LPI

is an optimal unsupervised approximation to the Linear Discriminant Analysis algorithm which is supervised. Therefore, LPI can have more discriminant power than LSI. There are some other linear subspace learning algorithms, such as informed projection [6] and Linear Dependent Dimensionality Reduction [25]. However, none of them has shown discriminating power.

Finally, it would be interesting to note that LPI is fundamentally based on manifold theory [14], [15]. LPI tries to find a linear approximation to the eigenfunctions of the Laplace Beltrami operator on the compact Riemannian manifold (see [15] for details). Therefore, LPI is capable of discovering the nonlinear structure of the document space to some extent.

3.2 Clustering Based on Locality Preserving Indexing

Consider a set of documents $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n \in \mathbb{R}^m$. Suppose $\mathbf{x}_i$ has been normalized to 1, thus, the dot product of two document vectors is exactly the cosine similarity of the two documents. Our clustering algorithm is performed as follows:

1. **Constructing the adjacency graph**. Let G denote a graph with n nodes. The ith node corresponds to the document $\mathbf{x}_i$. We put an edge between nodes i and j if $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close," i.e., $\mathbf{x}_i$ is among p nearest neighbors of $\mathbf{x}_j$ or $\mathbf{x}_j$ is among p nearest neighbors of $\mathbf{x}_i$. Some fast p nearest neighbor search algorithms can be used here to improve the overall speed [12], [30].

2. **Choosing the weights**. If nodes i and j are connected, put

$$S_{ij} = \mathbf{x}_i^T \mathbf{x}_j.$$

Otherwise, put $S_{ij} = 0$. The weight matrix S of graph G models the local structure of the document space. We define D as a diagonal matrix whose entries are column (or row, since S is symmetric) sums of S, i.e., $D_{ii} = \sum_j S_{ji}$. We also define $L = D - S$, which is called the Laplacian matrix in spectral graph theory [5].

3. **Data Preprocessing and SVD Projection**. We remove the weighted mean of $\mathbf{x}$ from each $\mathbf{x}$

$$\hat{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{x}}, \quad \bar{\mathbf{x}} = \frac{1}{\left(\sum_i D_{ii}\right)} \left(\sum_i \mathbf{x}_i D_{ii}\right)$$

and project the document vector into the SVD subspace by throwing away those *zero* singular values.

$$\widehat{X} = U\Sigma V^T,$$

where $\widehat{X} = [\hat{\mathbf{x}}_1, \cdots, \hat{\mathbf{x}}_n]$. We denote the transformation matrix of SVD by W_{SVD}, i.e., $W_{SVD} = U$. After SVD projection, the document vector $\hat{\mathbf{x}}$ becomes $\tilde{\mathbf{x}}$:

$$\tilde{\mathbf{x}} = W_{SVD}^T \hat{\mathbf{x}}.$$

After this step, the term-document matrix X becomes $\widetilde{X} = [\tilde{\mathbf{x}}_1, \cdots, \tilde{\mathbf{x}}_n]$.

4. **LPI Projection**. Compute the eigenvectors and eigenvalues for the generalized eigen-problem:

$$\widetilde{X}L\widetilde{X}^T\mathbf{a} = \lambda\widetilde{X}D\widetilde{X}^T\mathbf{a}. \qquad (1)$$

Let $W_{LPI} = [\mathbf{a}_1, \cdots, \mathbf{a}_k]$ be the solutions of (1), ordered according to their eigenvalues, $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_k$. Thus, the embedding is as follows:

$$\mathbf{x} \to \mathbf{y} = W^T\hat{\mathbf{x}},$$
$$W = W_{SVD}W_{LPI}, \text{ and}$$
$$\hat{\mathbf{x}} = \mathbf{x} - \frac{1}{\left(\sum_i D_{ii}\right)} \left(\sum_i \mathbf{x}_i D_{ii}\right),$$

where $\mathbf{y}$ is a k-dimensional representation of the document $\mathbf{x}$. W is the transformation matrix.

5. **Clustering in the Lower-Dimensional Semantic Space**. Now, we get lower-dimensional representations of the original documents. In the reduced semantic space, those documents belonging to the same underlying class are close to one another. The traditional clustering methods (we choose k-means in this paper) can be applied in the reduced semantic space. The connection between LPI and LDA motivates us to use $k-1$ eigenvectors, where k is the number of clusters [14], [10].

3.3 Computational Analysis

In this section, we provide a computational analysis of our algorithm. One of the major modifications over the original LPI [14] lies in the third step of our algorithm. In the new algorithm, we remove the weighted mean of the document vectors and use SVD to ensure that the term-document matrix $\widetilde{X}$ is of full rank in row.

Making the matrix $\widetilde{X}$ of full rank in row can guarantee that the matrix $\widetilde{X}D\widetilde{X}^T$ is positive definite, which is necessary in solving the generalized eigen-problem (1) in the fourth step [13].

In the following, we discuss why it is necessary to remove the weighted mean from the original document vectors. We first analyze the properties of the eigenvector of the eigen-problem (1). Let $\mathbf{1} = [1, 1, \cdots, 1]^T$. We have the following theorem:

Theorem 1. *If there exists $\mathbf{a}_0$ which satisfies $\widetilde{X}^T\mathbf{a}_0 = \mathbf{1}$ (in other words, the vector $\mathbf{1}$ in the space spanned by the column vectors of $\widetilde{X}^T$), $\mathbf{a}_0$ is the eigenvector of (1) with respect to eigenvalue 0.*

Proof.

$$\widetilde{X}L\widetilde{X}^T\mathbf{a}_0 = \widetilde{X}(D - S)\mathbf{1} = \widetilde{X}(D\mathbf{1} - S\mathbf{1})$$
$$= \widetilde{X}\mathbf{0} = \mathbf{0} = 0\widetilde{X}D\widetilde{X}^T\mathbf{a}_0.$$

Therefore, $\mathbf{a}_0$ is the eigenvector of (1) with respect to eigenvalue 0.   □

Since the matrices $\widetilde{X}L\widetilde{X}^T$ and $\widetilde{X}D\widetilde{X}^T$ are both positive semidefinite, the eigenvalues of (1) are no less than zero. Therefore, $\mathbf{a}_0$ will be the first eigenvector. However, it is clearly of no use for document representation since all the document vectors collapse into a single point along this direction.

The LPI eigen-problem (1) can be written as the Rayleigh Quotient format [13]:

$$\mathbf{a}_0 = \arg\min_{\mathbf{a}} \frac{\mathbf{a}^T \widetilde{X}L\widetilde{X}^T\mathbf{a}}{\mathbf{a}^T \widetilde{X}D\widetilde{X}^T\mathbf{a}}, \text{ and}$$

$$\mathbf{a}_i = \underset{\mathbf{a}_i^T \widetilde{X}D\widetilde{X}^T\mathbf{a}_j=0,\, 0\le j<i}{\arg\min} \frac{\mathbf{a}^T \widetilde{X}L\widetilde{X}^T\mathbf{a}}{\mathbf{a}^T \widetilde{X}D\widetilde{X}^T\mathbf{a}}.$$

If $\mathbf{1}$ is in the space spanned by the column vectors of $\widetilde{X}^T$, the eigenvectors of LPI eigen-problem (1) have the following property:

$$\widetilde{X}^T\mathbf{a}_0 = \mathbf{1} \text{ and } \mathbf{a}_i^T\widetilde{X}D\mathbf{1} = 0, \ i > 0. \tag{2}$$

Now, let us consider the data preprocessing step of our algorithm. We can show that, after removing the weighted mean of document vectors, for any vector $\mathbf{a}$, $\mathbf{a}^T\widetilde{X}D\mathbf{1} = 0$.

$$\mathbf{a}^T\widetilde{X}D\mathbf{1} = \mathbf{a}^T W_{SVD}^T\hat{X}D\mathbf{1} = \mathbf{a}^T W_{SVD}^T[\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \cdots, \hat{\mathbf{x}}_n]D\mathbf{1}$$
$$= \mathbf{a}^T W_{SVD}^T\left(\sum_i \hat{\mathbf{x}}_i D_{ii}\right)$$
$$= \mathbf{a}^T W_{SVD}^T\left(\sum_i \mathbf{x}D_{ii} - \sum_i \bar{\mathbf{x}}D_{ii}\right)$$
$$= \mathbf{a}^T W_{SVD}^T(\sum_i \mathbf{x}D_{ii} - \bar{\mathbf{x}}\sum_i D_{ii}) = \mathbf{a}^T W_{SVD}^T\mathbf{0} = \mathbf{0}.$$

Thus, the preprocessing step of our algorithm can guarantee that the solutions of (1) will not contain the trivial eigenvector as described above and all the eigenvectors will satisfy the second part of (2).

