Tensor Embedding Methods

Guang Dai & Dit-Yan Yeung
Department of Computer Science,
Hong Kong University of Science and Technology,
Clear Water Bay, Kowloon, Hong Kong
{daiguang,dyyeung}@cs.ust.hk

Abstract

Over the past few years, some embedding methods have been proposed for feature extraction and dimensionality reduction in various machine learning and pattern classification tasks. Among the methods proposed are Neighborhood Preserving Embedding (NPE), Locality Preserving Projection (LPP) and Local Discriminant Embedding (LDE) which have been used in such applications as face recognition and image/video retrieval. However, although the data in these applications are more naturally represented as higher-order tensors, the embedding methods can only work with vectorized data representations which may not capture well some useful information in the original data. Moreover, high-dimensional vectorized representations also suffer from the curse of dimensionality and the high computational demand. In this paper, we propose some novel tensor embedding methods which, unlike previous methods, take data directly in the form of tensors of arbitrary order as input. These methods allow the relationships between dimensions of a tensor representation to be efficiently characterized. Moreover, they also allow the intrinsic local geometric and topological properties of the manifold embedded in a tensor space to be naturally estimated. Furthermore, they do not suffer from the curse of dimensionality and the high computational demand. We demonstrate the effectiveness of the proposed tensor embedding methods on a face recognition application and compare them with some previous methods. Extensive experiments show that our methods are not only more effective but also more efficient.

Introduction

Feature extraction and dimensionality reduction are among the most fundamental problems studied in several related areas, including machine learning, pattern recognition, data mining, and computer vision. Over the past few decades, the most representative methods for feature extraction and dimensionality reduction are Principal Component Analysis (PCA) and Linear Discriminant Analysis (LDA) for the unsupervised and supervised learning settings, respectively. However, it is well known that both methods are optimal only if the underlying data distribution is Gaussian (Duda, Hart, & Stork 2000). While PCA and LDA are linear dimensionality reduction methods, nonlinear extensions based on the kernel approach (Schölkopf & Smola 2002) have also been proposed to give nonlinear methods called kernel PCA and kernel Fisher discriminant analysis.

Recently, there are growing interests in exploiting the intrinsic manifold structure of the data for achieving dimensionality reduction. When there exists nonlinearity in the manifold structure, Euclidean distance is incapable of characterizing the geometric structure of the data and hence traditional linear methods like PCA and LDA no longer work well. Some recent nonlinear dimensionality reduction (or called manifold learning) methods, including Isomap (Tenenbaum, Silva, & Langford 2000), Locally Linear Embedding (LLE) (Roweis & Saul 2000), and Laplacian Eigenmap (Belkin & Niyogi 2003), can preserve the local or global geometric properties of the nonlinear manifold structure. However, these methods in their original form only work on a given set of data points but cannot be used for embedding new data points. (Bengio *et al.* 2004) proposed an approach based on the Nyström formula for achieving out-of-sample extension in manifold learning. However, their approach which makes use of data-dependent kernels is in general computationally more demanding. More recently, Neighborhood Preserving Embedding (NPE) (He *et al.* 2005) and Locality Preserving Projection (LPP) (He & Niyogi 2004) were proposed as optimal linear approximations to LLE and Laplacian Eigenmap, respectively. Since both methods define the embedding everywhere in the ambient space but not just on a given set of data points, extension to new data points is straightforward and efficient. Moreover, by integrating label information and neighborhood information, Local Discriminant Embedding (LDE) (Chen, Chang, & Liu 2005) was proposed as a manifold-based embedding method for classification.

These embedding methods assume that the data are in vectorized representations. In many real-world applications, however, the data are more naturally represented as higher-order tensors. These tensors have to be unfolded into one-dimensional vectors first before the embedding methods can be applied. In so doing, some useful information in the original data may not be captured well. Moreover, high-dimensional vectorized representations also suffer from the curse of dimensionality and the high computational demand. Recently, extensions were proposed to the original PCA, LDA and LPP to allow these methods to work directly on two-dimensional (2D) matrices rather than one-dimensional vectors (Yang *et al.* 2004; Ye, Janardan, & Li 2005; He, Cai, & Niyogi 2006). Some attempts have also been made to take into consideration higher-order tensor spaces (Wang *et al.* 2005; Xu *et al.* 2005; Vasilescu & Terzopoulos 2003), but they are only defined based on the Euclidean distance metric.

In this paper, we propose three tensor embedding methods, called Tensor Neighborhood Preserving Embedding (TNPE), Tensor Locality Preserving Projection (TLPP) and Tensor Local Discriminant Embedding (TLDE), based on multilinear algebra and differential geometry. Unlike previous methods (He *et al.* 2005; He & Niyogi 2004; Chen, Chang, & Liu 2005), our methods take data directly in the form of tensors of arbitrary order as input. With the data points sampled randomly from the manifold embedded in an arbitrary tensor space $\mathbb{R}^{I_1\times\cdots\times I_k}$, the tensor embedding methods find a set of transformation matrices for defining the embedded tensor subspaces that together give an optimal approximation to the manifold preserving some local geometric and topological properties.

