Element Rearrangement for Tensor-Based Subspace Learning

Shuicheng Yan[1], Dong Xu[2], Stephen Lin[3], Thomas S. Huang[1], and Shih-Fu Chang[2]
[1]Beckman Institute, University of Illinois at Urbana-Champaign, USA
[2] Department of Electrical Engineering, Columbia University, USA
[3] Microsoft Research Asia, Beijing, China
Contact:scyan@ifp.uiuc.edu

Abstract

The success of tensor-based subspace learning depends heavily on reducing correlations along the column vectors of the mode-k flattened matrix. In this work, we study the problem of rearranging elements within a tensor in order to maximize these correlations, so that information redundancy in tensor data can be more extensively removed by existing tensor-based dimensionality reduction algorithms. An efficient iterative algorithm is proposed to tackle this essentially integer optimization problem. In each step, the tensor structure is refined with a spatially-constrained Earth Mover's Distance *procedure that incrementally rearranges tensors to become more similar to their low rank approximations, which have high correlation among features along certain tensor dimensions. Monotonic convergence of the algorithm is proven using an auxiliary function analogous to that used for proving convergence of the Expectation-Maximization algorithm. In addition, we present an extension of the algorithm for conducting supervised subspace learning with tensor data. Experiments in both unsupervised and supervised subspace learning demonstrate the effectiveness of our proposed algorithms in improving data compression performance and classification accuracy.*

1. Introduction

Image data naturally contains a significant amount of information redundancy, as evidenced by spatial coherence and structural commonalities found within images and image sets. For processing and analysis of images, it is often advantageous to pare away these redundancies so that the intrinsic features of the data are revealed. This is the goal of dimensionality reduction techniques, which aim to decrease the size of a feature space by removing correlations among the features. Dimensionality reduction has proven to be useful for unsupervised learning tasks such as data compression, and facilitates supervised learning by identifying fewer but more effective features.

Frequently-used dimensionality reduction techniques such as Principal Components Analysis (PCA) [12], Linear Discriminant Analysis (LDA) [1] and Tensorfaces [13] commonly unfold each image into a single column vector. While correlations among different pixels can be reduced in these vector-based techniques, lengthy column vectors typically result in the curse of dimensionality, and classification performance may be degraded because of the small-sample-size problem.

Recently, a large number of works (e.g., [2] [14] [15] [17] [18]) have sought to process image data in their original form, i.e., images as matrices instead of as vectors. By aiming to reduce correlations only within image rows and columns, rather than among all pixels in an image, the curse of dimensionality is avoided and the small-sample-size problem becomes greatly diminished. For image data represented in matrix form [17] [18], appreciably higher recognition performance has been experimentally reported, especially in cases with small training sets. Similar improvements have also been found for the more general case of tensor data, where correlations are removed along column vectors of mode-k flattened matrices [14] [15]. We will henceforth consider matrix data as a special case of tensors.

Correlations within tensor image data, however, are not limited to the elements along certain tensor dimensions. Work on natural image statistics indicate that such correlations are frequently present among different regions both spatially within an image and temporally through an image sequence [11]. For enhancement of subspace learning, a natural question that arises is whether these non-column correlations can be reduced while preserving the performance benefits of processing image data in its original tensor form.

In this paper, we propose to address this problem by rearranging the elements within tensors to increase the correlations among features along certain tensor dimensions, which we will refer to as intra-tensor correlations. An illustration of element rearrangement is shown in Fig. 1. By aligning elements in a way that increases correlations along the three dimensions of a 3^{rd} order tensor, greater reductions in dimensionality can be achieved with existing

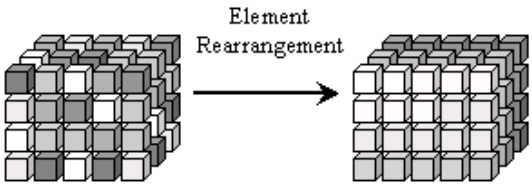


Figure 1. An illustration of element rearrangement for a 3^{rd} order tensor.

tensor-based methods.

We show in Section 2 that rearranging tensor elements to maximize intra-tensor correlations is essentially an *integer optimization* problem with a nonlinear objective function. Since this problem is NP-hard, we present in this work an approximate iterative solution. As described in Section 3, we first employ the Concurrent Subspace Analysis [14] technique to compute projection matrices from the training data tensors, then the elements in these tensors are rearranged to become more similar to the reconstructions of the training tensors by the projection matrices. This procedure is then iterated using the rearranged tensors. Reconstructed tensors are used to guide the rearrangement process because they are similar to the training data tensors and have high correlation among features along certain tensor dimensions. For greater computational feasibility, the displacement of elements within a tensor is constrained in each iteration to a limited distance. Element rearrangement is formulated as a spatially-constrained *Earth Mover's Distance* [9] problem, where the flows from the elements of the original tensor to those of the target tensor naturally constitute a pure network flow model [5]. An integer solution can then be reached by general linear programming. In Section 4, we prove that this iterative algorithm monotonically converges via an auxiliary function analogous to that used for proving convergence of the Expectation-Maximization algorithm [4].