4 THEORETICAL ANALYSIS

In this section, we give the theoretical analysis of our algorithm. First, we will discuss the relationship between LPI, LSI, and LDA. We will show that the key difference between them is the different ways to construct the adjacency graph. The adjacency graph construction in LPI gives a best approximation to LDA which is supervised. Second, our clustering algorithm has intrinsic connections to spectral clustering and manifold learning. This makes it clear that our algorithm is capable of dealing with complex data.

4.1 Relationship between LPI and LSI

LPI is essentially obtained from a graph model. The graph structure represents the geometrical structure of the document space. In our algorithm, a p-nearest neighbor graph is constructed to discover the *local* manifold structure. Intuitively, LPI with a complete graph should discover the *global* structure. In this section, we present a theoretical analysis on the relationship between LPI and LSI. Specifically, we show that LPI with a complete graph is similar to LSI.

As shown in Section 2, LSI tries to solve the maximum eigenvalue problem:

$$XX^T\mathbf{a} = \lambda\mathbf{a}.$$

In LPI, recall that the weight on an edge linking $\mathbf{x}_i$ and $\mathbf{x}_j$ is set to their inner product $\mathbf{x}_i^T\mathbf{x}_j$. Thus, the affinity matrix S of the complete graph can be written as $X^T X$. Since we first apply SVD to remove the components corresponding to the zero singular value, the matrix XX^T is of full rank. The generalized minimum eigenvalue problem of LPI can be written as follows:

$$XLX^T\mathbf{a} = \lambda XDX^T\mathbf{a}$$
$$\Rightarrow X(D-S)X^T\mathbf{a} = \lambda XDX^T\mathbf{a}$$
$$\Rightarrow XSX^T\mathbf{a} = (1-\lambda)XDX^T\mathbf{a} \tag{3}$$
$$\Rightarrow XX^TXX^T\mathbf{a} = (1-\lambda)XDX^T\mathbf{a}.$$

If we assume the diagonal matrix D is close to the identity matrix, $XDX^T \approx XX^T$, the *minimum* eigenvalues of (3) correspond to the *maximum* eigenvalues of the following equation:

$$XX^TXX^T\mathbf{a} = \lambda XX^T\mathbf{a}.$$

Since XX^T is of full rank, we get:

$$XX^T\mathbf{a} = \lambda\mathbf{a},$$

which is just the eigenvalue problem of LSI. The above analysis shows that LPI with a complete graph is actually similar to LSI. Both of them discover the global structure. The only difference is that there is a diagonal matrix D in LPI which reflects the importance of the different document vectors. In LSI, every document vector is treated as equally important. In LPI, the weight of document $\mathbf{x}_i$ is D_{ii}. We define $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$ as the center vector of these document vectors. In complete graph situation, we have

$$D_{ii} = \sum_{j=1}^n S_{ij} = \sum_{j=1}^n (X^T X)_{ij} = \sum_{j=1}^n \mathbf{x}_i^T\mathbf{x}_j$$
$$= \mathbf{x}_i^T\sum_{j=1}^n \mathbf{x}_j = n\mathbf{x}_i^T\bar{\mathbf{x}} = \delta\mathbf{x}_i^T\frac{\bar{\mathbf{x}}}{||\bar{\mathbf{x}}||},$$

where $\delta = n||\bar{\mathbf{x}}||$ is a constant. Note that all the $\mathbf{x}_i$s are normalized to 1. Thus, they are distributed on a unit hypersphere. $\bar{\mathbf{x}}/||\bar{\mathbf{x}}||$ is also on this unit hypersphere. Thus, D_{ii} evaluates the cosine of the angle between vectors $\mathbf{x}_i$ and $\bar{\mathbf{x}}$. In other words, D_{ii} evaluates the cosine similarity between document $\mathbf{x}_i$ and the center. The closer to the center the document is, the larger weight it has. Some previous studies [28] show that such D will improve the performance and our experiments will also show this.

4.2 Relationship between LPI and LDA

In a supervised mode, the label information is available, so we can apply Linear Discriminant Analysis (LDA) to reduce the document space to a low-dimensional space in which the documents of different classes are far from each other and, at the same time, the documents of the same class are close to each other. LDA is optimal in the sense of discrimination.

Suppose the data points belong to k classes. LDA can be obtained by solving the following maximization problem:

$$\mathbf{a}_{opt} = \arg\max_{\mathbf{a}} \frac{|\mathbf{a}^T S_b\mathbf{a}|}{|\mathbf{a}^T S_w\mathbf{a}|},$$
$$S_b = \sum_{i=1}^k n_i(\mathbf{m}^i - \mathbf{m})(\mathbf{m}^i - \mathbf{m})^T, \text{ and }$$
$$S_w = \sum_{i=1}^k\left(\sum_{j=1}^{n_i}(\mathbf{x}_j^i - \mathbf{m}^i)(\mathbf{x}_j^i - \mathbf{m}^i)^T\right).$$

This leads to the following generalized maximum eigenvalue problem:

$$S_b\mathbf{a} = \lambda S_w\mathbf{a}, \tag{4}$$

where $\mathbf{m}$ is the total sample mean vector, n_i is the number of samples in the ith class, $\mathbf{m}^i$ is the average vector of the ith class, and $\mathbf{x}_j^i$ is the jth sample in the ith class. We call S_w the *within-class scatter matrix* and S_b the *between-class scatter matrix*.

We define:

$$W_{ij} = \begin{cases} \frac{1}{n_l}, & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ both belong to the } l\text{th class;} \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

and

$$D_{ij} = \begin{cases} \sum_j W_{ij}, & \text{if } i = j \\ 0, & \text{if } i \neq j. \end{cases}$$

It is easy to check that the row sum of W is 1, therefore, the diagonal matrix D is exactly the identity matrix I.

$$L = D - W = I - W.$$

With some algebraic steps [14], we can show that, if the sample mean is zero, the eigen-problem of (4) is equivalent to

$$XLX^T\mathbf{a} = \lambda XX^T\mathbf{a}. \quad (6)$$

This analysis tells us that, if the affinity matrix S in LPI is defined as the W in (5), the result of LPI will be identical to the LDA.

In the clustering scenario, the label information is not available. Therefore, the matrix W is unknown. However, based on the assumption that neighboring points probably belong to the same underlying class, the affinity matrix S in our LPI algorithm gives an optimal approximation to the similarity matrix W in (5). Even though the label information is not available, we can still discover the discriminating structure to some extent by using LPI. Thus, in the subspace obtained by LPI, those data points belonging to the same underlying class are close to one another. Note that an upper bound of the number of nonzero eigenvalues of LDA is $k - 1$, where k is the number of classes (please see [10] for details). This motivates us to use $k - 1$ eigenvectors in LPI for clustering the documents into k categories.

4.3 The Construction of Affinity Matrix S

The previous two sections show that the key difference among LSI, LPI, and LDA is the construction of the affinity matrix (the weighted matrix of graph). The LSI tries to discover the global structure (with the complete weighted graph). The LPI tries to discover the local geometrical structure of the document space (with a p-nearest neighbor graph). The LDA is performed in supervised mode, thus the graph can be constructed to reflect the label information. From this point of view, the LPI can be performed in either an supervised, unsupervised, or semisupervised manner. The label information can be used to guide the construction of affinity matrix S.

The construction of the affinity matrix is also very important in spectral clustering [23], [21] and spectral embedding [3]. It includes two steps: constructing the graph and setting the weight. In our algorithm, we construct a p-nearest neighbor graph and choose the dot product (cosine similarity) as the weight. There are also some other choices as discussed below.