In summary, our tensor embedding methods possess the following appealing properties. (1) While NPE, LPP and LDE can only work with vectorized data representations, TNPE, TLPP and TLDE can work with more general tensor representations of arbitrary order which include vectorized data representations as special cases. (2) Both TNPE and TLPP may be applied in a semi-supervised or supervised manner by incorporating the label information into the neighborhood graphs. (3) The curse of dimensionality problem is effectively avoided and the computational complexity is significantly reduced. (4) The solutions can be obtained by solving generalized eigenvalue problems in an iterative manner. (5) Our embedding methods explicitly discover the intrinsic local geometric structure of the manifold in an arbitrary tensor space, and the low-dimensional embedding can be found by preserving the neighborhood structure. (6) Similar to NPE, LPP and LDE, our embedding methods define the embedding everywhere in the ambient space but not just on a given set of data points.

Brief Review of NPE, LPP and LDE

Given a set of n data points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^m$, NPE, LPP and LDE seek a transformation matrix that maps each data point $\mathbf{x}_i$ to a corresponding lower-dimensional data point $\mathbf{y}_i$. Different criteria are used in different methods for finding the transformation matrix.

NPE can be regarded as a linear approximation to LLE. Unlike traditional PCA which focuses on preserving the global Euclidean structure, NPE attempts to preserve the local neighborhood structure of the manifold. Moreover, unlike manifold learning methods such as Isomap, LLE and Laplacian Eigenmap, NPE can be directly applied to any data point including out-of-sample points. Similar to LLE, NPE first constructs a neighborhood graph for $\mathcal{X}$ based on either the K-nearest-neighbor or ϵ-neighborhood criterion. It then computes the affinity matrix $\mathbf{S} = [s_{ij}]_{n\times n}$ which is normalized such that each row of $\mathbf{S}$ sums to one. To obtain a linear approximation to LLE, the optimization problem for NPE is given by:

$$\arg\min_{\mathbf{p}} \sum_i (\mathbf{p}^T\mathbf{x}_i - \sum_j s_{ij}\mathbf{p}^T\mathbf{x}_j)^2 = \mathbf{p}^T\mathbf{X}\mathbf{M}\mathbf{X}^T\mathbf{p}, \qquad (1)$$
$$\text{s.t.} \quad \mathbf{p}^T\mathbf{X}\mathbf{X}^T\mathbf{p} = 1,$$

where $\mathbf{p}$ is the transformation vector, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]$, $\mathbf{M} = (\mathbf{I} - \mathbf{S})^T(\mathbf{I} - \mathbf{S})$, and $\mathbf{I}$ is the $n \times n$ identity matrix.

Similar to NPE, LPP is an optimal linear approximation to Laplacian Eigenmap. Given $\mathcal{X}$, it also first constructs a neighborhood graph and then computes the affinity matrix $\mathbf{S}$. The optimization problem for LPP is given by:

$$\arg\min_{\mathbf{p}} \sum_{ij} (\mathbf{p}^T\mathbf{x}_i - \mathbf{p}^T\mathbf{x}_j)^2 s_{ij} = \mathbf{p}^T\mathbf{X}\mathbf{L}\mathbf{X}^T\mathbf{p}, \qquad (2)$$
$$\text{s.t.} \quad \mathbf{p}^T\mathbf{X}\mathbf{D}\mathbf{X}^T\mathbf{p} = 1,$$

where $\mathbf{L} = \mathbf{D} - \mathbf{S}$ is the graph Laplacian and $\mathbf{D}$ is a diagonal matrix with the diagonal entries $d_{ii} = \sum_j s_{ij}$.

LDE is explicitly formulated for classification problems. Based on the assumption that any subset of data points belonging to the same class lies in a submanifold, LDE seeks to find an optimal transformation matrix by integrating the class label information of the data points and the neighborhood information between data points. The goal is to preserve the within-class neighborhood relationship while dissociating the submanifolds for different classes from each other. The within-class neighborhood relationship is represented by a within-class neighborhood graph G while the between-class neighborhood relationship is represented by a between-class neighborhood graph G'. We then define the affinity matrices $\mathbf{S} = [s_{ij}]_{n\times n}$ and $\mathbf{S}' = [s'_{ij}]_{n\times n}$ for G and G', respectively. The optimization problem for LDE is defined as:

$$\arg\min_{\mathbf{p}} \sum_{ij} (\mathbf{p}^T\mathbf{x}_i - \mathbf{p}^T\mathbf{x}_j)^2 s_{ij}, \qquad (3)$$
$$\text{s.t.} \quad \sum_{ij} (\mathbf{p}^T\mathbf{x}_i - \mathbf{p}^T\mathbf{x}_j)^2 s'_{ij} = 1.$$

Note that the original optimization problem formulated in (Chen, Chang, & Liu 2005) is a maximization problem, which is equivalent to the minimization problem given here. Our preference for the latter is mainly for consistency with the optimization problems for NPE and LPP.

Our Tensor Embedding Methods

Basic Terminology on Tensor Operations

Before devising our tensor embedding methods, we first review some basic terminology on tensor operations.

Let $\mathcal{A}$ be a tensor of size $I_1 \times \cdots \times I_k$. The order of $\mathcal{A}$ is k and the fth dimension (or mode) of $\mathcal{A}$ is of size I_f. In addition, we denote the index of a single entry within a tensor by subscripts, such as $\mathcal{A}_{i_1\ldots i_k}$.

