Rank-R Approximation of Tensors Using Image-as-Matrix Representation

Hongcheng Wang, Narendra Ahuja
Beckman Institute, University of Illinois at Urbana-Champaign, USA
{*wanghc,ahuja* }*@vision.ai.uiuc.edu*

Abstract

We present a novel multilinear algebra based approach for reduced dimensionality representation of image ensembles. We treat an image as a matrix, instead of a vector as in traditional dimensionality reduction techniques like PCA, and higher-dimensional data as a tensor. This helps exploit spatio-temporal redundancies with less information loss than image-as-vector methods. The challenges lie in the computational and memory requirements for large ensembles. Currently, there exists a rank-R approximation algorithm which, although applicable to any number of dimensions, is efficient for only low-rank approximations. For larger dimensionality reductions, the memory and time costs of this algorithm become prohibitive. We propose a novel algorithm for rank-R approximations of third-order tensors, which is efficient for arbitrary R but for the important special case of 2D image ensembles, e.g. video. Both of these algorithms reduce redundancies present in all dimensions. Rank-R tensor approximation yields the most compact data representation among all known image-as-matrix methods. We evaluated the performance of our algorithm vs. other approaches on a number of datasets with the following two main results. First, for a fixed compression ratio, the proposed algorithm yields the best representation of image ensembles visually as well as in the least squares sense. Second, proposed representation gives the best performance for object classification.

1. Introduction

Many computer vision applications require processing large amounts of multidimensional data, such as face/object databases for recognition or retrieval, video sequences for security and surveillance, 3D CT and MRI images for medical analysis and 3D shape sequences for animation. The dimensions can be a mix of space and time, e.g., a video sequence where two of the dimensions are spatial and the third temporal. As data size and the amount of redundancy increase fast with dimensionality, it is desirable to obtain compact and concise representations of the data, e.g., by identifying suitable basis functions for subspace representation. The process of finding a set of compact bases and projections of the data on these bases as representation is referred to as dimensionality reduction.

For a large number of dimensions, dimensionality reduction is a computation and memory intensive process. The focus of this paper is on the development of a novel multilinear algebra algorithm which reduces redundancies in all dimensions by treating the data as a tensor (image in 2D case) and is time and memory efficient.

2. Related Work

Traditional methods for reducing the dimensionality of image ensembles usually transform an image into a vector by concatenating rows. One example of these methods is Principle Component Analysis (PCA), which has been widely used in face representation [7], face recognition [8] and many other applications. PCA is used to find a set of mutually orthogonal basis functions which capture the largest variation in the training data. The features are obtained by projecting each zero mean image onto a p dimensional subspace, which can be obtained by the Singular Value Decomposition (SVD). Vasilescu and Terzopoulos [10, 11] proposed using Higher-Order Orthogonal Iteration (HOOI) Algorithm given by De Lathauwer et al. [5]. They also apply this method to the image-as-vector representation. It was shown in [10] that the Root Mean Squared Error (RMSE) of the reconstruction for the same compression ratio is higher for HOOI than PCA. One inherent problem of the image-as-vector representation lies in that the spatial redundancies within each image matrix are not fully utilized,

0-7695-2372-2/05/$20.00 (c) 2005 IEEE

and some of the information about local spatial relationships is lost.

Realizing the intrinsic problem of matrix-as-vector formulation, some researchers in computer vision and machine learning have recently begun to treat an image as a matrix. Shashua and Levin [6] proposed representing a collection of images using a set of rank-1 matrices. Yang et al. [14] recently proposed a Two-Dimension PCA (2DPCA) by constructing an image covariance matrix using the original image matrices. As noted in [14], 2DPCA-based image representation is not as memory efficient as PCA in terms of storage requirements since 2DPCA requires more coefficients than PCA. Ye et al. [15, 16] proposed a method called Low Rank Approximation of Matrices (LRAM). In contrast to PCA, LRAM projects the original data onto a (p_1, p_2)-dimensional space such that the projection has the largest variance among all (p_1, p_2)-dimensional spaces.