In the step of graph construction, we put an edge between nodes i and j if $\mathbf{x}_i$ and $\mathbf{x}_j$ are "close." There are two variations:

1. **p-nearest neighbors**. Nodes i and j are connected by an edge if $\mathbf{x}_i$ is among p nearest neighbors of $\mathbf{x}_j$ or $\mathbf{x}_j$ is among p nearest neighbors of $\mathbf{x}_i$. The advantage of this method is that it is simpler to

choose connected edges, which tends to lead to connected graphs. However, the edges so chosen are less geometrically intuitive.

2. **ϵ neighbors**. Nodes i and j are connected by an edge if $\|\mathbf{x}_i - \mathbf{x}_j\|^2 < \epsilon$. The advantage of this method is that it is geometrically motivated and the relationship is naturally symmetric. However, the method often leads to graphs with several disconnected components, and it is difficult to choose ϵ.

In the step of setting the weight, there are several choices:

1. **0-1 weighting**. $S_{ij} = 1$ if and only if nodes i and j are connected by an edge. This is the simplest weighting method and is very easy to compute.

2. **Gaussian kernel weighting**. If nodes i and j are connected, put

$$S_{ij} = e^{-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|^2}{t}}.$$

The Gaussian kernel weighting is also called heat kernel weighting. It has an intrinsic connection to the Laplace Beltrami operator on differentiable functions on a manifold [3].

3. **Polynomial kernel weighting**. If nodes i and j are connected, put

$$S_{ij} = (\mathbf{x}_i^T\mathbf{x}_j + 1)^d.$$

The parameter d in the equation indicates the degree of the polynomial kernel. Order d polynomial kernels can discover a nonlinear structure with polynomial basis functions of order d.

4. **Dot-product weighting**. If nodes i and j are connected, put

$$S_{ij} = \mathbf{x}_i^T\mathbf{x}_j.$$

Note that, if $\mathbf{x}$ is normalized to 1, the dot product of two vectors is equivalent to the cosine similarity of the two vectors. The dot-product weighting can discover the linear Euclidean structure of the document space.

We will compare the different weighting choices in our experiments section.

4.4 Relationship between Clustering Using LPI and Spectral Clustering

The spectral clustering algorithm [23], [26], [21] can be thought of as a combination of spectral dimensionality reduction [3] and a traditional clustering method, such as k-means.

Spectral dimensionality reduction or spectral embedding is the key part of spectral clustering. In this step, a weighted graph S is first constructed as described in Section 4.3. We define D as a diagonal matrix whose entries are column (or row, since S is symmetric) sums of S, $D_{ii} = \sum_j S_{ji}$. We also define $L = D - S$. All the spectral clustering methods can be reduced to an eigen-problem. The different methods have different eigen-problems.

1. Normalized cut [23] and Min-Max cut [8] have the following *minimum* eigenvalue problem:

$$L\mathbf{y} = \lambda D\mathbf{y}. \quad (7)$$

TABLE 1
Thirty Semantic Categories from Reuters-21578 Used in Our Experiments

category	num of doc	category	num of doc	category	num of doc
earn	3713	money-supply	87	iron-steel	37
acq	2055	gnp	63	ipi	36
crude	321	cpi	60	nat-gas	33
trade	298	cocoa	53	veg-oil	30
money-fx	245	alum	45	tin	27
interest	197	grain	45	cotton	24
ship	142	copper	44	bop	23
sugar	114	jobs	42	wpi	20
coffee	110	reserves	38	pet-chem	19
gold	90	rubber	38	livestock	18

2. Ng's method [21] has the following *maximum* eigenvalue problem:

$$D^{-1/2}SD^{-1/2}\mathbf{y} = \lambda\mathbf{y}. \tag{8}$$

3. Average association [23] and Ratio cut [4] have the following *minimum* eigenvalue problem:

$$L\mathbf{y} = \lambda\mathbf{y}. \tag{9}$$

Let $\mathbf{z} = D^{1/2}\mathbf{y}$ with some mathematical deduction of (7), we have:

$$
\begin{aligned}
& L\mathbf{y} = \lambda D\mathbf{y} \\
\Rightarrow & (D - S)\mathbf{y} = \lambda D\mathbf{y} \\
\Rightarrow & S\mathbf{y} = (1 - \lambda)D\mathbf{y} \\
\Rightarrow & SD^{-1/2}D^{1/2}\mathbf{y} = (1 - \lambda)D^{1/2}D^{1/2}\mathbf{y} \\
\Rightarrow & SD^{-1/2}\mathbf{z} = (1 - \lambda)D^{1/2}\mathbf{z} \\
\Rightarrow & D^{-1/2}SD^{-1/2}\mathbf{z} = (1 - \lambda)\mathbf{z}.
\end{aligned}
$$

Thus, the *maximum* eigenvalue problem (8) has a direct connection to the *minimum* eigenvalue problem (7). Let $\mathbf{y}$ be the eigenvector of (7) with eigenvalue λ, then $D^{1/2}\mathbf{y}$ is the eigenvector of (8) with eigenvalue $1 - \lambda$. In the following discussion, we only consider the minimum eigenvalue problem (7). After the spectral embedding step, the eigenvector $\mathbf{y}$ will be the low dimensional representation of original data points.

Recall the eigen-problem in LPI:

$$XLX^T\mathbf{a} = \lambda XDX^T\mathbf{a}. \tag{10}$$

The eigenvector $\mathbf{a}$ will be the mapping function and $X^T\mathbf{a}$ will be the low-dimensional representation of original data points. The LPI imposes a constraint on (7) that $\mathbf{y}$ should be the linear combination of the column vectors of X^T [15].

The dimensionality reduction method in traditional spectral clustering is nonlinear and the embedding result is only defined on "training" data points. The dimensionality

reduction method in our clustering algorithm is linear and the mapping function $\mathbf{a}$ can be applied to the unseen data which provides us more flexibility. When the data set is very large, to learn such an embedding is computationally expensive. Since the LPI is linear and defined everywhere, we can use part of the data to learn such an embedding. Our experiments will show this.

It is interesting to see in which condition the LPI result will be identical to the result of spectral dimensionality reduction in traditional spectral clustering. In this case, our clustering algorithm will be identical to the traditional spectral clustering [3].

Theorem 2. *If X is a full rank square matrix, the embedding result of LPI ($X^T\mathbf{a}$ in (10)) is identical to embedding result in spectral clustering ($\mathbf{y}$ in (7)) if the S is the same.*

Proof. Let $\mathbf{y} = X^T\mathbf{a}$. Equation (10) can be rewritten as follows:

$$XL\mathbf{y} = \lambda XD\mathbf{y}.$$

Since X is a full rank square matrix, the inverse of X always exists. Thus, the above equation can be changed to

$$X^{-1}XL\mathbf{y} = \lambda X^{-1}XD\mathbf{y}.$$

Finally, we get

$$L\mathbf{y} = \lambda D\mathbf{y}.$$

Thus, we have proven the embedding result of LPI ($X^T\mathbf{a}$ in (10)) is identical to embedding result of spectral clustering ($\mathbf{y}$ in (7)). □

In our algorithm, we use SVD projection in our data preprocessing step to remove those components corresponding to the zero singular value. If the rank of original term-document matrix X equals to the number of documents, the X will be a full rank square matrix after SVD projection. In document clustering, the number of terms is often larger than the number of documents, thus, if all the document vectors $\mathbf{x}$ are linearly independent, the X will be a full rank square matrix after SVD projection.