Definition 1 *The scalar product $\langle\mathcal{A},\mathcal{B}\rangle$ of two tensors $\mathcal{A},\mathcal{B} \in \mathbb{R}^{I_1\times\cdots\times I_k}$ is defined as $\langle\mathcal{A},\mathcal{B}\rangle \overset{def}{=} \sum_{i_1}\cdots\sum_{i_k}\mathcal{A}_{i_1\ldots i_k}\mathcal{B}^*_{i_1\ldots i_k}$, where $*$ denotes complex conjugation. Furthermore, the Frobenius norm of a tensor $\mathcal{A}$ is defined as $\|\mathcal{A}\|_F \overset{def}{=} \sqrt{\langle\mathcal{A},\mathcal{A}\rangle}$.*

Definition 2 *The f-mode product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1\times\cdots\times I_k}$ and a matrix $\mathbf{U} \in \mathbb{R}^{J_f\times I_f}$ is an $I_1 \times \cdots \times I_{f-1} \times J_f \times I_{f+1} \times \cdots \times I_k$ tensor denoted as $\mathcal{A} \times_f \mathbf{U}$, where the corresponding entries are given by $(\mathcal{A} \times_f \mathbf{U})_{i_1\ldots i_{f-1}j_f i_{f+1}\ldots i_k} \overset{def}{=} \sum_{i_f}\mathcal{A}_{i_1\ldots i_{f-1}i_f i_{f+1}\ldots i_k}\mathbf{U}_{j_f i_f}$.*

Definition 3 *Let $\mathcal{A}$ be an $I_1 \times \cdots \times I_k$ tensor and $(\pi_1, \ldots, \pi_{k-1})$ be any permutation of the entries of the set $\{1, \ldots, f-1, f+1, \ldots, k\}$. The f-mode unfolding of the tensor $\mathcal{A}$ into an $I_f \times \Pi_{l=1}^{k-1} I_{\pi_l}$ matrix, denoted as $\mathbf{A}^{(f)}$, is defined by $\mathcal{A} \in \mathbb{R}^{I_1 \times \cdots \times I_k} \Rightarrow_f \mathbf{A}^{(f)} \in \mathbb{R}^{I_f \times \Pi_{l=1}^{k-1} I_{\pi_l}}$, where $\mathbf{A}^{(f)}_{i_f j} = \mathcal{A}_{i_1 \ldots i_k}$ with $j = 1 + \sum_{l=1}^{k-1}(i_{\pi_l} - 1)\prod_{\hat{l}=1}^{l-1} I_{\pi_{\hat{l}}}$.*

In general, the goal of linear dimensionality reduction in a tensor space can be described as follows. Given n data points $\mathcal{A}_1, \ldots, \mathcal{A}_n$ in the tensor space $\mathbb{R}^{I_1 \times \cdots \times I_k}$, a linear tensor embedding method seeks to find k transformation matrices $\mathbf{U}_i \in \mathbb{R}^{l_i \times I_i}$ $(l_i < I_i, i = 1, \ldots, k)$ such that n corresponding embedded data points $\mathcal{B}_1, \ldots, \mathcal{B}_n \in \mathbb{R}^{l_1 \times \cdots \times l_k}$ can be obtained as $\mathcal{B}_i = \mathcal{A}_i \times_1 \mathbf{U}_1 \times_2 \cdots \times_k \mathbf{U}_k$ $(i = 1, \ldots, n)$.

Tensor Neighborhood Preserving Embedding

Given n data points $\mathcal{A}_1, \ldots, \mathcal{A}_n$ from an unknown manifold $\mathcal{M}$ embedded in a tensor space $\mathbb{R}^{I_1 \times \cdots \times I_k}$, TNPE finds k optimal projections $\mathbf{U}_i \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$ such that the local topological structure of $\mathcal{M}$ is preserved and the intrinsic geometric property is effectively captured. Our formulation of TNPE is essentially based on that of NPE.

We construct a neighborhood graph G to capture the intrinsic geometric structure of $\mathcal{M}$ and apply the heat kernel to define the affinity matrix $\mathbf{S} = [s_{ij}]_{n \times n}$ as:

$$s_{ij} = \begin{cases} \exp(-\|\mathcal{A}_i - \mathcal{A}_j\|_F^2/t), & \text{if } \mathcal{A}_j \in O(K, \mathcal{A}_i); \\ 0, & \text{otherwise.} \end{cases}$$

where $O(K, \mathcal{A}_i)$ denotes the set of K nearest neighbors of $\mathcal{A}_i$ and t is a positive constant. Alternatively, G may be constructed based on the ϵ-neighborhood. $\mathbf{S}$ is normalized such that each row sums to one. In case label information is partially or fully available, it can be incorporated into the neighborhood graph so that embedding can be done in a semi-supervised or supervised manner. Based on the k transformation matrices $\mathbf{U}_i \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$, let $\mathcal{B}_i$ denote the point in the embedded tensor space corresponding to $\mathcal{A}_i$. In order to preserve the local structure explicitly, we define the following objective function based on the Frobenius norm of a tensor:

$$\arg\min \; Q(\mathbf{U}_1, \ldots, \mathbf{U}_k) = \sum_i \|\mathcal{B}_i - \sum_j s_{ij}\mathcal{B}_j\|_F^2 \quad (4)$$

$$= \sum_i \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k - \sum_j s_{ij}\mathcal{A}_j \times_1 \cdots \times_k \mathbf{U}_k\|_F^2.$$