Multilinear algebra has recently received broad attention in computer vision and signal processing. Higher-Order Singular Value Decomposition (HOSVD) has been used in computer vision applications such as face recognition [9] by Vasilescu et al. and facial expression decomposition [12] by Wang and Ahuja. Most recently, a Tensor Rank-One Decomposition algorithm (TROD) was proposed by Wang and Ahuja [13] for compact representation of multidimensional data (i.e. tensors). By iteratively finding the best approximation of the residual tensor in the least-squares sense, a higher-order tensor can be decomposed into a collection of rank-one tensors. This method significantly reduces the reconstruction RMSE for image ensembles compared to PCA for the same compression ratio. Best rank - $(R_1, R_2, \cdots, R_N)$ approximation of higher-order tensors has been theoretically studied by Lathauwer et al. [5]. This approach projects the tensor data onto a $R_1, R_2, \cdots, R_N$ - dimensional space with rank constraints. However, the time and space cost of this approach make it impractical for computer vision applications.

In this paper, we propose a novel rank-R tensor approximation approach using multilinear algebra. By projecting the tensor data $\mathcal{A} \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ onto the $(R_1, R_2, \cdots, R_N)$-dimensional space , we expect to obtain the closest approximation of the original tensor. The advantages of our method are as follows:

- We exploit redundancies of the data in all dimensions to obtain compact representation of high-order tensors.
- Rank-R approximation is more accurate for representing the data than the sum of R rank-1 approximations because the columns of bases in rank-R approximation are mutually orthonormal.
- The time and space complexities are lower compared to the existing algorithm for generalized rank-R tensor approximation. This is very important for large dataset applications.

The rest of the paper is organized as follows. We first give a brief overview of multilinear algebra and the formulation of Rank-R approximation of tensors in Section 3. Then we describe two algorithms for this problem: generalized rank-R approximation of tensors in Section 4 and rank-R approximation of third-order tensors in Section 5. In Section 6, we report experimental results on the quality and computational complexity of the representation, and its efficacy in object recognition. Conclusions are given in Section 7.

3. Rank-R Approximation of Tensors

3.1 Overview of Multilinear Algebra

In this section, we first introduce the relevant preliminary material concerning multilinear algebra. In the notation we use, matrices are denoted by italic capitals $(A, B, \cdots)$, and tensors by calligraphic letters $(\mathcal{A}, \mathcal{B}, \cdots)$.

A high-order tensor is denoted as: $\mathcal{A} \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$. The n-mode product of a tensor $\mathcal{A}$ by a matrix $U \in^{J_n \times I_n}$, denoted by $\mathcal{A} \times_n U$, is defined by a tensor with entries: $(\mathcal{A} \times_n U)_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_N} u_{j_n i_n}$. The scalar product of two tensors $\mathcal{A}, \mathcal{B} \in \Re^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as: $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} a_{i_1 i_2 \cdots i_N} b_{i_1 i_2 \cdots i_N}$. The Frobenius norm of a tensor $\mathcal{A} \in \Re^{I_1 \times I_2 \times \cdots \times I_N}$ is then defined as $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. The n-rank of $\mathcal{A}$, denoted by $R_n = rank_n(\mathcal{A})$, is the dimension of the vector space spanned by the n-mode vectors. For example, an Nth-order tensor has rank 1 when it equals the outer product of N vectors $U^1, U^2, \ldots, U^N$, i.e., $a_{i_1 i_2 \ldots i_N} = u^{(1)}_{i_1} u^{(2)}_{i_2} \ldots u^{(N)}_{i_N}$, for all values of the indices, written as: $\mathcal{A} = U^{(1)} \circ U^{(2)} \circ \cdots \circ U^{(N)}$. Unfolding a tensor $\mathcal{A}$ along the nth mode is denoted as *uf(A, n)*. References [2, 4, 5] are good sources of details of multilinear algebra.