We demonstrate in Section 5 the utility of tensor element rearrangement for data compression and supervised subspace learning. Since rearrangement is employed in both applications as a preprocessing step to increase intra-tensor correlations, it can be used in conjunction with any tensor-based dimensionality reduction technique. For supervised subspace learning, we extend the proposed algorithm using LDA as an example. In this extension, the weighted and centered class means are used as training samples in LDA, and the element rearrangement algorithm is then performed on these new tensors. With this approach, the most discriminant information becomes encoded in the first few dimensions of the derived subspace.

2. Problem Formulation

In this section, we first briefly review Concurrent Subspace Analysis (CSA) and then introduce our formulation of the tensor element rearrangement problem for maximizing intra-tensor correlation. We express the training sample set in tensor form as $\{\mathbf{X_i} \in \mathbb{R}^{m_1 \times m_2 \times \ldots \times m_n}, i = 1, 2, \ldots, N\}$, where n is the order of the tensor, m_k is the tensor size along dimension k and N is the number of samples. Before describing the criteria for guiding the element rearrangement of a tensor, we review two tensor operators. The *norm* of a tensor $\mathbf{X}$ is defined as

$$\|\mathbf{X}\| = \sqrt{\sum_{i_1=1}^{m_1} \sum_{i_2=1}^{m_2} \cdots \sum_{i_n=1}^{m_n} \mathbf{X}_{i_1,i_2,\cdots,i_n}^2}. \quad (1)$$

The *mode-k product* of a tensor $\mathbf{A}$ and a matrix $U \in \mathbb{R}^{m'_k \times m_k}$ is defined as $\mathbf{B} = \mathbf{A} \times_k U$, where $\mathbf{B}_{i_1,\ldots,i_{k-1},i,i_{k+1},\ldots,i_n} = \sum_{j=1}^{m_k} \mathbf{A}_{i_1,\ldots,i_{k-1},j,i_{k+1},\ldots,i_n} \times U_{ij}, i = 1, \ldots, m'_k$.

Concurrent Subspace Analysis targets an optimal reconstruction for objects represented as high order tensors. We express its projection matrices as $U^k \in \mathbb{R}^{m_k \times m'_k}, k = 1, \ldots, n$, and denote the reconstructed tensor as $\mathbf{X}_i^{REC} \in \mathbb{R}^{m_1 \times m_2 \times \ldots \times m_n}$, which can be computed as $\mathbf{X}_i^{REC} = \mathbf{X}_i \times_k U^k U^{k^T}|_{k=1}^n$, where $\times_k U^k U^{k^T}|_{k=1}^n = \times_1 U^1 U^{1^T} \times_2 U^2 U^{2^T} \cdots \times_n U^n U^{n^T}$. The projection matrices $U^k|_{k=1}^n = U^1, U^2, \cdots, U^n$ can be solved using the following objective function:

$$\sum_{i=1}^{N} \|\mathbf{X}_i - \mathbf{X}_i \times_k U^k U^{k^T}|_{k=1}^n\|^2. \quad (2)$$

There is no closed form solution for $U^k|_{k=1}^n$, so in CSA a variant of High-Order SVD (HOSVD) [6] is proposed to iteratively optimize U^k while fixing the other projection matrices $U^1, \ldots, U^{k-1}, U^{k+1}, \ldots, U^n$. Further details can be found in [14].

In this work, we utilize CSA in formulating the problem of element rearrangement. For ease of understanding, we denote the position of a tensor element as $(i_1^p, i_2^p, \cdots, i_n^p)$ and its global index as p, where $p = 1 + \sum_{l=1}^{n} (i_l^p - 1) \prod_{o=l+1}^{n} m_o$. Similarly, the position $(i_1^q, i_2^q, \cdots, i_n^q)$ corresponds to the global index q. We define the rearrangement operator as $R \in \{0,1\}^{(m_1 \times m_2 \times \cdots \times m_n) \times (m_1 \times m_2 \times \cdots \times m_n)}$, and $(\mathbf{X}_i(R))_q = (\mathbf{X}_i)_p$ if $R_{pq} = 1$. For operator R, we have the properties $\sum_p R_{pq} = 1$ and $\sum_q R_{pq} = 1$, meaning that the elements in tensors $\mathbf{X}_i$ and $\mathbf{X}_i(R)$ have a one-to-one correspondence. In element rearrangement, we wish to minimize the objective function $F(U^k|_{k=1}^n, R)$ expressed as

$$\sum_{i=1}^{N} \|\mathbf{X}_i(R) - \mathbf{X}_i(R) \times_k U^k U^{k^T}|_{k=1}^n\|^2. \quad (3)$$

Algorithm 1 : Sub-procedure for tensor matching.

$\arg\min_{R_{pq}} \sum_p \sum_{q\in N_p^\gamma} c_{pq}R_{pq}$, s.t.

1: $0 \le R_{pq} \le 1$;
2: $\sum_{p;q\in N_p^\gamma} R_{pq} = 1,\ \forall q$;
3: $\sum_{q;p\in N_q^\gamma} R_{pq} = 1,\ \forall p$;

Essentially, this function seeks rearranged tensors that can be best approximated by their reconstructed low rank tensors, which have high correlation among features along certain tensor dimensions.

This objective function presents a complicated integer optimization problem with nonlinear objective functions. Since this problem is NP-hard, we present in the following section an approximate solution to iteratively compute the projection matrices $U^k|_{k=1}^n$ and the rearrangement operator R. Note that when R is fixed, $U^k|_{k=1}^n$ can be computed with CSA. In the following sections, we therefore focus on how to compute the rearrangement operator R.