To eliminate an arbitrary scaling factor in the transformation matrices, we impose the following constraint: $\sum_i \|\mathcal{B}_i\|_F^2 = \sum_i \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 = 1$. Hence the optimization problem for TNPE can be expressed as:

$$\arg\min \; Q(\mathbf{U}_1, \ldots, \mathbf{U}_k) \quad (5)$$

$$= \sum_i \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k - \sum_j s_{ij}\mathcal{A}_j \times_1 \cdots \times_k \mathbf{U}_k\|_F^2,$$

$$\text{s.t.} \quad \sum_i \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 = 1.$$

Note that this optimization problem is a high-order nonlinear programming problem with a high-order nonlinear constraint, making direct computation of the transformation matrices infeasible. In general, this type of problems can be solved approximately by employing an iterative scheme which was originally proposed for low-rank approximation of second-order tensors (Ye 2004) and later extended for higher-order tensors (Wang *et al.* 2005). In what follows, we will discuss how to solve the problem in (5) by such an iterative scheme. Assuming that $\mathbf{U}_1, \ldots, \mathbf{U}_{f-1}, \mathbf{U}_{f+1}, \ldots, \mathbf{U}_k$ are known, we denote $\mathcal{A}_i \times_1 \mathbf{U}_1 \cdots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \cdots \times_k \mathbf{U}_k$ by the tensor $\mathcal{Y}_i^f$. Then, by the corresponding f-mode unfolding, we have $\mathcal{Y}_i^f \Rightarrow_f \mathbf{Y}_i^{(f)}$. In addition, we have $\mathcal{Y}_i^f \times_f \mathbf{U}_f = \mathbf{U}_f \mathbf{Y}_i^{(f)}$. From the property of trace, we have $\mathrm{tr}(\mathbf{H}\mathbf{H}^T) = \|\mathbf{H}\|_F^2$ for any matrix $\mathbf{H}$. The objective function and the constraint in (5) can thus be rewritten in the following alternative form based on trace:
$P_f(\mathbf{U}_f) = \sum_i \|\mathcal{Y}_i^f \times_f \mathbf{U}_f - \sum_j s_{ij}\mathcal{Y}_j^f \times_f \mathbf{U}_f\|_F^2 = \sum_i \|\mathbf{U}_f\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{U}_f\mathbf{Y}_j^{(f)}\|_F^2 = \sum_i \mathrm{tr}\{\mathbf{U}_f(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})^T\mathbf{U}_f^T\} = \mathrm{tr}\{\mathbf{U}_f\big(\sum_i(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})^T\big)\mathbf{U}_f^T\}$ and $\sum_i \|\mathcal{Y}_i^f \times_f \mathbf{U}_f\|_F^2 = \sum_i \|\mathbf{U}_f\mathbf{Y}_i^{(f)}\|_F^2 = \sum_i \mathrm{tr}\{\mathbf{U}_f\mathbf{Y}_i^{(f)}\mathbf{Y}_i^{(f)T}\mathbf{U}_f^T\} = \mathrm{tr}\{\mathbf{U}_f\big(\sum_i \mathbf{Y}_i^{(f)}\mathbf{Y}_i^{(f)T}\big)\mathbf{U}_f^T\}$.

Thus, the optimization problem in (5) can be reformulated as:

$$\arg\min \; P_f(\mathbf{U}_f) = \quad (6)$$

$$\mathrm{tr}\Big\{\mathbf{U}_f\Big(\sum_i(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})^T\Big)\mathbf{U}_f^T\Big\},$$

$$\text{s.t.} \quad \mathrm{tr}\Big\{\mathbf{U}_f\Big(\sum_i \mathbf{Y}_i^{(f)}\mathbf{Y}_i^{(f)T}\Big)\mathbf{U}_f^T\Big\} = 1.$$

The unknown transformation matrix $\mathbf{U}_f$ can be found by solving for the eigenvectors corresponding to the l_f smallest eigenvalues in the generalized eigenvalue equation $\big(\sum_i(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \sum_j s_{ij}\mathbf{Y}_j^{(f)})^T\big)\mathbf{u} = \lambda\big(\sum_i \mathbf{Y}_i^{(f)}\mathbf{Y}_i^{(f)T}\big)\mathbf{u}$. The other transformation matrices can be obtained using a similar iterative procedure by solving the corresponding generalized eigenvalue problems. Algorithm 1 summarizes the complete TNPE algorithm.

Tensor Locality Preserving Projection

Similar to Laplacian Eigenmap and LPP, TLPP provides a way to linearly approximate the eigenfunctions of the Laplace Beltrami operator in a tensor space. Therefore it can model the geometric and topological properties of an unknown manifold embedded in a tensor space with some data points sampled randomly from the manifold.