TABLE 2
Thirty Semantic Categories from TDT2 Used in Our Experiments

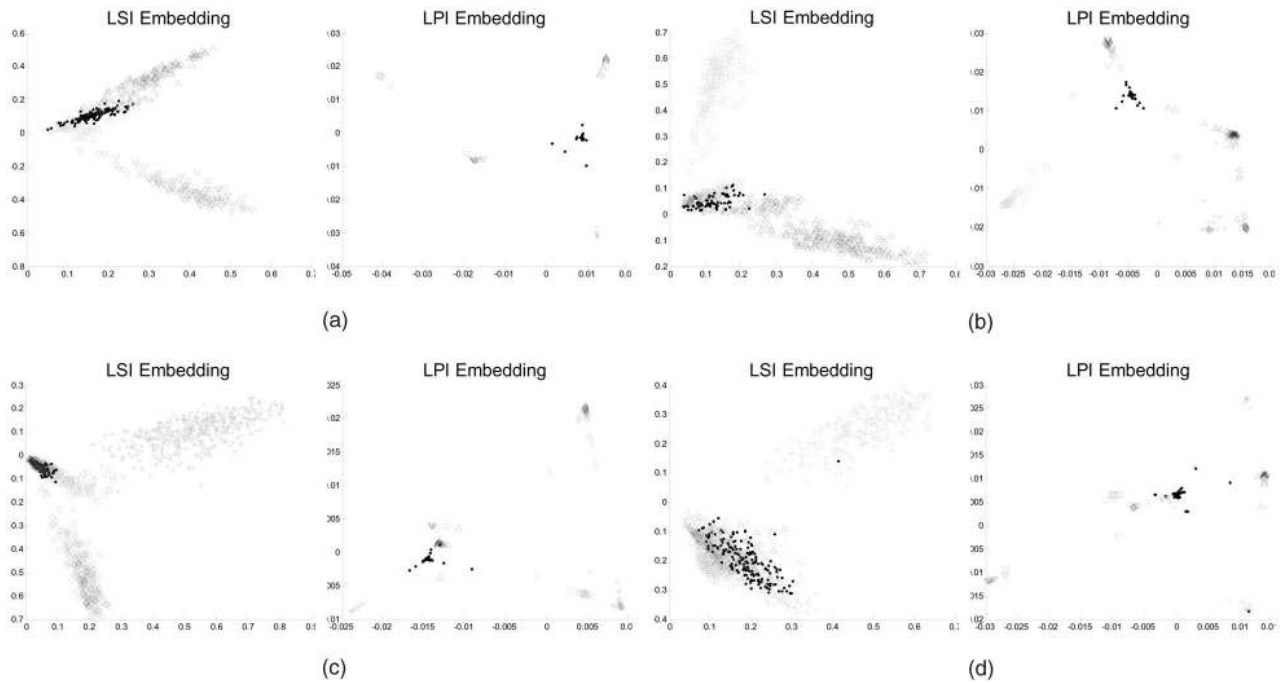| category	num of doc	category	num of doc	category	num of doc
20001	1844	20048	160	20096	76
20015	1828	20033	145	20021	74
20002	1222	20039	141	20026	72
20013	811	20086	140	20008	71
20070	441	20032	131	20056	66
20044	407	20047	123	20037	65
20076	272	20019	123	20065	63
20071	238	20077	120	20005	58
20012	226	20018	104	20074	56
20023	167	20087	98	20009	52

Fig. 1. Two-dimensional embedding results of LSI and LPI on the TDT2 corpus. The dimensions of document vectors are reduced to two using LSI and LPI. Each color (shape) represents a topic. As can be seen, LPI is more powerful than LSI as to separating the documents with different semantics. (a) Five classes. (b) Six classes. (c) Seven classes. (d) Eight classes.

5 EXPERIMENTAL RESULTS

In this section, several experiments were performed to show the effectiveness of our proposed algorithm. Two standard document collections were used in our experiments: Reuters-21578 and TDT2. We compared our proposed algorithm with clustering based on LSI, spectral clustering method, and Nonnegative Matrix Factorization clustering method [28].

5.1 Data Corpora

Reuters-21578 corpus[1] contains 21,578 documents in 135 categories. In our experiments, we discarded those

documents with multiple category labels, and selected the largest 30 categories. It left us with 8,067 documents in 30 categories as described in Table 1.

The TDT2 corpus[2] consists of data collected during the first half of 1998 and taken from six sources, including two newswires (APW, NYT), two radio programs (VOA, PRI), and two television programs (CNN, ABC). It consists of 11,201 on-topic documents which are classified into 96 semantic categories. In this data set, we also removed those documents appearing in two or more categories and used the largest 30 categories, thus leaving us with 9,394 documents in 30 categories as described in Table 2.

---

1. Reuters-21578 corpus is at http://www.daviddlewis.com/resources/testcollections/reuters21578/.

2. Nist Topic Detection and Tracking corpus is at http://www.nist.gov/speech/tests/tdt/tdt98/index.html.

TABLE 3
Performance Comparisons on TDT2 Corpus

k	Accuracy						
	Kmeans	LSI(best)	LSI	LPI(best)	LPI	LE	NMF-NCW
2	0.989	0.992	0.977	0.998	0.998	0.998	0.985
3	0.974	0.985	0.944	0.996	0.996	0.996	0.953
4	0.959	0.970	0.894	0.996	0.996	0.996	0.964
5	0.948	0.961	0.914	0.993	0.993	0.993	0.980
6	0.945	0.954	0.879	0.993	0.992	0.992	0.932
7	0.883	0.903	0.849	0.990	0.988	0.987	0.921
8	0.874	0.890	0.829	0.989	0.987	0.988	0.908
9	0.852	0.870	0.810	0.987	0.983	0.984	0.895
10	0.835	0.850	0.786	0.982	0.979	0.978	0.898
ave.	0.918	0.931	0.876	0.992	0.990	0.990	0.937
k	Mutual Information						
	Kmeans	LSI(best)	LSI	LPI(best)	LPI	LE	NMF-NCW
2	0.962	0.965	0.925	0.981	0.981	0.981	0.939
3	0.946	0.962	0.894	0.977	0.976	0.976	0.924
4	0.932	0.942	0.856	0.979	0.979	0.979	0.951
5	0.935	0.942	0.892	0.975	0.973	0.973	0.965
6	0.936	0.939	0.878	0.975	0.974	0.974	0.923
7	0.884	0.892	0.849	0.969	0.968	0.966	0.915
8	0.889	0.895	0.841	0.970	0.967	0.967	0.911
9	0.875	0.878	0.831	0.970	0.966	0.967	0.905
10	0.865	0.869	0.813	0.962	0.959	0.958	0.897
ave.	0.914	0.920	0.864	0.973	0.971	0.97	0.926

Each document is represented as a term-frequency vector. We simply removed the stop words and no further preprocessing was done. Each document vector is normalized to one and the Euclidean distance is used as the distance measure.

5.2 Evaluation Metric

The clustering performance is evaluated by comparing the obtained label of each document with that provided by the document corpus. Two metrics, the accuracy (AC) and the normalized mutual information metric ($\overline{MI}$), are used to measure the clustering performance [28]. Given a document x_i, let r_i and s_i be the obtained cluster label and the label provided by the corpus, respectively. The AC is defined as follows:

$$AC = \frac{\sum_{i=1}^{n} \delta(s_i, map(r_i))}{n},$$

where n is the total number of documents, $\delta(x, y)$ is the delta function that equals one if $x = y$ and equals zero otherwise, and $map(r_i)$ is the permutation mapping function that maps each cluster label r_i to the equivalent label

from the data corpus. The best mapping can be found by using the Kuhn-Munkres algorithm [19].

Let C denote the set of clusters obtained from the ground truth and C' obtained from our algorithm. Their mutual information metric $MI(C, C')$ is defined as follows:

$$MI(C, C') = \sum_{c_i \in C, c'_j \in C'} p(c_i, c'_j) \cdot log_2 \frac{p(c_i, c'_j)}{p(c_i) \cdot p(c'_j)},$$

where $p(c_i)$ and $p(c'_j)$ are the probabilities that a document arbitrarily selected from the corpus belongs to the clusters c_i and c'_j, respectively, and $p(c_i, c'_j)$ is the joint probability that the arbitrarily selected document belongs to the clusters c_i as well as c'_j at the same time. In our experiments, we use the normalized mutual information $\overline{MI}$ as follows:

$$\overline{MI}(C, C') = \frac{MI(C, C')}{\max(H(C), H(C'))},$$

where $H(C)$ and $H(C')$ are the entropies of C and C', respectively. It is easy to check that $\overline{MI}(C, C')$ ranges from 0 to 1. $\overline{MI} = 1$ if the two sets of clusters are identical, and $\overline{MI} = 0$ if the two sets are independent.