Like SVD for matrices, HOSVD has been recently developed for tensors [4]. Any tensor $\mathcal{A}$ can be expressed as the product: $\mathcal{A} = \mathcal{S} \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_N U^{(N)}$, where, $\mathcal{S}$ is a $I_1 \times I_2 \times \ldots \times I_N$ tensor of which the subtensors $\mathcal{S}_{i_n = \alpha}$ have the properties of

Table 1. Comparison of Data Representation Using Different Methods

Method	Image As	Formulation	# of Scalars
PCA (p principle components)	vector	$\tilde{A}_i = \Phi \cdot P_i + \overline{A}$	$p(I_1 \cdot I_2 + I_3)$
Rank-1 (r rank-1 tensors)	matrix	$\tilde{\mathcal{A}} = \sum_{i=1}^{r} \lambda_i \cdot U_i^{(1)} \circ U_i^{(2)} \circ U_i^{(3)}$	$r(I_1 + I_2 + I_3 + 1)$
LRAM (dimension is d)	matrix	$\tilde{A}_i = L \cdot D_i \cdot R^T$	$d(I_1 + I_2 + I_3 d)$
Rank-R (dimension is R)	matrix	$\tilde{\mathcal{A}} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)}$	$R(R^2 + I_1 + I_2 + I_3)$

all-orthogonality and ordering based on the Frobenius-norms $\|\mathcal{S}_{i_n=\alpha}\|$, and $U^{(n)} = (U_1^{(n)} U_2^{(n)} \dots U_{I_n}^{(n)})$ is a unitary $(I_n \times I_n)$ matrix.

3.2 Problem Formulation

Given a real Nth-order tensor $\mathcal{A} \in \Re^{I_1 \times I_2 \times \dots \times I_N}$, find a tensor $\tilde{\mathcal{A}} \in \Re^{I_1 \times I_2 \times \dots \times I_N}$, having $\text{Rank}_1(\tilde{\mathcal{A}}) = R_1$, $\text{Rank}_2(\tilde{\mathcal{A}}) = R_2$, ..., $\text{Rank}_N(\tilde{\mathcal{A}}) = R_N$, that minimizes the least-squares cost function:

$$\tilde{\mathcal{A}} = \arg\min_{\hat{\mathcal{A}}} \left\| \mathcal{A} - \hat{\mathcal{A}} \right\|^2 \tag{1}$$

The desired tensor is represented as:

$$\tilde{\mathcal{A}} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times \dots \times_N U^{(N)} \tag{2}$$

where, $U^{(1)} \in \Re^{I_1 \times R_1}$, $U^{(2)} \in \Re^{I_2 \times R_2}$, $\dots$, $U^{(N)} \in \Re^{I_N \times R_N}$ and $\mathcal{B} \in \Re^{R_1 \times R_2 \times \dots \times R_N}$. $U^{(i)}$ has orthonormal columns for $1 \le i \le N$.

For image ensembles, we use third-order tensors, $\mathcal{A} \in \Re^{I_1 \times I_2 \times I_3}$ ($I_1 \times I_2$ is the dimensionality of each image, and I_3 is the number of images). We also assume $R_1 = R_2 = R_3 = R$ in this paper for simplicity. Hence we name our approach rank-R approximation of tensors.

3.3 Data Representation

Two major characteristics of our rank-R approximation approach are that (i) it treats the image as a matrix as opposed to simplifying it into a vector, and (ii) it reduces redundancies in all dimensions.

Representation of the data using PCA consists of p eigenvectors, $\Phi \in \Re^{I_1 I_2 \times p}$ and reduced representations $P \in \Re^{I_3 \times p}$. Tensor rank-one decomposition [13] consists of the projection λ corresponding to each rank-one tensor $U^{(1)} \circ U^{(2)} \circ \dots \circ U^{(N)}$. LRAM [15, 16] projects each image using two matrices $L \in \Re^{I_1 \times d}$ and $R \in \Re^{I_2 \times d}$, and the projection is $D \in \Re^{d \times d}$. For rank-R approximation, a tensor is approximated by a core tensor $\mathcal{B} \in \Re^{R \times R \times R}$ and three subspaces $U^{(1)} \in \Re^{I_1 \times R}, U^{(2)} \in \Re^{I_2 \times R}$ and $U^{(3) \in \Re^{I_3 \times R}}$. A comparison of these methods is given in Table 1, where $\overline{A}$ is the mean of the data.