Based on n data points $\mathcal{A}_1, \ldots, \mathcal{A}_n$ from a manifold $\mathcal{M} \subset \mathbb{R}^{I_1 \times \cdots \times I_k}$, we first construct a neighborhood graph G to represent the local geometric structure of $\mathcal{M}$. The corresponding affinity matrix $\mathbf{S} = [s_{ij}]_{n \times n}$ is defined based on the heat kernel as:

$$s_{ij} = \begin{cases} \exp(-\|\mathcal{A}_i - \mathcal{A}_j\|_F^2/t), & \text{if } \mathcal{A}_j \in O(K, \mathcal{A}_i) \\ & \text{or } \mathcal{A}_i \in O(K, \mathcal{A}_j); \\ 0, & \text{otherwise.} \end{cases}$$

Let $\mathbf{U}_i \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$ be the corresponding transformation matrices. Based on the neighborhood graph G,

Algorithm 1 TNPE

```
**Input**: $\mathcal{A}_1, \ldots, \mathcal{A}_n$ from $\mathcal{M} \subset \mathbb{R}^{I_1 \times \cdots \times I_k}$ and $l_1 \times \cdots \times l_k$.
1. Construct $G$ and compute $\mathbf{S}$;
2. Compute the embedding as follows:
Initialize $\mathbf{U}_1^0 = \mathbf{I}_{I_1}, \ldots, \mathbf{U}_k^0 = \mathbf{I}_{I_k}$;
for $t = 1, \ldots, T_{\text{max}}$ do
  for $f = 1, \ldots, k$ do
    $\mathcal{Y}_i^f = \mathcal{A}_i \times_1 \mathbf{U}_1 \cdots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \cdots \times_k \mathbf{U}_k$;
    $\mathcal{Y}_i^f \Rightarrow_f \mathbf{Y}_i^{(f)}$;
    $\mathbf{H}_1 = \sum_i (\mathbf{Y}_i^{(f)} - \sum_l s_{ij} \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \sum_l s_{ij} \mathbf{Y}_j^{(f)})^T$;
    $\mathbf{H}_2 = \sum_i \mathbf{Y}_i^{(f)} \mathbf{Y}_i^{(f)T}$;
    $\mathbf{H}_1 \mathbf{U}_f^t = \mathbf{H}_2 \mathbf{U}_f^t \mathbf{\Lambda}_k$, $\mathbf{U}_f^t \in \mathbb{R}^{l_f \times I_f}$ ;
    if $\|\mathbf{U}_f^t - \mathbf{U}_f^{t-1}\|_F < \varepsilon$ for each $f$ then
      break;
    end if
  end for
end for
Output: $\mathbf{U}_i = \mathbf{U}_i^t \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$.
```

Algorithm 2 TLPP

```
Input: $\mathcal{A}_1, \ldots, \mathcal{A}_n$ from $\mathcal{M} \subset \mathbb{R}^{I_1 \times \cdots \times I_k}$ and $l_1 \times \cdots \times l_k$.
1. Construct $G$ and compute $\mathbf{S}$;
2. Compute the embedding as follows:
Initialize $\mathbf{U}_1^0 = \mathbf{I}_{I_1}, \ldots, \mathbf{U}_k^0 = \mathbf{I}_{I_k}$;
for $t = 1, \ldots, T_{\text{max}}$ do
  for $f = 1, \ldots, k$ do
    $\mathcal{Y}_i^f = \mathcal{A}_i \times_1 \mathbf{U}_1 \cdots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \cdots \times_k \mathbf{U}_k$;
    $\mathcal{Y}_i^f \Rightarrow_f \mathbf{Y}_i^{(f)}$;
    $\mathbf{H}_1 = \sum_{i,j} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_l^{(f)})^T s_{ij}$;
    $\mathbf{H}_2 = \sum_i \mathbf{Y}_i^{(f)} \mathbf{Y}_i^{(f)T} d_{ii}$;
    $\mathbf{H}_1 \mathbf{U}_f^t = \mathbf{H}_2 \mathbf{U}_f^t \mathbf{\Lambda}_k$, $\mathbf{U}_f^t \in \mathbb{R}^{l_f \times I_f}$ ;
    if $\|\mathbf{U}_f^t - \mathbf{U}_f^{t-1}\|_F < \varepsilon$ for each $f$ then
      break;
    end if
  end for
end for
Output: $\mathbf{U}_i = \mathbf{U}_i^t \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$.
```

the optimization problem for TLPP can be expressed as:

$$\begin{aligned} &\arg\min \; Q(\mathbf{U}_1, \ldots, \mathbf{U}_k) \qquad (7) \\ &= \sum_{i,j} \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k - \mathcal{A}_j \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 s_{ij}, \\ &\text{s.t.} \quad \sum_i \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 d_{ii} = 1. \end{aligned}$$

In general, the larger the value of $d_{ii} = \sum_j s_{ij}$ is, the more important is the data point $\mathcal{B}_i$ in the embedded tensor space for representing the data point $\mathcal{A}_i$. It is easy to see that the objective function will give a high penalty if neighboring points $\mathcal{A}_i$ and $\mathcal{A}_j$ are mapped far apart. Thus if two points $\mathcal{A}_i$ and $\mathcal{A}_j$ are close to each other, then the corresponding points $\mathcal{B}_i$ and $\mathcal{B}_j$ in the embedded tensor space are also expected to be close to each other. Similar to TNPE, we solve this optimization problem by applying an iterative scheme. Assuming that $\mathbf{U}_1, \ldots, \mathbf{U}_{f-1}, \mathbf{U}_{f+1}, \ldots, \mathbf{U}_k$ are known, we denote $\mathcal{A}_i \times_1 \mathbf{U}_1 \cdots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \cdots \times_k \mathbf{U}_k$ by $\mathcal{Y}_i^f$. In addition, since $\mathcal{Y}_i^f \Rightarrow_f \mathbf{Y}_i^{(f)}$ and based on the properties of tensor and trace, we reformulate the optimization function in (7) as follows:

$$\begin{aligned} &\arg\min \; P_f(\mathbf{U}_f) = \qquad (8) \\ &\text{tr}\big\{\mathbf{U}_f\big(\sum_{i,j} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T s_{ij}\big)\mathbf{U}_f^T\big\}, \\ &\text{s.t.} \quad \text{tr}\big\{\mathbf{U}_f\big(\sum_i \mathbf{Y}_i^{(f)} \mathbf{Y}_i^{(f)T} d_{ii}\big)\mathbf{U}_f^T\big\} = 1. \end{aligned}$$