TABLE 4
Performance Comparisons on Reuters Corpus

k	Accuracy						
	Kmeans	LSI(best)	LSI	LPI(best)	LPI	LE	NMF-NCW
2	0.871	0.913	0.864	0.963	0.923	0.923	0.925
3	0.775	0.815	0.768	0.884	0.816	0.816	0.807
4	0.732	0.773	0.715	0.843	0.793	0.793	0.787
5	0.671	0.704	0.654	0.780	0.737	0.737	0.735
6	0.655	0.683	0.642	0.760	0.719	0.719	0.722
7	0.623	0.651	0.610	0.724	0.694	0.694	0.689
8	0.582	0.617	0.572	0.693	0.650	0.650	0.662
9	0.553	0.587	0.549	0.661	0.625	0.625	0.623
10	0.545	0.573	0.540	0.646	0.615	0.615	0.616
ave.	0.667	0.702	0.657	0.773	0.730	0.730	0.730
k	Mutual Information						
	Kmeans	LSI(best)	LSI	LPI(best)	LPI	LE	NMF-NCW
2	0.600	0.666	0.569	0.793	0.697	0.697	0.705
3	0.567	0.594	0.536	0.660	0.601	0.601	0.600
4	0.598	0.621	0.573	0.671	0.635	0.635	0.634
5	0.563	0.567	0.538	0.633	0.603	0.603	0.587
6	0.579	0.587	0.552	0.636	0.615	0.615	0.603
7	0.573	0.572	0.547	0.629	0.617	0.617	0.600
8	0.556	0.557	0.530	0.615	0.587	0.587	0.583
9	0.549	0.545	0.532	0.605	0.586	0.586	0.560
10	0.552	0.549	0.528	0.607	0.586	0.586	0.561
ave.	0.571	0.584	0.545	0.650	0.614	0.614	0.604

5.3 LPI Embedding versus LSI Embedding

Our theoretical analysis shows that LPI is able to map the documents related to the same semantics as close to each other as possible. This motivates us to perform traditional clustering in the LPI subspace rather than directly in the original space. In this section, we first present some embedding results by using LPI and LSI.

Fig. 1 shows the 2D embedding results on the TDT2 corpus. The experiments were conducted on 5, 6, 7, and 8 classes, respectively. The parameter p was set to 15 in LPI. As can be seen, LPI is more powerful than LSI as to separating the documents with different semantics.

5.4 Clustering Results

To demonstrate how our method improves the performance of document clustering, we compared five methods on two data sets: Reuters-21578 and TDT2. These five methods are listed below:

- k-means on original term-document matrix (Kmeans), which is treated as our baseline,
- k-means after LSI (LSI),
- k-means after LPI (LPI),
- Spectral Clustering (k-means after Laplacian Eigenmaps, or LE), and
- Nonnegative Matrix Factorization-based clustering (NMF-NCW, [28]).

Note that the two methods LPI and LE need to construct a graph on the documents. In the following experiments, we used the same graph for these two methods and the parameter p was set to 15. The weighted Nonnegative Matrix Factorization-based document clustering algorithm (NMF-NCW, [28]) is a recently proposed algorithm, which has shown to be very effective in document clustering. Please see [28] for details.

Tables 3 and 4 showed the experimental results on the TDT2 and the Reuters corpus, respectively. The evaluations were conducted with different number of clusters, ranging from two to 10. For each given cluster number k, 50 tests were conducted on different randomly chosen clusters and the average performance was computed over these 50 tests. For each test, the k-means algorithm was applied 10 times with different start points and the best result in terms of the objective function of k-means was recorded.

After LSI, LPI, or Laplacian Eigenmaps, how to determine the dimensions of the subspace is still an open problem. In
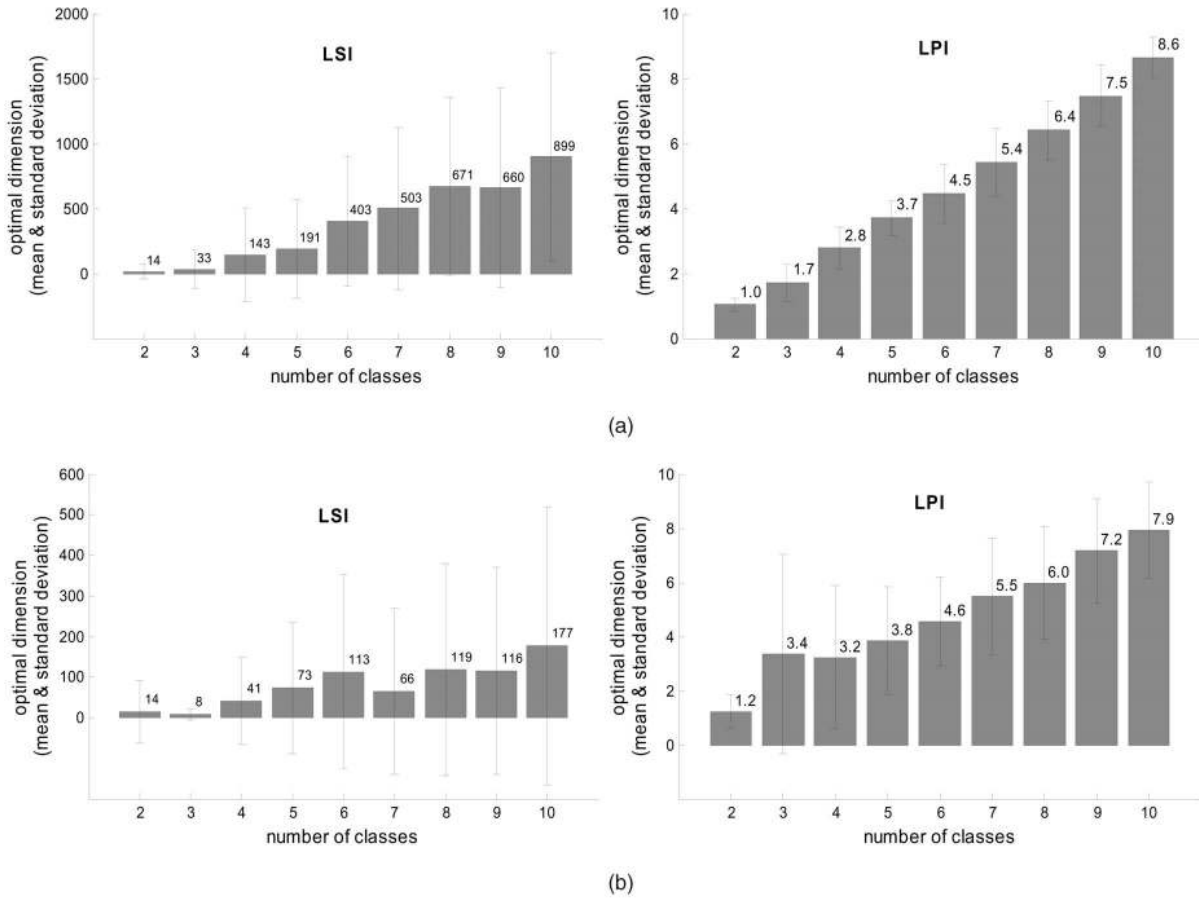
Fig. 2. Optimal dimension with different number of classes. Each bar shows the average of 50 test runs, the error bar indicates the standard deviation. (a) TDT2 corpus. (b) Reuters corpus.

the k cluster situation, we choose the first $k - 1$ dimensions in LPI based on our previous analysis. For Laplacian Eigenmaps, since the first eigenvector is **1**, we use the following $k - 1$ dimensions. Note that, in typical spectral clustering, the dimensions of the subspace are set to the number of clusters [21], which is the same with our selection in spirit (since [21] does not remove the first **1** eigenvector). For LSI, we choose the k dimensions for comparison. Besides such determined dimensions, for LSI and LPI, we also compute their best performance on different dimensions in each test. We iterate all the dimensions for the best clustering performance and average all the 50 best results. In a real situation, it might not be possible to iterate all the dimensions to get the best performance.