3. Approximate Iterative Solution

Given the projection matrices $U^k|_{k=1}^n$, minimization of $F(U^k|_{k=1}^n, R)$ with respect to element rearrangement operator R is an integer optimization problem. For the t-th iteration step, let $\{\mathbf{X}_i^t, i = 1, 2, \cdots, N\}$ be the tensor data rearranged from $\mathbf{X}_i^{t-1}$. CSA can be applied on $\mathbf{X}_i^t$ to compute the projection matrices $U^{k,t}, k = 1, 2, \cdots, n$, then the reconstructed tensor at the t-th iteration can be computed as

$$\mathbf{X}_i^{t\,REC} = \mathbf{X}_i^t \times_1 U^{1,t}(U^{1,t})^T \cdots \times_n U^{n,t}(U^{n,t})^T. \quad (4)$$

To facilitate optimization, we impose a constraint that at each iteration each element p of a tensor can be moved only within its neighboring area N_p^γ, bounded by a distance of $\gamma = 2\sqrt{2}$ pixels in this work. Therefore, $R_{pq} = 0$ if $q \notin N_p^\gamma$. In each step, we fix the term $\mathbf{X}_i(R) \times_k U^k U^{k^T}|_{k=1}^n$ in Eq. (3) to be $\mathbf{X}_i^{t\,REC}$, the rationale of which will be explained in Section 4. Then, the objective function in Eq. (3) can be rewritten as

$$F(R) = \sum_{i=1}^{N} \|\mathbf{X}_i^t(R) - \mathbf{X}_i^{t\,REC}\|^2. \quad (5)$$

Let us define

$$c_{pq} = \sum_{i=1}^{N} \|(\mathbf{X}_i^t)_{i_1^p,\cdots,i_n^p} - (\mathbf{X}_i^{t\,REC})_{i_1^q,\cdots,i_n^q}\|^2. \quad (6)$$

Since $R_{pq} \in \{0, 1\}$, the objective function can then be expressed as

$$F(R) = \sum_p \sum_{q\in N_p^\gamma} c_{pq}R_{pq}. \quad (7)$$

Algorithm 2 : Full algorithm for tensor rearrangement.

Given the sample tensors $\{\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_N\}$, and the dimensions $(m'_1, m'_2, \cdots, m'_n)$.

1: Initialize $\mathbf{X}_i^1 = \mathbf{X}_i$;
2: For $t = 1, 2, \ldots$

a: Compute $U^{k,t}, k = 1, 2, \cdots, n$, using CSA [14] based on the tensor data $\mathbf{X}_i^t, i = 1, 2, \cdots, N$. If $t > 1$, $U^{k,t}$ is initialized as $U^{k,t-1}$ in CSA.
b: Compute the rearrangement operator R by solving the linear programming problem in Algorithm 1.
c: Update the tensor $\mathbf{X}_i^t$ to $\mathbf{X}_i^{t+1}$ according to R.
d: If $R_{pq} = 0$ for all $p \neq q$, then exit.

3: Output the rearranged data $\{\mathbf{X}_i^t, i = 1, \cdots, N\}$.

The correspondence between the elements of the original tensor and the rearranged tensor must be one-to-one, which imposes the following constraints:

$$\sum_{p;q\in N_p^\gamma} R_{pq} = 1,\ \forall q, \quad \sum_{q;p\in N_q^\gamma} R_{pq} = 1,\ \forall p. \quad (8)$$

The problem is then simplified into an integer optimization problem with a linear objective function and linear constraints. If we relax the integer constraints for the elements R_{pq}, then the problem becomes a special spatially-constrained *Earth Mover's Distance* [9] problem as outlined in Algorithm 1.

The linear programming problem described in Algorithm 1 will have an integer solution, as guaranteed by the following theorem.

Theorem-1 [5][7]: An integer programming problem

$$\begin{gathered}\min_R \textstyle\sum_{p,q} c_{pq}R_{pq},\ s.t. \\ \textstyle\sum_q R_{pq} = 1,\ \forall p,\ \text{ and }\ \sum_p R_{pq} = 1,\ \forall q,\end{gathered} \quad (9)$$

can be solved as a linear programming problem with the constraints $0 \le R_{pq} \le 1,\ \forall p, q$.

The equivalence of integer programming and linear programming arises from the unimodular coefficient matrix of the integer programming problem in Eq. (9), and is based on the assumption that linear programming is solved with the simplex method, which is computationally efficient. Further details on this theorem can be found in [7]. It is straightforward to prove that our proposed spatially-constrained procedure in Algorithm 1 is equal to Eq. (9) with $c_{p,q} = +\infty$ for $q \notin N_p^\gamma$. Hence the linear programming solution for the procedure in Algorithm 1 is also an integer solution, meaning that for any given p (or q), there exists only one R_{pq} that is one for all q (or p). Consequently, the element rearrangement operator R_{pq} constructs a one-to-one correspondence between the original and target tensors.

Based on the derived solution R, we rearrange the tensor data and update the training samples to obtain $\mathbf{X}_i^{t+1}$. Then, CSA and the proposed procedure are repeated until convergence, which occurs when R contains no more element movement, i.e., $R_{pq} = 0$ if $p \neq q$. The overall solution method is outlined in Algorithm 2.