The unknown transformation matrix $\mathbf{U}_f$ can be obtained by solving for the eigenvectors corresponding to the l_f smallest eigenvalues in the generalized eigenvalue equation $\big(\sum_{i,j} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T s_{ij}\big)\mathbf{u} = \lambda\big(\sum_i \mathbf{Y}_i^{(f)} \mathbf{Y}_i^{(f)T} d_{ii}\big)\mathbf{u}$. The other transformation matrices can be obtained in a similar manner. Algorithm 2 summarizes the complete TLPP algorithm.

Tensor Local Discriminant Embedding

Unlike TNPE and TLPP which can be used for unsupervised, semi-supervised or supervised learning tasks, TLDE is only defined for supervised learning tasks. As in (Chen, Chang, & Liu 2005), given n data points $\mathcal{A}_1, \ldots, \mathcal{A}_n \in \mathbb{R}^{I_1 \times \cdots \times I_k}$ and the corresponding labels $y_1, \ldots, y_n \in \{1, \ldots, c\}$, we assume that any subset of data points belonging to the same class lies in a submanifold $\mathcal{M} \subset \mathbb{R}^{I_1 \times \cdots \times I_k}$. TLDE aims to find k transformation matrices $\mathbf{U}_1, \ldots, \mathbf{U}_k$ by integrating the class label information and the neighborhood information to dissociate the submanifolds for different classes from each other. Like LDE, we first construct within-class and between-class neighborhood graphs G and G' to represent the local within-class and between-class neighborhood relationships, respectively. The corresponding affinity matrices $\mathbf{S} = [s_{ij}]_{n \times n}$ and $\mathbf{S}' = [s'_{ij}]_{n \times n}$ are then defined based on the heat kernel:

$$s_{ij} = \begin{cases} \exp(-\|\mathcal{A}_i - \mathcal{A}_j\|_F^2/t), & \text{if } (\mathcal{A}_j \in O(K, \mathcal{A}_i) \\ & \text{or } \mathcal{A}_i \in O(K, \mathcal{A}_j)) \\ & \text{and } y_i = y_j; \\ 0, & \text{otherwise.} \end{cases}$$

and

$$s'_{ij} = \begin{cases} \exp(-\|\mathcal{A}_i - \mathcal{A}_j\|_F^2/t), & \text{if } (\mathcal{A}_j \in O(K, \mathcal{A}_i) \\ & \text{or } \mathcal{A}_i \in O(K, \mathcal{A}_j)) \\ & \text{and } y_i \neq y_j; \\ 0, & \text{otherwise.} \end{cases}$$

The optimization problem for TLDE can be expressed in the following form:

$$\begin{aligned} &\arg\min \; Q(\mathbf{U}_1, \ldots, \mathbf{U}_k) \qquad (9) \\ &= \sum_{i,j} \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k - \mathcal{A}_j \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 s_{ij}, \\ &\text{s.t.} \quad \sum_{i,j} \|\mathcal{A}_i \times_1 \cdots \times_k \mathbf{U}_k - \mathcal{A}_j \times_1 \cdots \times_k \mathbf{U}_k\|_F^2 s'_{ij} = 1. \end{aligned}$$

From this optimization problem, it is easy to note that neighboring points in the original space with the same class label tend to remain close to each other in the embedded tensor space, while preventing points of other classes from entering the neighborhood.

We again apply an iterative scheme to solve this optimization problem. We assume that $\mathbf{U}_1, \ldots, \mathbf{U}_{f-1}, \mathbf{U}_{f+1}, \ldots,$

Algorithm 3 TLDE

```
**Input**: $\mathcal{A}_1, \ldots, \mathcal{A}_n$ from $\mathcal{M} \subset \mathbb{R}^{I_1 \times \cdots \times I_k}$, $\{y_i \mid y_i \in \{1, \ldots, h\}\}_{i=1}^n$ and $l_1 \times \cdots \times l_k$.
1. Construct neighborhood graphs $G$ and $G'$ and compute affinity matrices $\mathbf{S}$ and $\mathbf{S}'$ for $G$ and $G'$, respectively;
2. Compute the embedding as follows:
Initialize $\mathbf{U}_1^0 = \mathbf{I}_{I_1}, \ldots, \mathbf{U}_k^0 = \mathbf{I}_{I_k}$;
**for** $t = 1, \ldots, T_{\max}$ **do**
  **for** $f = 1, \ldots, k$ **do**
    $\mathcal{Y}_i^f = \mathcal{A}_i \times_1 \mathbf{U}_1 \cdots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \cdots \times_k \mathbf{U}_k$;
    $\mathcal{Y}_i^f \Rightarrow_f \mathbf{Y}_i^{(f)}$;
    $\mathbf{H}_1 = \sum_{i,j} s_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T$;
    $\mathbf{H}_2 = \sum_{i,j} s'_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T$;
    $\mathbf{H}_1 \mathbf{U}_f^t = \mathbf{H}_2 \mathbf{U}_f^t \mathbf{\Lambda}_k$, $\mathbf{U}_f^t \in \mathbb{R}^{l_f \times I_f}$ ;
    **if** $\|\mathbf{U}_f^t - \mathbf{U}_f^{t-1}\|_F < \varepsilon$ for each $f$ **then**
      **break**;
    **end if**
  **end for**
**end for**
**Output**: $\mathbf{U}_i = \mathbf{U}_i^t \in \mathbb{R}^{l_i \times I_i}$ $(i = 1, \ldots, k)$.
```