In Tables 3 and 4, LSI, LPI, and LE indicate this determined dimension while "LSI (best)" and "LPI (best)" are the best performance. Fig. 2 shows the optimal dimensions with different numbers of clusters by using LPI and LSI. The optimal dimension in LSI is much higher than LPI. Also, the variance of the dimensions obtained by using LSI is much higher than that obtained by using LPI. For LPI, the optimal number is nearly $k - 1$, where k is the number of clusters. This figure showed that LPI is more powerful than LSI in finding the intrinsic dimensionality of the document space. Thus, LPI is very suitable for clustering.

The experimental results showed that LSI seems not promising in dimension reduction for clustering because the k-means on the LSI subspace is even worse than k-means on the original document space. As can be seen, LPI

performed much better than LSI. We also see that the result of Laplacian Eigenmaps is nearly identical to the result of LPI. Actually, in our experiments, for 312 of 450 (50 × 9) tests on Reuters corpus and 430 of 450 (50 × 9) tests on TDT2 corpus, the X is full rank square matrix, thus, the results of clustering using LPI are identical to those of clustering using Laplacian Eigenmaps.

In [28], Xu et al. compared the NFM-NCW method with the spectral clustering method. In their comparison, they constructed the affinity matrix in spectral clustering as a complete graph. While, in our LSI and LE methods, the p-nearest neighbor graphs which put more focus on the local geometric document structure were used. More experiments on the different graph construction will be given in the next section.

5.5 Clustering Performance on Different Graph Model

The construction of the adjacency graph is one of the key points in our algorithm. Also, it is the key difference among LSI, LPI, and LDA. In this section, we test our algorithm under different adjacency graph constructions to see how the different graph structures will affect the clustering performance.

5.5.1 Local versus Global

In LPI clustering, one needs to set the number of nearest neighbors, i.e., the value of p, which defines the "locality." As we examined in Section 4.1, LPI approximates LSI when
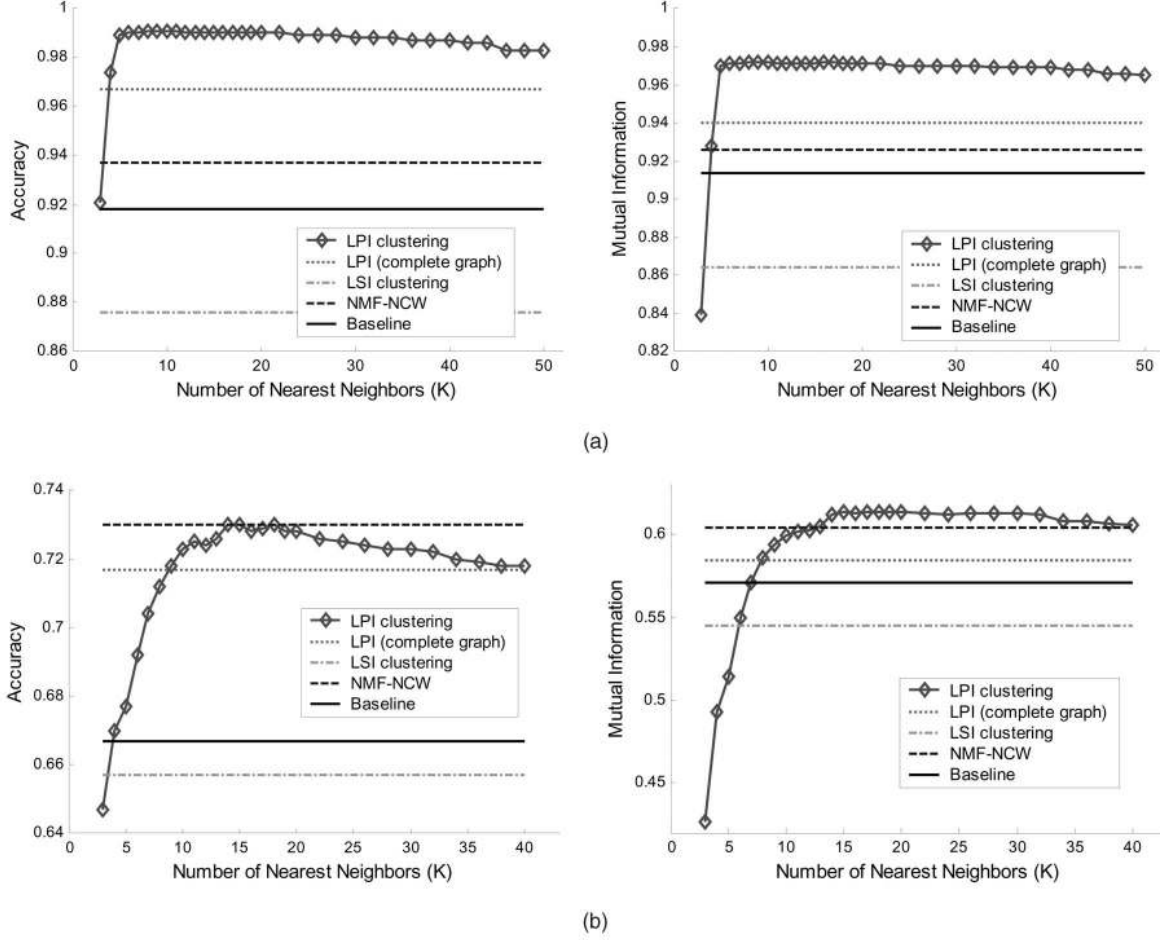
Fig. 3. Graph model—local versus global. (a) TDT2 corpus. (b) Reuters corpus.

p tends to be infinite (complete graph). In Fig. 3, we show the relationship between the clustering performance and the value of p. Here, the clustering performance is the average over $2 \sim 10$ classes. The value of p varies from 3 to 40. As can be seen, the performance of LPI clustering reaches its peak when p is 6 in TDT2 corpus and 15 in Reuters21578 corpus. As p increases, the performance decreases. This experiment shows that the local structure is more important than the global structure as to discovering the semantic structure of the document space.

In Fig. 3, we can also find that, even performed under complete graph, clustering after LPI is better than clustering after LSI. Our theoretical analysis in Section 4.1 shows that the only difference between LPI with complete graph and LSI is that LPI has the D matrix. Xu et al. [28] show a similar result that D can really improve the clustering result.

5.5.2 Gaussian versus Polynomial versus 0-1

Our algorithm uses the dot-product weighting in constructing the nearest neighbor graph. We can also use other kinds of weighting methods as described in Section 4.3. In this experiment, we examine the LPI clustering performance under different weighting choices.

We compare four kinds of weighting methods:

1. 0-1 weighting,
2. Gaussian kernel weighting, where the parameter t is set as 1,

3. dot-product weighting, and
4. polynomial kernel weighting with degrees 2 and 5.

The clustering results are shown in Table 5. We can see that LPI-based clustering is insensitive to the weighting function. Even the simplest one (0-1 weighting) can achieve the similar results. This tells us that the local document structure (p-neighbor graph) is essential in LPI-based clustering whereas the specific weighting values in the connected edges are not so influential.