Complexity Analysis. Each step iterates between CSA and linear programming for tensor element rearrangement. For a 2^{nd} order tensor of size 64×64 and $\gamma = 2\sqrt{2}$, there exist about $4096 \times 25 = 102400$ parameters[1] and $4096 \times 2 = 8192$ constraints. The constraint matrix is very sparse since each R_{pq} appears in it only three times. The linear program can hence be rapidly processed. In our experiments on a 2.8G CPU with 1.0G memory, the training time for each iteration is about 30 seconds with unoptimized Matlab code, and algorithm convergence is reached in about twenty iterations.

Discussion. In Algorithm 1, the spatial constraints restrict tensor element movement in each iteration to within a limited neighborhood. This bound was placed mainly to ensure the computational feasibility of the linear programming problem. Without these constraints, the number of parameters would balloon to 4096^2 for a 2^{nd} order tensor of size 64×64, which is computationally prohibitive both in complexity and memory requirements. Although movement is constrained within a neighborhood in each iteration, movement between distant pixels is possible after several iterations.

4. Justifications and Extensions

4.1. Theoretical Justifications

To prove the convergence of Algorithm 2, we will make use of an auxiliary function similar to that used in the Expectation-Maximization algorithm [4]. Let

$$F(\mathbf{X}_i|_{i=1}^N) = \sum_{i=1}^N \|\mathbf{X}_i - \mathbf{X}_i \times_k U^k U^{k^T}|_{k=1}^n\|^2,$$

$$G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N) = \sum_{i=1}^N \|\mathbf{X}_i - \mathbf{X'}_i \times_k U'^k U'^{k^T}|_{k=1}^n\|^2,$$

where $U^k|_{k=1}^n$ are computed from $\mathbf{X}_i|_{i=1}^N$ by minimizing $\sum_{i=1}^N \|\mathbf{X}_i - \mathbf{X}_i \times_k U^k U^{k^T}|_{k=1}^n\|^2$ with the CSA algorithm; and similarly, $U'^k|_{k=1}^n$ are computed from $\mathbf{X'}_i|_{i=1}^N$ by minimizing $\sum_{i=1}^N \|\mathbf{X'}_i - \mathbf{X'}_i \times_k U'^k U'^{k^T}|_{k=1}^n\|^2$ with CSA.

Definition: The function $G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N)$ is an auxiliary function for $F(\mathbf{X}_i|_{i=1}^N)$ if the following two conditions are satisfied: 1) $F(\mathbf{X}_i|_{i=1}^N) \leq G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N)$, and 2) $G(\mathbf{X}_i|_{i=1}^N, \mathbf{X}_i|_{i=1}^N) = F(\mathbf{X}_i|_{i=1}^N)$.

Theorem-2: The function $G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N)$ defined above is an auxiliary function for $F(\mathbf{X}_i|_{i=1}^N)$.

[1]The actual number of parameters is 98596 because some neighboring pixels do not exist for border pixels.

Proof. It is obvious that the second condition for an auxiliary function is satisfied. Here, we prove that the first condition also holds. For fixed $\mathbf{X}_i|_{i=1}^N$, we denote

$$Q(\tilde{\mathbf{Z}}_i|_{i=1}^N, \tilde{U^k}|_{k=1}^n) = \sum_{i=1}^N \|\mathbf{X}_i - \tilde{\mathbf{Z}}_i \times_k \tilde{U^k}|_{k=1}^n\|^2,$$

where $\tilde{\mathbf{Z}}_i \in \mathbf{R}^{m'_1 \times \cdots m'_n}$ and $\tilde{U^k} \in \mathbf{R}^{m_k \times m'_k}$. For given $\tilde{U^k}|_{k=1}^n$, the optimum is given by $\tilde{\mathbf{Z}}_i = \mathbf{X}_i \times_k \tilde{U^k}^T|_{k=1}^n$, so we have

$$F(\mathbf{X}_i|_{i=1}^N) \leq Q((\mathbf{X}_i \times_k U'^k|_{k=1}^n)|_{i=1}^N, U'^k|_{k=1}^n) \leq$$
$$Q((\mathbf{X'}_i \times_k U'^k|_{k=1}^n)|_{i=1}^N, U'^k|_{k=1}^n) = G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N).$$

$G(\mathbf{X}_i|_{i=1}^N, \mathbf{X'}_i|_{i=1}^N)$ is therefore an auxiliary function for $F(\mathbf{X}_i|_{i=1}^N)$. □

Notice that the first inequality is satisfied with the assumption that the CSA computation of projection matrices from tensors $\mathbf{X}_i|_{i=1}^N$ is initialized as $U'^k|_{k=1}^n$, and Algorithm 2 follows this rule if the auxiliary function takes the parameters as the rearranged tensors from two successive steps.

Theorem-3: From Algorithm 1 and 2, the objective function $F(\mathbf{X}_i^t|_{i=1}^N)$ will monotonically decrease until convergence.