$\mathbf{U}_k$ are known and denote $\mathcal{A}_i \times_1 \mathbf{U}_1 \ldots \times_{f-1} \mathbf{U}_{f-1} \times_{f+1} \mathbf{U}_{f+1} \ldots \times_k \mathbf{U}_k$ by $\mathcal{Y}_i^f$. We can rewrite the Frobenius norm in (9) in terms of the trace and express the optimization problem as:

$$
\begin{aligned}
& \arg\min \; P_f(\mathbf{U}_f) \qquad\qquad (10) \\
&= \mathrm{tr}\{\mathbf{U}_f^T (\sum_{i,j} s_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T) \mathbf{U}_f\}, \\
\text{s.t.} \quad & \mathrm{tr}\{\mathbf{U}_f^T \sum_{i,j} s'_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T \mathbf{U}_f\} = 1.
\end{aligned}
$$

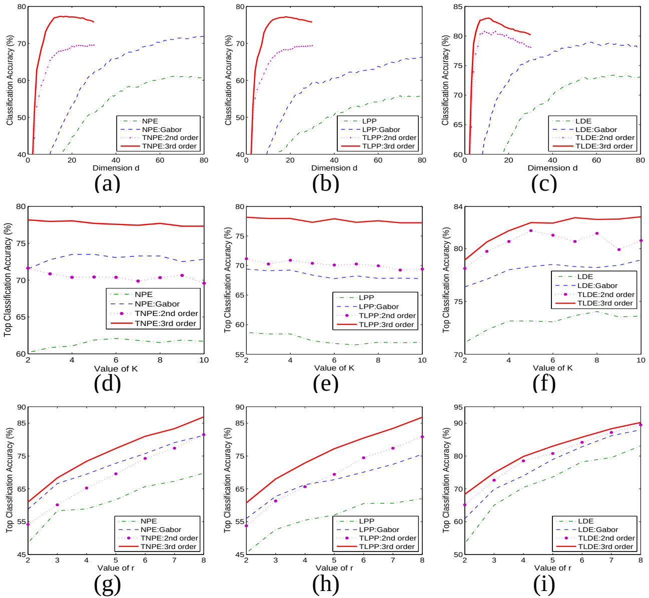
It can be seen that the columns of the unknown transformation matrix $\mathbf{U}_f$ are the eigenvectors corresponding to the l_f smallest eigenvalues in the generalized eigenvalue problem $\big(\sum_{i,j} s_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T\big)\mathbf{u} = \lambda \big(\sum_{i,j} s'_{ij} (\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})(\mathbf{Y}_i^{(f)} - \mathbf{Y}_j^{(f)})^T\big)\mathbf{u}$. So all the transformation matrices $\mathbf{U}_1, \ldots, \mathbf{U}_k$ can be found in an iterative manner. Algorithm 3 summarizes the complete TLDE algorithm.

Experimental Results

While the proposed tensor embedding algorithms are very general in that they can be applied to many problems, we conduct some face recognition experiments in this paper as an illustrative example to demonstrate the advantages of these embedding methods. Face recognition is a suitable domain for illustration because research in the past few years shows that face images generally correspond to low-dimensional manifolds embedded in the ambient space.

Our experiments are based on the FERET face database which has become the *de facto* benchmark for evaluating face recognition algorithms. We randomly select 47 subjects from the FERET database with 10 different gray-scale images for each subject. The images have been subsampled to a resolution of 56×46. They can be encoded in either a 2576-dimensional vectorized representation for traditional methods or a second-order tensor representation of size 56×46 for tensor embedding methods. Moreover, we also apply 40 Gabor filters corresponding to five scales and eight orientations. Similarly, the images can be encoded in either a 103040-dimensional vectorized Gabor representation for traditional methods or a third-order tensor representation of size $56 \times 46 \times 40$ for tensor embedding methods. Our data set is randomly partitioned into two disjoint subsets for training and testing. To compare the recognition performance with different training set sizes, r images per subject are randomly selected for training and the rest for testing. For each value of r, we repeat the experiments 10 times on different randomly selected training sets and the average classification results based on the nearest neighbor classifier are reported.

Since traditional methods such as PCA and LDA can be reformulated as special cases of manifold learning methods, the experiments reported here only compare TNPE, TLPP and TLDE with NPE, LPP and LDE. For simplicity, all neighborhood graphs are constructed based on the K-nearest-neighbor criterion. As discussed above, while the neighborhood graphs for NPE, LPP, TNPE and TLPP are constructed without incorporating any label information, the within-class and between-class neighborhood graphs for LDE and TLDE are constructed based on the same K-nearest-neighbor criterion by incorporating label information. In addition, in order to reduce the computational demand in NPE, LPP and LDE, PCA is employed as an intermediate step.