5.6 Generalization Capability

Both LSI and LPI try to learn an optimal embedding function in the data set. In real applications, some new data might be registered into the data set. The performance on the new data can reflect the generalization capability of the algorithms. This generalization capability is very important for the clustering methods performed on the reduced dimensional space. In these clustering algorithms, learning the low-dimensional representation is time consuming and scales with the number of data points. LSI and LPI are linear and their mapping functions are defined everywhere. Thus, in large data set situations, LSI and LPI can choose part of the data to learn such mapping functions and map all the data points to the low-dimensional space which can speed up the whole process. It is hard for the spectral clustering to adopt such a technique since traditional spectral clustering

TABLE 5
Comparison Using Different Weighting Functions

	TDT2 Corpus									
	Accuracy					Mutual Information				
k	0-1	Gaussian	DotProd	Poly2	Poly5	0-1	Gaussian	DotProd	Poly2	Poly5
2	0.998	0.998	0.998	0.998	0.998	0.982	0.983	0.982	0.982	0.981
3	0.997	0.997	0.997	0.997	0.997	0.981	0.981	0.980	0.981	0.981
4	0.996	0.995	0.995	0.996	0.995	0.979	0.977	0.976	0.978	0.976
5	0.994	0.994	0.994	0.994	0.994	0.977	0.976	0.976	0.976	0.977
6	0.993	0.993	0.993	0.992	0.993	0.975	0.975	0.974	0.974	0.974
7	0.990	0.990	0.989	0.989	0.990	0.970	0.969	0.968	0.968	0.969
8	0.989	0.986	0.986	0.988	0.988	0.968	0.965	0.965	0.967	0.966
9	0.983	0.986	0.985	0.985	0.985	0.964	0.968	0.966	0.966	0.966
10	0.974	0.974	0.973	0.974	0.974	0.954	0.954	0.954	0.953	0.953
ave.	0.990	0.990	0.990	0.990	0.990	0.972	0.972	0.971	0.972	0.971
	Reuters Corpus									
	Accuracy					Mutual Information				
k	0-1	Gaussian	DotProd	Poly2	Poly5	0-1	Gaussian	DotProd	Poly2	Poly5
2	0.928	0.924	0.923	0.923	0.922	0.715	0.703	0.697	0.697	0.693
3	0.821	0.815	0.816	0.814	0.812	0.613	0.604	0.601	0.607	0.599
4	0.809	0.791	0.793	0.796	0.790	0.644	0.631	0.635	0.636	0.625
5	0.735	0.733	0.737	0.733	0.732	0.604	0.603	0.603	0.603	0.600
6	0.716	0.713	0.719	0.715	0.715	0.616	0.613	0.615	0.613	0.609
7	0.697	0.698	0.694	0.700	0.695	0.624	0.618	0.617	0.621	0.617
8	0.649	0.650	0.650	0.647	0.649	0.587	0.585	0.587	0.584	0.588
9	0.617	0.619	0.625	0.617	0.632	0.579	0.582	0.586	0.580	0.590
10	0.611	0.613	0.615	0.612	0.612	0.583	0.584	0.586	0.584	0.584
ave.	0.731	0.728	0.730	0.728	0.729	0.618	0.614	0.614	0.614	0.612

uses nonlinear embedding methods, which is only defined on "training" samples.

To compare the generalization capability of LSI and LPI, we designed the following experiment: In each test in the previous experiment, we only chose part of the data to learn the embedding function, embedded the whole data set, and then performed clustering in the subspace. The size of the training set ranged from 5 percent to 90 percent of the data set. For each case, 10 times of random selection were performed to minimize the impact of occasionality.

The average accuracy (averaged over $2 \sim 10$ classes) and normalized mutual information (averaged over $2 \sim 10$ classes) are shown in Fig. 4. It is clear that the performance improves with the number of training samples. Both LSI and LPI have good generalization capability, however, the performance of LSI is always below the baseline which makes LSI less practical. For LPI, it achieved similar performance to that using all the samples when only 30 percent of training samples were used. This makes it practical for clustering large sets of documents.

5.7 Discussions

We summarize the experiments below:

1. In document clustering, dimension reduction can make significant improvement (the best performances obtained by both LSI and LPI are better than the baseline). However, in the real world, one cannot test all the possibilities to find the optimal dimensionality. Therefore, LSI is less practical since it is hard to estimate the optimal dimensionality. In contrast to LSI, clustering after LPI might be a better choice. It is easy for LPI to estimate the optimal dimension and the performance is always better than baseline.

2. In dimension reduction for clustering, the local geometric structure is more important than the global structure. Based on the assumption that neighboring points probably belong to the same underlying class, the p-nearest graph in our LPI algorithm gives an optimal approximation to the labeled similarity matrix W in (5). Thus, even in an unsupervised mode, LPI has discriminating power to some extent which provides a better low-dimensional representation for clustering.
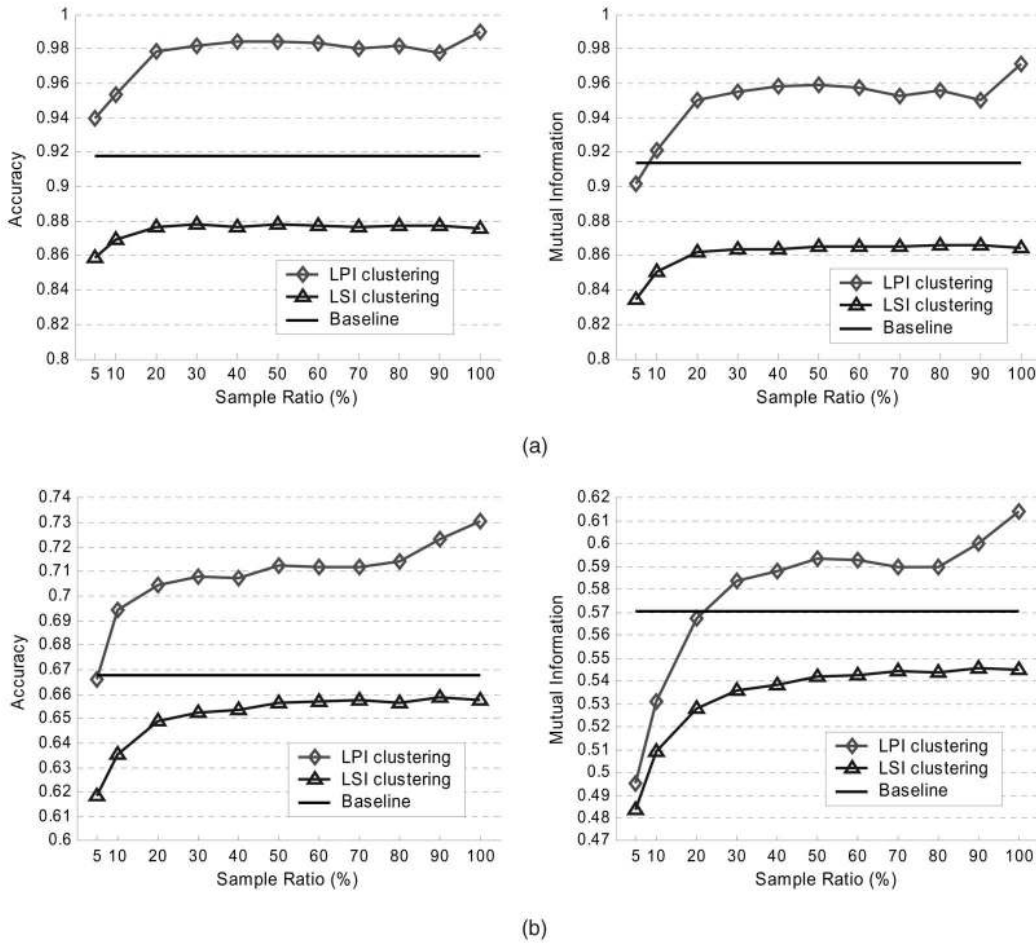
Fig. 4. Generalization capability of LSI and LPI. (a) TDT2 corpus. (b) Reuters corpus.

3. In document clustering, clustering after LPI is a good linear approximation to spectral clustering. The experiments on generalization capability showed that LPI clustering might be more applicable than spectral clustering in that LPI is linear and is defined everywhere rather than just on the training data points.

6 CONCLUSIONS

A novel document clustering algorithm based on Locality Preserving Indexing is proposed in this paper. Based on the analysis of the computational properties of LPI, we presented the modified LPI algorithm for document clustering. Analysis on the relationship among LSI, LPI, and LDA indicates that the affinity graph is the key to distinguish these algorithms. The p-nearest neighbor graph makes LPI approximate to LDA, which is supervised. A complete graph makes LPI similar to LSI. Extensive experiments on Reuters-21578 and TDT2 showed that our algorithm performed much better than the LSI-based clustering algorithm and close to the traditional spectral clustering algorithm. Moreover, the linearity of LPI makes our clustering algorithm more applicable than spectral clustering when the data set is large.