In image ensemble applications, the number of images is usually much greater than the dimension d or R for dimensionality reduction, i.e., $I_3 >> d$ and $I_3 >> R$. Therefore, $(I_1 + I_2 + I_3 d)d > (R^2 + I_1 + I_2 + I_3)R$ if we assume $R = d$, i.e., the representation of original data using rank-R approximation is more compact than that using LRAM.

Rank-R approximation of tensors can be used to extract features of the image ensemble. By projecting the original tensor data onto $R_1, R_2, \dots, R_N$ axis system, we obtain a new tensor. By projecting it to any combination of two axes system, we define a feature of the slice along the plane defined by the two axes. The projections of a third-order tensor on any two axes are defined as: $\mathcal{B}_{xy} = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T}$, $\mathcal{B}_{yz} = \mathcal{A} \times_2 U^{(2)^T} \times_3 U^{(3)^T}$ and $\mathcal{B}_{xz} = \mathcal{A} \times_1 U^{(1)^T} \times_3 U^{(3)^T}$. Given a test image, which can be represented as a tensor $\mathcal{A}$ of size $I_1 \times I_2 \times 1$, the matrix B

$$B = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T} \tag{3}$$

can be used as a feature of the image.

4. Generalized Rank-R Approximation of Tensors

Recall the cost function from Equation 1, i.e.

$$f = \left\| \mathcal{A} - \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_N U^{(N)} \right\|^2 \tag{4}$$

For given matrices $U^{(1)}, U^{(2)}, \dots, U^{(N)}$, $\mathcal{B}$ can be obtained by solving a classical linear least-squares problem: $\mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \cdots \times_N U^{(N)} = \mathcal{A}$. Since $U^{(1)}, U^{(2)}, \dots, U^{(N)}$ have orthonormal columns, we have $\mathcal{B} = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T} \cdots \times_N U^{(N)^T}$. As stated in [5], the minimization in Equation 1 is equivalent to the maximization, over the matrices $U^{(1)}, U^{(2)}, \dots, U^{(N)}$ having orthonormal columns, of

Algorithm 1: Generalized Rank-R Approximation of Tensors

Data: Given $\mathcal{A}$ and R
Result: Find $U^k, (1 \le k \le N)$ and $\mathcal{B}$.

Initialize $U_0^{(k)} \in \Re^{I_k \times R_k}$, $1 \le k \le N$;
while $\sim$ *stop* **do**
$\tilde{U}_{j+1}^{(1)} = uf(\mathcal{A}, 1) \cdot kr(U_j^{(2)}, U_j^{(3)}, , \cdots, U_j^{(N)})$;
$U_{j+1}^{(1)} = svds(\tilde{U}_{j+1}^{(1)}, R)$;
$\tilde{U}_{j+1}^{(2)} = uf(\mathcal{A}, 2) \cdot kr(U_j^{(3)}, U_{j+1}^{(1)}, , \cdots, U_j^{(N)})$;
$U_{j+1}^{(2)} = svds(\tilde{U}_{j+1}^{(2)}, R)$;
$\cdots$
$\tilde{U}_{j+1}^{(N)} = uf(\mathcal{A}, N) \cdot kr(U_{j+1}^{(1)}, \cdots, U_{j+1}^{(N-1)})$;
$U_{j+1}^{(N)} = svds(\tilde{U}_{j+1}^{(N)}, R)$;
$\mathcal{B} = \mathcal{A} \times_1 U_{j+1}^{(1)^T} \times_2 U_{j+1}^{(2)^T} \times \cdots \times_N U_{j+1}^{(N)^T}$;
$if(\|\mathcal{B}_{j+1}\|^2 - \|\mathcal{B}_j\|^2 < \varepsilon)$ *stop*
end

the function:

$$g(U^{(1)}, \cdots, U^{(N)}) = \left\| \mathcal{A} \times_1 U^{(1)^T} \cdots \times_N U^{(N)^T} \right\|^2 \quad (5)$$