Proof. From Theorem-2, we have $G(\mathbf{X}_i^{t+1}|_{i=1}^N, \mathbf{X}_i^{t+1}|_{i=1}^N) \leq G(\mathbf{X}_i^{t+1}|_{i=1}^N, \mathbf{X}_i^t|_{i=1}^N)$. As seen from Algorithm 1, $G(\mathbf{X}_i^{t+1}|_{i=1}^N, \mathbf{X}_i^t|_{i=1}^N) \leq G(\mathbf{X}_i^t|_{i=1}^N, \mathbf{X}_i^t|_{i=1}^N)$. Consequently, we can conclude that $0 \leq F(\mathbf{X}_i^{t+1}|_{i=1}^N) \leq F(\mathbf{X}_i^t|_{i=1}^N)$, and the objective function $F(\mathbf{X}_i^t|_{i=1}^N)$ will monotonically decrease until convergence. □

4.2. Extension for Supervised Subspace Learning

The purpose of the tensor rearrangement algorithm is to enhance intra-tensor correlations, which is useful for tensor data compression. Another important task of subspace learning is to derive low dimensional representations with strong discriminative power for different data classes. In this subsection, we examine how element rearrangement can enhance the discriminating power of tensor-based subspaces.

For the task of classification, the class labels of training samples $\mathbf{X}_i$ are denoted by $c_i \in \{1, 2, \cdots, N_c\}$, where N_c is the total number of classes, and n_c is the number of samples in the c-th class. Here, we discuss the popular supervised subspace learning algorithm, Linear Discriminant Analysis (LDA).

LDA can be extended to handle tensor data, as discussed in [3, 15, 18]. In the following, we take LDA as an example to show how to enhance discriminative ability by tensor element rearrangement.

LDA seeks a lower dimensional representation that minimizes intra-class scatter and at the same time maximizes

inter-class scatter. Let the vector representation of the training data be denoted by $\{x_1, x_2, \cdots, x_N\}$. The vector and tensor based LDA formulations can then be respectively expressed as

$$\max_{U} \frac{\sum_{c=1}^{N_c} n_c \|U^T(\overline{x}_c - \overline{x})\|^2}{\sum_{i=1}^{N} \|U^T(x_i - \overline{x}_{c_i})\|^2}, \tag{10}$$

$$\max_{U^k|_{k=1}^n} \frac{\sum_{c=1}^{N_c} n_c \|(\overline{\mathbf{X}}_c - \overline{\mathbf{X}}) \times_k U^{k^T}|_{k=1}^n\|^2}{\sum_{i=1}^{N} \|(\mathbf{X}_i - \overline{\mathbf{X}}_{c_i}) \times_k U^{k^T}|_{k=1}^n\|^2}, \tag{11}$$

where $\overline{\mathbf{X}}_c$ and $\overline{x}_c$ represent the mean of the c-th class, while $\overline{\mathbf{X}}$ and $\overline{x}$ are the means of all samples in tensor and vector forms respectively.

We develop a two-step procedure for improving algorithmic classification capability by element rearrangement. First, we compute the null space of the denominator of Eq. (10), which we represent as P_N, and then we reconstruct each data sample using P_N as

$$y_i = P_N P_N^T x_i. \tag{12}$$

We denote the corresponding tensor representation of the reconstructed data as $\{\mathbf{Y}_i, i = 1, 2, \cdots, N\}$. For the reconstructed data, the intra-class scatter is zero, hence the optimization of Eq. (11) is simplified to

$$\max_{U^k|_{k=1}^n} \sum_{c=1}^{N_c} n_c \|(\overline{\mathbf{Y}}_c - \overline{\mathbf{Y}}) \times_k U^{k^T}|_{k=1}^n\|^2,$$

which is equivalent to minimizing

$$\sum_{c=1}^{N_c} n_c \|(\overline{\mathbf{Y}}_c - \overline{\mathbf{Y}}) - (\overline{\mathbf{Y}}_c - \overline{\mathbf{Y}}) \times_k U^k U^{k^T}|_{k=1}^n\|^2.$$

This objective function is the same as Eq. (3) except that the samples are the weighted and centered class means $\{\sqrt{n_c}(\overline{\mathbf{Y}}_c - \overline{\mathbf{Y}}), c = 1, 2, \cdots, N_c\}$. Algorithm 2 can hence be used here to minimize the objective function by tensor element rearrangement.

After the pixel rearrangement process, tensor-based LDA is then conducted on the rearranged training data to learn the projection matrices $U^k|_{k=1}^n$. The data is projected to a lower dimension by the derived projection matrices, and then classified with a proper classifier. In this work, we use the Nearest Neighbor classifier for simplicity.

5. Experiments

We present a set of experiments to verify the effectiveness of the tensor element rearrangement algorithms for both unsupervised and supervised tasks. We focus on image input, namely 2^{nd} order tensors. For performance analysis on face image compression, we take as examples the two unsupervised learning algorithms Generalized Low Rank Approximations of Matrices (GLRAM) [17], which is a special case of Concurrent Subspace Analysis (CSA) for 2^{nd} order tensor input, and 2DPCA [16], which is a special case of GLRAM that computes only one projection matrix. For supervised learning, we take 2DLDA [18], which is the 2^{nd} order tensor-based version of LDA, as an example to examine performance on face recognition. When element rearrangement is included with these algorithms, they are labeled as 2DPCA-ER, GLRAM-ER, and 2DLDA-ER.