Several questions remain to be investigated in our future work:

1. In this paper, we gave some empirical estimation on the dimensionality using LPI. However, it lacks a strong theoretical foundation. How to estimate the dimensionality in theory remains to be investigated. Also, it remains unclear how to estimate the number of topics hidden in the document set. These two problems seem to be two sides of a coin in that the dimensionality can be inferred from the number of topics as suggested in our experiments.
2. As shown in Section 4, LPI provides an optimal approximation to LDA which is supervised. Specifically, we approximate the graph defined in (5) by a p-nearest neighbor graph. The more accurate the approximation is, the more discriminating power our algorithm has. However, it is unclear if there exists a better approximation and how to obtain it.
3. Clustering is inherently an unsupervised learning process. In all of our experiments, the data points are unlabeled. However, sometimes a small set of labeled data points might be available. In such a case, the unsupervised learning becomes semiunsupervised (or, semisupervised) learning. Thus, the optimal projection can be obtained by preserving locality as well as separating the data points with

different labels. It is unclear how to obtain such constrained LPI, though it seems to be promising.

REFERENCES

[1] L. Baker and A. McCallum, "Distributional Clustering of Words for Text Classification," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 96-103, Aug. 1998.

[2] B.T. Bartell, G.W. Cottrell, and R.K. Belew, "Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 161-167, June 1992.

[3] M. Belkin and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," *Advances in Neural Information Processing Systems 14,* pp. 585-591, Cambridge, Mass.: MIT Press, 2001.

[4] P.K. Chan, D.F. Schlag, and J.Y. Zien, "Spectral k-Way Ratio-Cut Partitioning and Clustering," *IEEE Trans. Computer-Aided Design,* vol. 13, pp. 1088-1096, 1994.

[5] F.R.K. Chung, *Spectral Graph Theory.* Am. Math. Soc., 1997.

[6] David Cohn, "Informed Projections," *Advances in Neural Information Processing Systems 15,* pp. 849-856, Cambridge, Mass: MIT Press, 2002.

[7] S.C. Deerwester, S.T. Dumais, T.K. Landauer, G.W. Furnas, and R.A. Harshman, "Indexing by Latent Semantic Analysis," *J. Am. Soc. Information Science,* vol. 41, no. 6, pp. 391-407, 1990.

[8] C. Ding, X. He, H. Zha, M. Gu, and H.D. Simon, "A Min-Max Cut Algorithm for Graph Partitioning and Data Clustering," *Proc. Int'l Conf. Data Mining,* pp. 107-114, Nov. 2001.

[9] C.H. Ding, "A Similarity-Based Probability Model for Latent Semantic Indexing," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 58-65, Aug. 1999.

[10] R.O. Duda, P.E. Hart, and D.G. Stork, *Pattern Classification,* second ed. Hoboken, N.J.: Wiley-Interscience, 2000.

[11] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proc. Int'l Conf. Knowledge Discovery and Data Mining,* pp. 226-231, Aug. 1996.

[12] K. Funkunaga and P. Navendra, "A Branch and Bound Algorithm for Computing k-Nearest Neighbors," *IEEE Trans. Computers,* vol. 24, no. 7, pp. 750-753, 1975.

[13] G.H. Golub and C.F. Van Loan, *Matrix Computations,* third ed. Johns Hopkins Univ. Press, 1996.

[14] X. He, D. Cai, H. Liu, and W.-Y. Ma, "Locality Preserving Indexing for Document Representation," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 96-103, July 2004.

[15] X. He and P. Niyogi, "Locality Preserving Projections," *Advances in Neural Information Processing Systems 16,* Cambridge, Mass.: MIT Press, 2003.

[16] A.K. Jain and R.C. Dubes, *Algorithms for Clustering Data.* Upper Saddle River, N.J.: Prentice-Hall, Inc., 1988.

[17] T. Li, S. Ma, and M. Ogihara, "Document Clustering via Adaptive Subspace Iteration," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 218-225, July 2004.

[18] X. Liu, Y. Gong, W. Xu, and S. Zhu, "Document Clustering with Cluster Refinement and Model Selection Capabilities," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 191-198, Aug. 2002.

[19] L. Lovasz and M. Plummer, *Matching Theory.* North Holland, Budapest: Akadémiai Kiadó, 1986.

[20] J. Mcqueen, "Some Methods for Classification and Analysis of Multivariate Observations," *Proc. Fifth Berkeley Symp. Math. Statistics and Probability,* pp. 281-297, 1967.

[21] A.Y. Ng, M. Jordan, and Y. Weiss, "On Spectral Clustering: Analysis and an Algorithm," *Advances in Neural Information Processing Systems 14,* pp. 849-856, Cambridge, Mass.: MIT Press, 2001.

[22] C.H. Papadimitriou, P. Raghavan, H. Tamaki, and S. Vempala, "Latent Semantic Indexing: A Probabilistic Analysis," *Proc. 17th ACM Symp. Principles of Database Systems,* June 1998.

[23] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 22, no. 8, pp. 888-905, 2000.

[24] S. Siersdorfer and S. Sizov, "Restrictive Clustering and Metaclustering for Self-Organizing Document Collections," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 226-233, July 2004.

[25] N. Srebro and T. Jaakkola, "Linear Dependent Dimensionality Reduction," *Advances in Neural Information Processing Systems 16,* Cambridge, Mass.: MIT Press, 2003.

[26] Y. Weiss, "Segmentation Using Eigenvectors: A Unifying View," *Proc. Int'l Conf. Computer Vision,* pp. 975-982, Sept. 1999.

[27] W. Xu and Y. Gong, "Document Clustering by Concept Factorization," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 202-209, July 2004.

[28] W. Xu, X. Liu, and Y. Gong, "Document Clustering Based on Non-Negative Matrix Factorization," *Proc. Int'l Conf. Research and Development in Information Retrieval,* pp. 267-273, Aug. 2003.

[29] H. Zha, C. Ding, M. Gu, X. He, and H. Simon, "Spectral Relaxation for k-Means Clustering," *Advances in Neural Information Processing Systems 14,* pp. 1057-1064, Cambridge, Mass.: MIT Press, 2001.

[30] B. Zhang and S.N. Srihari, "A Fast Algorithm for Finding k-Nearest Neighbors with Non-Metric Dissimilarity," *Proc. Eighth Int'l Workshop Frontiers in Handwriting Recognition,* 2002.

Deng Cai received the BEng and MEng degrees, both in automation, from Tsinghua University, China, in 2000 and 2003, respectively. He is currently working toward the PhD degree in the Department of Computer Science, University of Illinois at Urbana Champaign. His research interests are machine learning, data mining, information retrieval, and computer vision.

Xiaofei He received the BS degree in computer science from Zhejiang University, China, in 2000. He is currently working toward the PhD degree in the Department of Computer Science, The University of Chicago. His research interests are machine learning, information retrieval, and computer vision.

Jiawei Han received the PhD degree in computer science from the University of Wisconsin in 1985. He is a professor in the Department of Computer Science at the University of Illinois at Urbana Champaign. Previously, he was an Endowed University Professor at Simon Fraser University, Canada. He has been working on research in data mining, data warehousing, stream data mining, spatial and multimedia data mining, biological data mining, social network analysis, text and Web mining, and software bug mining, with more than 300 conference and journal publications. He has chaired or served in many program committees of international conferences and workshops, including ACM SIGKDD conferences (2001 best paper award chair, 1996 PC cochair), SIAM-Data Mining conferences (2001 and 2002 PC cochair), ACM SIGMOD conferences (2000 exhibit program chair), ICDE conferences (2004, 2002, and 1995 PC vice chair), and ICDM conferences (2005 PC cochair). He also served or is serving on the editorial boards for *Data Mining and Knowledge Discovery*, *IEEE Transactions on Knowledge and Data Engineering*, the *Journal of Intelligent Information Systems*, and the *Journal of Computer Science and Technology*. He is currently serving on the board of directors for the executive committee of the ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD). Dr. Han has received three IBM Faculty Awards, the Outstanding Contribution Award at ICDM (2002), the ACM Service Award (1999), the ACM Fellow (2004), the ACM SIGKDD Innovations Award (2004), and the IEEE Computer Society 2004 Technical Achievement Award (2005). He is the first author of the textbook *Data Mining: Concepts and Techniques* (Morgan Kaufmann, 2001). He is a senior member of the IEEE.