We apply the Alternative Least Squares (ALS) [3, 5] to find the (local) optimal solution of Equation 1. In each step, we optimize only one of the matrices, while keep others fixed. For example, with $U^{(1)}, \cdots, U^{(n-1)}, U^{(n+1)}, \cdots, U^{(N)}$ fixed, we project tensor $\mathcal{A}$ onto the $(R_1, \cdots, R_{n-1}, R_{n+1}, \cdots, R_N)$ - dimensional space, i.e., $U_{j+1}^{(n)} = \mathcal{A} \times {}_1U_{j+1}^{(1)^T} \times \cdots \times {}_{n-1}U_{j+1}^{(n-1)^T} \times {}_{n+1}U_j^{(n+1)^T} \times \cdots \times {}_NU_j^{(N)^T}$. and then the columns of $U^{(n)}$ can be found as an orthonormal basis for the dominant subspace of the projection. This equation can also be expressed in matrix format for the ease of implementation: $U_{j+1}^{(n)} = uf(\mathcal{A}, n) \cdot (U_{j+1}^{(1)} \otimes \cdots \otimes U_{j+1}^{(n-1)} \otimes U_j^{(n+1)} \otimes \cdots \otimes U_j^{(N)})$, where $\otimes$ represents Kronecker product, denoted as $kr(U_{j+1}^{(1)}, \cdots, U_{j+1}^{(n-1)}, U_j^{(n+1)}, \cdots, U_j^{(N)})$. The algorithm for rank-R approximation of tensors is described in Algorithm 1.

One issue in the implementation is the initialization of the algorithm. In the original algorithm proposed by [5], the values of $U^{(n)} \in \Re^{I_n \times R_n}$ were initialized with the truncation of the HOSVD. The columns of the column-wise orthogonal matrices span the space of the dominant left singular vectors of the matrix unfoldings $uf(\mathcal{A}, n)$ $(1 \le n \le N)$. While the computation of HOSVD is very expensive $(\mathcal{O}(I_1 I_2 \cdots I_N))$, we use $U_0^{(n)} = \left[\begin{array}{cc} I_{R_n} & 0 \end{array} \right]^T$ or $U_0^{(n)} =$ uniformly distributed random numbers (though columns are not necessarily orthonormal). Like many iterative search methods, empirically we have not seen any major difference in the results obtained using these initializations. However, our initializations are much simpler to compute than HOSVD.

Algorithm 2: Rank-R Approximation of Third-Order Tensors

Data: Given a third-order tensor, $\mathcal{A}$, and R
Result: Find $U^k, (1 \le k \le 3)$ and $\mathcal{B}$.

Initialize $U_0^{(k)} \in \Re^{I_k \times R_k}$, $1 \le k \le 3$;
while $\sim$ *stop* **do**
$M_{21} = \sum_{k=1}^{I_3} A_k^T U_j^{(1)} U_j^{(1)^T} A_k$;
$U_{j+1}^{(2)} = svds(M_{21}, R_2)$;
$M_{32} = \sum_{i=1}^{I_1} A_i^T U_{j+1}^{(2)} U_{j+1}^{(2)^T} A_i$;
$U_{j+1}^{(3)} = svds(M_{32}, R_3)$;
$M_{13} = \sum_{i=1}^{I_2} A_j U_{j+1}^{(3)} U_{j+1}^{(3)^T} A_j^T$;
$U_{j+1}^{(1)} = svds(M_{13}, R_1)$;
$\mathcal{B} = \mathcal{A} \times_1 U_{j+1}^{(1)^T} \times_2 U_{j+1}^{(2)^T} \times_3 U_{j+1}^{(3)^T}$;
$if(\|\mathcal{B}_{j+1}\|^2 - \|\mathcal{B}_j\|^2 < \varepsilon)$ *stop*
end