We use the CMU PIE [10] and FERET [8] databases for experiments. For the CMU PIE database, we choose five near frontal poses ($C27$, $C05$, $C29$, $C09$ and $C07$) and illumination indexed as 08, 11, 10 and 13. Due to incompleteness of data, only 63 persons are used in this work, with each person having 20 images. We also test our algorithm on a subset of the FERET database. This subset includes 1,400 images of 200 individuals (each with seven images labeled as ba, bc, bd, be, bf, bg, and bh). All the gray-level images are aligned by fixing the locations of the two eyes, normalizing in size to a resolution of 64×64 pixels, and preprocessing with histogram equilibrium. We also normalize the grayscale values by dividing by 255 and then subtracting 0.5. Typical examples before grayscale value normalization are given in the first row of Figs. 5 and 6.

5.1. Data Compression

For greater clarity, we set the reduced dimension m_1' to d for 2DPCA and 2DPCA-ER. Also, for GLRAM and GLRAM-ER we assume that the height and the width after dimensionality reduction are the same, such that $m_1' = m_2' = d$. We first take 2DPCA-ER and GLRAM-ER as examples to illustrate algorithmic convergence. In Fig. 2, we plot the *Root Mean Squared Error* (RMSE), defined as $\sqrt{\sum_{i=1}^{N} \|\mathbf{X}_i^t - \mathbf{X}_i^{t^{REC}}\|^2/N}$, for different numbers of iterations and $d = 5$. Convergence to a local minimum is evident for both algorithms.

In Fig. 3, we plot the element-rearranged original image data computed with GLRAM-ER on the CMU PIE database for different values of d. From the results, we can see that smaller values of d lead to greater correlation among the elements of each row/column vector. This is observed because most of the image information has been concentrated into the first d principal components along the row/column dimensions.

To examine data compression performance, we use in Fig. 4 the FERET and CMU PIE databases to compare the RMSEs of 2DPCA-ER to 2DPCA, and GLRAM-ER to GLRAM for different values of d. The proposed element rearrangement algorithm is shown to efficiently remove more redundancy, especially when d is small. When d is sufficiently large, the RMSEs from GLRAM-ER and 2DPCA-ER are very close to those of GLRAM and 2DPCA, since in these cases the RMSE without tensor element rearrangement is already very low.

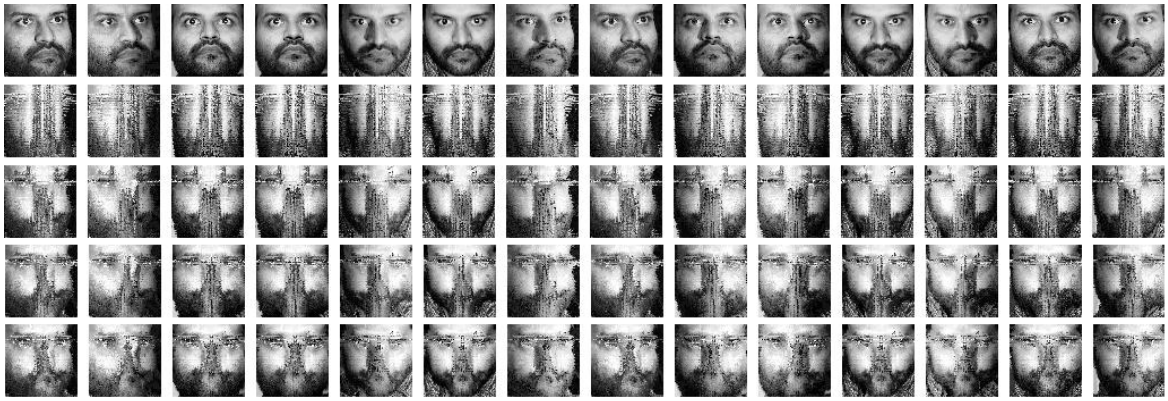
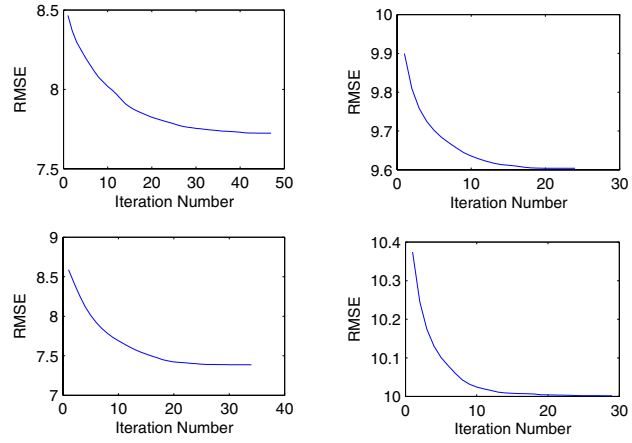
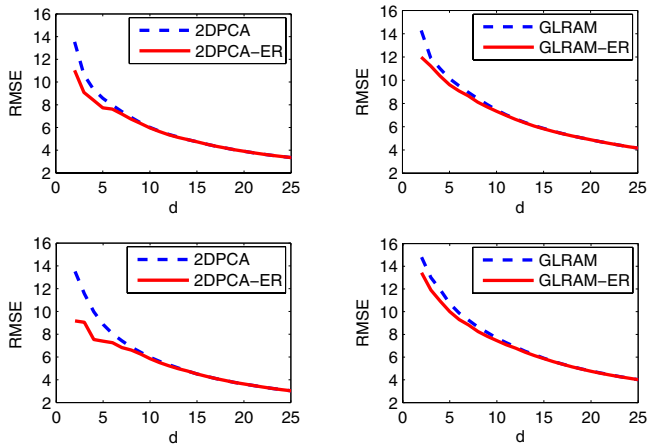


Figure 3. Comparison of fourteen element-rearranged images from the CMU PIE database based on the GLRAM-ER algorithm. The top row displays the original images. The subsequent rows show the rearranged images for d = 2, 3, 4, and 5, respectively.