(a) (b) (c)
(d) (e) (f)
(g) (h) (i)

Figure 1: Comparative study based on the FERET database. (a)∼(c) Classification accuracy (%) vs. dimensionality d, with $K = 10, r = 5$; (d)∼(f) Top classification accuracy (%) vs. neighborhood size K, with $r = 5$; (g)∼(i) Top classification accuracy (%) vs. training set size r, with $K = 10$.

Figure 1 shows the comparative results for NPE, LPP, LDE, TNPE, TLPP and TLDE. For simplicity, for a given

d, we perform our tensor embedding methods on the tensor space $\mathbb{R}^{l_1 \times \cdots \times l_k}$ with $l_i = d\,(i = 1, \ldots, k)$. From Figure 1(a)∼(c), we can see that our tensor embedding methods outperform NPE, LPP and LDE over a wide range of dimensionality choices. The advantages persist even by varying the neighborhood size and the training set size, as shown in Figure 1(d)∼(f) and Figure 1(g)∼(i), respectively. While the second-order tensor representation and third-order tensor representation already outperform the original vectorized representation and vectorized Gabor representation, respectively, the third-order tensor representation gives even better classification results.

Let us consider the space requirements of different algorithms in the original forms. Without considering the affinity matrices which are typically very sparse, it is easy to notice the following. For 56×46 tensors, the maximum matrix size involved in NPE, LPP and LDE is 2676×2676 while that in our methods is only 56×56. On the other hand, for $56 \times 46 \times 40$ tensors, the maximum matrix size involved in NPE, LPP and LDE is 103040×103040 while that in our methods is only 56×1840. More generally, we compare the space complexity and time complexity of our methods with the previous methods in the original forms. For simplicity, we assume that the number of iterations in the algorithms of our tensor embedding methods is a small constant value which does not depend on the experimental settings. Our experiments show that this is a reasonable assumption to adopt. We let $H = \Pi_{i=1}^{k} I_i$ where k is the order of the tensors and $I_{max} = \max\{I_1, \ldots, I_k\}$. Table 1 compares the space complexity and time complexity of our methods with the previous methods. Since the number of data points n is usually far less than H in many real-world applications, it is clear that our tensor embedding methods are more appealing in terms of both complexity measures.

Table 1: Comparison of our tensor embedding methods with NPE, LPP and LDE in terms of the space complexity and time complexity.

	NPE, LPP & LDE	TNPE, TLPP & TLDE
SPACE	$O(H^2)$	$O(nH)$
TIME	$O(H^3)$	$O(k(n^2 H I_{max} + I_{max}^3))$

Conclusion

Based on some recently proposed embedding methods, we have developed generalizations which can take data directly in the form of tensors of arbitrary order as input. Not only do our methods inherit the attractive characteristics of the previous methods in terms of exploiting the intrinsic local geometric and topological properties of the manifold, they are also appealing in terms of significant reduction in both space complexity and time complexity. Face recognition experiments based on the FERET database demonstrate that our tensor embedding methods give very impressive results.

Acknowledgments

This research has been supported by Competitive Earmarked Research Grant HKUST621305 from the Research Grants Council of the Hong Kong Special Administrative Region, China.

References

Belkin, M., and Niyogi, P. 2003. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation* 15(6):1373–1396.

Bengio, Y.; Delalleau, O.; Roux, N.; Paiement, J.; Vincent, P.; and Ouimet., M. 2004. Learning eigenfunctions links spectral embedding and kernel PCA. *Neural Computation* 16(10):2197–2219.

Chen, H.; Chang, H.; and Liu, T. 2005. Local discriminant embedding and its variants. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 846–853.

Duda, R.; Hart, P.; and Stork, D. 2000. *Pattern Classification*. Wiley.

He, X., and Niyogi, P. 2004. Locality preserving projections. In *Advances in Neural Information Processing Systems 16*.

He, X.; Cai, D.; Yan, S.; and Zhang, H. 2005. Neighborhood preserving embedding. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, 1208–1213.

He, X.; Cai, D.; and Niyogi, P. 2006. Tensor subspace analysis. In *Advances in Neural Information Processing Systems 18*.

Roweis, S., and Saul, L. 2000. Nonlinear dimensionality reduction by locally linear embedding. *Science* 290(5500):2323–2326.

Schölkopf, B., and Smola, A. 2002. *Learning with Kernels*. Cambridge, MA: MIT Press.

Tenenbaum, J.; Silva, V.; and Langford, J. 2000. A global geometric framework for nonlinear dimensionality reduction. *Science* 290(5500):2319–2323.

Vasilescu, M., and Terzopoulos, D. 2003. Multilinear subspace analysis for image ensembles. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 93–99.

Wang, H.; Wu, Q.; Shi, L.; Yu, Y.; and Ahuja, N. 2005. Out-of-core tensor approximation of multi-dimensional matrices of visual data. *ACM Transactions on Graphics* 24(3):527–535.

Xu, D.; Yan, S.; Zhang, L.; Zhang, H.; Liu, Z.; and Shum, H. 2005. Concurrent subspaces analysis. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 203–208.

Yang, J.; Zhang, D.; Frangi, A.; and Yang, J. 2004. Two-dimensional pca: A new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 26(1):131–137.

Ye, J.; Janardan, R.; and Li, Q. 2005. Two-dimensional linear discriminant analysis. In *Advances in Neural Information Processing Systems 17*.

Ye, J. 2004. Generalized low rank approximations of matrices. In *Proceedings of the Twenty-First International Conference on Machine Learning*.