Algorithm 1 has both advantages and disadvantages. Most of the time spent in the algorithm is in the computation of eigenvectors using SVD. The time complexity of SVD on a $r \times c$ matrix is $\mathcal{O}(rc \min(r, c))$. Therefore, the total time cost is $\mathcal{O}\left\{\max\{I_n R^2 \min(I_n, R^2)\}\right\}$ $(1 \le n \le N)$, which means it is efficient for low-rank approximation, i.e. when R is small. Moreover, it provides a generalized framework for approximating tensors of any order. On the other hand, as R increases, the time and memory requirements increase fast making the algorithm inefficient for large R.

5. Rank-R Approximation of Third-Order Tensors

Algorithm 1 is designed for approximating tensors of any order. In this section, we consider the important special case of image ensembles, which are third-order tensors. We present a specific algorithm for any rank approximation of third-order tensors, called slice projection.

Our approach is inspired by the work of Ye [15, 16]. The basic idea of slice projection for rank-R approxi-

mation of tensors is similar to Algorithm 1 in that a tensor is transformed into matrices for the convenience of manipulation. In Algorithm 1, a tensor is unfolded along different coordinate axes to formulate matrices, while here a third-order tensor is represented as slices along the three coordinate axes, A_i, A_j and A_k, where $1 \le i \le I_1$, $1 \le j \le I_2$ and $1 \le k \le I_3$. Each slice is represented by a matrix orthogonal to that direction. By projecting the slices along each direction to two corresponding coordinate axes under the rank constraints, we can find the best approximation of the original slices. We need to maximize the norm of the best approximation of original tensor, which corresponds to maximizing the summation of the norms of the slice projections in three directions. Then our problem is formulated as follows: given a tensor $\mathcal{A}$ (hence A_i, A_j and A_k), find $U^{(1)}, U^{(2)}$ and $U^{(3)}$ which solve

$$\max_{U^{(1)},U^{(2)},U^{(3)}} \quad \sum_{i=1}^{I_1} \left\| U^{(2)^T} A_i U^{(3)} \right\|^2 + \sum_{j=1}^{I_2} \left\| U^{(1)^T} A_j U^{(3)} \right\|^2 + \sum_{k=1}^{I_3} \left\| U^{(1)^T} A_k U^{(2)} \right\|^2 \quad (6)$$

where $U^{(1)} \in \Re^{I_1 \times R_1}$, $U^{(2)} \in \Re^{I_2 \times R_2}$ and $U^{(3)} \in \Re^{I_3 \times R_3}$ have orthonormal columns; $1 \le i \le I_1$, $1 \le j \le I_2$ and $1 \le k \le I_3$. The following theorem describes how to find a locally optimal solution using an iterative procedure.

Theorem 1 (Slice-Projection Theorem). *Let $U^{(1)}, U^{(2)}$ and $U^{(3)}$ be the optimal solution to the maximization problem in Equation 6. Then*

- *Given $U^{(1)}$ and $U^{(2)}$, $U^{(3)}$ consists of the R_3 eigenvectors of the matrix $M_{31} = \sum_{j=1}^{I_2} A_j^T U^{(1)} U^{(1)^T} A_j$ (and $M_{32} = \sum_{i=1}^{I_1} A_i^T U^{(2)} U^{(2)^T} A_i$) corresponding to the largest R_3 eigenvalues.*
- *Given $U^{(1)}$ and $U^{(3)}$, $U^{(2)}$ consists of the R_2 eigenvectors of the matrix $M_{23} = \sum_{i=1}^{I_1} A_i U^{(3)} U^{(3)^T} A_i^T$ (and $M_{21} = \sum_{k=1}^{I_3} A_k^T U^{(1)} U^{(1)^T} A_k$) corresponding to the largest R_2 eigenvalues.*
- *Given $U^{(2)}$ and $U^{(3)}$, $U^{(1)}$ consists of the R_1 eigenvectors of the matrix $M_{12} = \sum_{k=1}^{I_3} A_k U^{(2)} U^{(2)^T} A_k^T$ (and $M_{13} = \sum_{i=1}^{I_2} A_j U^{(3)} U^{(3)^T} A_j^T$) corresponding to the largest R_1 eigenvalues.*