2DPCA-ER GLRAM-ER

Figure 2. Algorithm convergence with element rearrangement. Root Mean Squared Error (RMSE) vs. number of iterations on the CMU PIE (top) and FERET (bottom) databases.



2DPCA vs. 2DPCA-ER GLRAM vs. GLRAM-ER

Figure 4. Root Mean Squared Error (RMSE) for different values of d on the CMU PIE (top) and FERET (bottom) databases.

Figs. 5 and 6 show the original images (first row), the reconstructed images from 2DPCA (second row), 2DPCA-ER (third row), GLRAM (fourth row), and GLRAM-ER (fifth row) on the FERET and CMU PIE databases (with $d = 5$). The results clearly show the reconstructed images from GLRAM-ER and 2DPCA-ER to be better than the images from GLRAM and 2DPCA in terms of visual quality and similarity to the original images.

Finally, we compare the RMSEs of PCA, 2DPCA, 2DPCA-ER, GLRAM and GLRAM-ER at various *compression ratios* (CR). CR is defined as Nm_1m_2/s with s as the number of scales required to represent the data. According to the work on GLRAM [17], for a given lower dimension d, we have $s = d(N + m_1 * m_2)$ for PCA, $s = d(N*m_2+m_1)$ for 2DPCA, and $s = (N*d+m_1+m_2)*d$ for GLRAM. For 2DPCA-ER and GLRAM-ER, the additional space s_{ad} for storing the element rearrangement index matrix must also be considered.

In our implementation, we employ a simple technique for compression of the index matrix, but more a sophisticated method could be used in its place. For a single 64*64 image, index values range from 1 to 4096, requiring 12 bits for encoding. We observe that for most tensor elements (more than 80% when d is small), the change in index from element rearrangement is relatively small, within 2^8 index values. Based on this observation, we use a 2-bit header to indicate four possible cases: *00* means that the element does not change in position, *01* means that the global index changes within 2^8 index values, *10* indicates the offset sign for the 01 case, and *11* means that the global index change is beyond 2^8. For case 01 the global index differences are encoded in eight bits, and for case 11 the full twelve bits are used to directly encode the new index after element rearrangement. Considering that 32 bits are needed to store the floating-point reconstruction error, we

Figure 6. Comparison of fourteen reconstructed images from the CMU PIE database. The top row displays the original images. The subsequent rows show reconstructed results from 2DPCA, 2DPCA-ER, GLRAM, and GLRAM-ER, respectively.

Figure 5. Comparison of seven reconstructed images from the FERET database. The top row displays the original images. The subsequent rows show reconstructed results from 2DPCA, 2DPCA-ER, GLRAM, and GLRAM-ER, respectively.

set $s_{ad} = (f_1 * 8 + f_2 * 12 + 2) * 4096/32$, where f_1 and f_2 are the percentages of pixels in case 01 and 11, respectively.

Fig. 7 plots the RMSE for different compression ratios. These results indicate better performance of GLRAM-ER than GLRAM, and also of 2DPCA-ER in comparison to 2DPCA. The performance of 2DPCA is the worst, likely because 2DPCA only removes redundancies among different rows, and does not remove redundancies among columns or along the person dimension [14] [17]. As observed in [17], GLRAM is not always better than the original PCA, possibly because GLRAM does not remove redundancies along the person dimension. As also noted in [16][17] , it is possible to apply PCA as a second-stage dimensionality reduction to remove more redundancy for 2DPCA, 2DPCA-ER, GLRAM and GLRAM-ER.

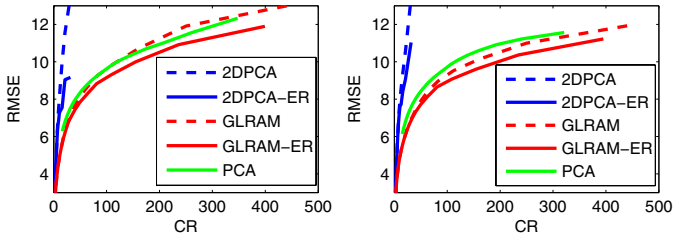


Figure 7. The Root Mean Squared Errors (RMSE) of different algorithms for different compression ratios on the FERET database (left) and the CMU PIE database (right).

5.2. Classification

To examine the effects of element rearrangement on classification performance, we compare 2DLDA-ER with 2DLDA on the CMU PIE and FERET databases. We also report the results from PCA, and LDA, and LDA is referred as 1DLDA here. For the CMU PIE database, four images per person are randomly chosen for training and the remaining sixteen images are used for testing. For the FERET database, the three images ba, bc and bh are used for training, and the other four images bd, be, bf and bg are used for testing.

A comparison of recognition rates is listed in Table 1, and recognition rates with respect to numbers of dimensions are plotted in Fig. 8. From these results, several observations can be made: 1) The tensor-based algorithm 2DLDA is usually better than 1DLDA for multi-view face recognition, which is consistent with findings in [15]; and 2) 2DLDA-ER demonstrates higher accuracy than 2DLDA, which supports the use of element rearrangement.

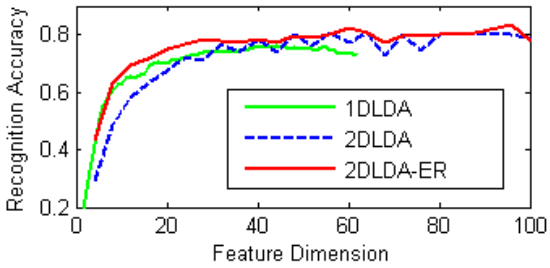


Figure 8. Top-one recognition accuracy on the CMU PIE database for LDA related algorithms.

Algorithm	FERET	CMU PIE
PCA [12]	45.1	43.3
1DLDA [1]	91.8	76.2
2DLDA [18]	94.8	81.9
2DLDA-ER	***95.6***	***83.7***

Table 1. The top-one recognition rates (%) on the CMU PIE and FERET databases

6. Conclusion

In this paper, we have studied the problem of how to rearrange tensor elements for better unsupervised and supervised subspace learning. For unsupervised learning, this problem was formulated to find a tensor element rearrangement operator that maximizes intra-tensor correlation. An approximate iterative solution based on a computationally feasible linear programming problem was proposed. In addition, the iterative algorithm was extended to supervised subspace learning problems for improvement of classification ability. The proposed algorithms have achieved encouraging results for both the unsupervised and supervised tasks.

There exist a number of potential directions for future work. Since the spatial structure of the original data objects are not preserved after rearrangement, the order and dimensions of the original tensors need not be maintained. How to rearrange elements into a more optimal tensor structure is an interesting topic for investigation. Another direction is to examine other classes of data objects that are structurally similar enough to benefit from element rearrangement. One possibility is human gait data, for which cues such as optical flow and motion estimation might additionally be used to guide the element rearrangement process.

Acknowledgment

This work was funded in part by the U.S. Government VACE program. The views and conclusions are those of the authors, not of the U.S. Government or its agencies.

References

[1] P. Belhumeur, J. Hespanha and D. Kriegman, *Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection*, IEEE Trans. PAMI, vol. 19, no. 7, pp. 711–720, 1997.

[2] H. Chen, H. Chang and T. Liu, *Local discriminant embedding and its variants*, Proc. IEEE CVPR, vol. 2, pp. 846–852, 2005.

[3] G. Dai and D. Yeung, *Tensor Embedding Methods*, Proceedings of National Conference on Artificial Intelligence, pp.330-335, 2006.

[4] A. Dempster, N. Laird and D. Rubin, *Maximum likelihood from incomplete data via the EM algorithm*. J. Royal Stat. Soc., vol. 39, pp. 1–38.

[5] P. Jensen and J. Bard, *Operations Research Models and Methods*, John Wiley and Sons, 2003.

[6] L. Lathauwer, B. Moor and J. Vandewalle, *A Multilinear Singular Value Decomposition*, SIAM Journal on Matrix Analysis and Applications, vol. 21, pp. 1253-1278, 2000.

[7] G. Nemhauser and L. Wolsey, *Integer and Combinatorial Optimization*, Wiley, New York, 1988.

[8] P. Phillips, H. Moon, S. Rizvi and P. Rauss, *The FERET Evaluation Methodology for Face-Recognition Algorithms*, IEEE Trans. PAMI, vol. 22, no. 10, pp. 1090-1104, 2000.

[9] Y. Rubner, C. Tomasi, and L. Guibas, *A Metric for Distributions with Applications to Image Databases*, Proc. IEEE ICCV, pp. 59–66, 1998.

[10] T. Sim, S. Baker and M. Bsat. *The CMU Pose, Illumination, and Expression Database*, IEEE Trans. PAMI, vol. 25, no. 12, pp. 1615–1618, 2003.

[11] E. Simoncelli and B. Olshausen, *Natural Image Statistics and Neural Representation*, Ann. Rev. Neuroscience, vol. 24, pp. 1193–1216, 2001.

[12] M. Turk and A. Pentland. *Face Recognition Using Eigenfaces*, Proc. IEEE CVPR, pp. 586–591, 1991.

[13] M.A.O. Vasilescu and D. Terzopoulos, *Multilinear Subspace Analysis for Image Ensembles*, Proc. IEEE CVPR, pp. 93–99, 2003.

[14] D. Xu, S.Yan, L. Zhang, H. Zhang, Z. Liu and H. Shum, *Concurrent Subspace Analysis*, Proc. IEEE CVPR, pp. 203–208, 2005.

[15] S. Yan, D. Xu, Q. Yang, L. Zhang, X. Tang and H. Zhang, *Discriminant Analysis with Tensor Representation*, Proc. IEEE CVPR, pp. 526–532, 2005.

[16] J. Yang, D. Zhang, A. Frangi and J. Yang, *Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition*, IEEE Trans. PAMI, vol. 26, no. 1, pp. 131–137, 2004.

[17] J. Ye, *Generalized Low Rank Approximations of Matrices*, Machine Learning, vol. 61, pp. 167–191, 2005.

[18] J. Ye, R. Janardan and Q. Li, *Two-Dimensional Linear Discriminant Analysis*, Advances in Neural Information Processing Systems, pp. 1569-1576, 2004.