Given $U^{(1)}$, $U^{(2)}$ and $U^{(3)}$, the projection of $\mathcal{A}$ onto the coordinate axes is represented as: $\mathcal{B} = \mathcal{A} \times U^{(1)^T} \times U^{(2)^T} \times U^{(3)^T}$.

Proof. Given U_1 and U_2, U_3 maximizes

$$\max_{U^{(3)}} \sum_{j=1}^{I_2} \left\| U^{(1)^T} A_j U^{(3)} \right\|^2 \text{ and } \sum_{i=1}^{I_1} \left\| U^{(2)^T} A_i U^{(3)} \right\|^2 \quad (7)$$

The first term in Equation 7 can be rewritten as

$$\begin{aligned} & \sum_{j=1}^{I_2} trace(U^{(3)^T} A_j^T U^{(1)} U^{(1)^T} A_j U^{(3)}) \\ = \; & trace(U^{(3)^T} (\sum_{j=1}^{I_2} A_j^T U^{(1)} U^{(1)^T} A_j) U^{(3)}) \\ = \; & trace(U^{(3)^T} M_{31} U^{(3)}). \end{aligned}$$

which is maximal for a given $U^{(1)}$ only if $U^{(3)} \in \Re^{I_3 \times R_3}$ consists of the R_3 eigenvectors of the matrix M_{31} corresponding to the largest R_3 eigenvalues. From the second term in Equation 7, we obtain $U^{(3)}$ which maximizes $trace(U^{(3)^T} M_{32} U^{(3)})$. In either case, $U^{(3)}$ is locally optimal for the maximization of Equation 7. Similarly, we can show other parts of the theorem.

□

This theorem provides us with an iterative procedure to find $U^{(1)}, U^{(2)}$ and $U^{(3)}$. By updating $U^{(1)}, U^{(2)}$ and $U^{(3)}$ iteratively, the procedure will converge to a (local) maximum of Equation 6. This is described in Algorithm 2. The advantage of Algorithm 2 is that it is time and memory efficient for any rank approximation of third-order tensors since the cost for finding the eigenvectors is only $\mathcal{O}(I_1^3 + I_2^3 + I_3^3)$, but the tradeoff is that this algorithm only works for third-order tensors. Our algorithm is more general than that proposed by Ye et al. [15, 16], since they consider the projection along only the temporal axis while our algorithm achieves reduction along both spatial and temporal axes. Figure 1 contrasts the different projection schemes used by these two approaches.

6. Experimental Results

In this section, we experimentally evaluate the performance of our proposed algorithm with respect to the quality of representation, computational complexity, and efficacy in object classification. The datasets we use include: a toy video of 129, 129×129 frames and two face databases: ORL dataset[1] of 400, 92×112

[1] `http://www.uk.research.att.com/facedatabase.html`

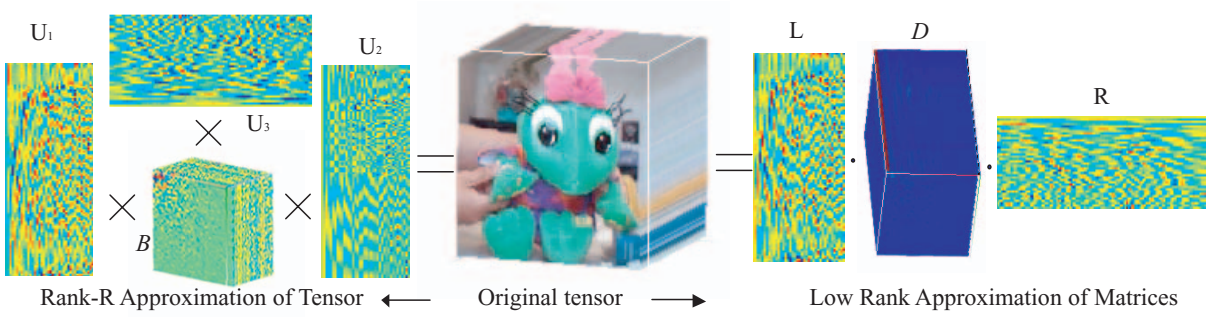

Figure 1. Illustration of Rank-R Approximations of Tensors vs. Low Rank Approximations of Matrices by Ye [15]. Dimension $R = 40$ and $d = 40$.

images and Yale dataset[2] of 165, 82×66 images. For the Yale dataset, all images have been centered, aligned using eyes positions, and cropped.

6.1 Compact Data Representation

We applied PCA, LRAM, Rank-1 Decomposition and Rank-R Approximation of Tensors methods on different datasets (We excluded HOSVD because it is stated in [10] that the reconstruction error is larger for HOSVD than PCA). The parameters we used to achieve a constant compression ratio for different algorithms are given in Table 1. The compression ratio is defined as: $\alpha = \frac{I_1 \times I_2 \times I_3}{s}$, where s is the number of scalars required for representation of the data (see Table 1). We compute the RMSE error of the reconstructed data with respect to the original data:

$$RMSE = \sqrt{\frac{1}{I_3} \left\| \mathcal{A} - \tilde{\mathcal{A}} \right\|^2}$$

as a measure of relative performance of the algorithms.

Figure 2(a-c) shows the reconstructions obtained using the same compression ratio but different representation methods. For the toy sequence (Figure 2(a)), the temporal redundancy is very strong since the scene does not change much from frame to frame, though spatial redundancies also exist. Interestingly, the reconstruction using LRAM is visually even worse than that using PCA. The reconstruction using LRAM is very blurry, and has some features (e.g. eyes) missing (Figure 2(a2)). This is because LRAM captures more redundancies in the spatial than in the temporal dimension, while PCA captures mostly the temporal redundancies in this case. The reconstructions are much better for tensor rank-one decomposition and rank-R approximation methods since both methods capture the redundancies in all dimensions. Moreover, since the basis columns of rank-R approximation are orthonormal (therefore more compact), rank-R approximation of tensors yields much better reconstruction than tensor rank-one decomposition. For the face datasets (Figure 2(b,c)), PCA is the worst among all methods since the spatial redundancies are not well captured due to its image-as-vector formulation. The critical features like eyes, mouth and nose begin to appear in Figure 2(b2,b3), and become pretty clear in Figure 2(b4). For the compression ratio $\alpha = 77 : 1$ (corresponding to using five principle components for PCA), the reconstruction using our algorithm (Figure 2(c4)) is visually quite close to the original image (Figure 2(e)). Figure 2(d-f) shows the reconstruction error for each dataset. These plots are consistent with the relative visual quality of the reconstructions. Our algorithm produces the best visual reconstructions as well as the smallest RMSE of all methods. As the compression ratio decreases, all four methods give similar performance. This is reasonable since the use of increasing number of components leads to steadily decreasing amount of information loss.

6.2 Computational Efficiency

Our experiments were performed using Matlab on a Pentium IV 1.5GHz machine with 512M RAM. Figure 3(a) gives convergence rates and shows that it usually takes 4 iterations for Algorithm 1 to converge to a locally optimal solution while it takes 2 iterations for Algorithm 2. Figure 3(b) gives the computation times for the toy sequence. We can see that for comparable representation errors (Figure 3(c)), the computation time of Algorithm 1 is slightly less than that of Algorithm 2 when $R < 12$; however, Algorithm 2 is much more efficient than Algorithm 1 when $R > 12$ due to both the high computational complexity and large

[2] `http://cvc.yale.edu/projects/yalefaces/yalefaces.html`

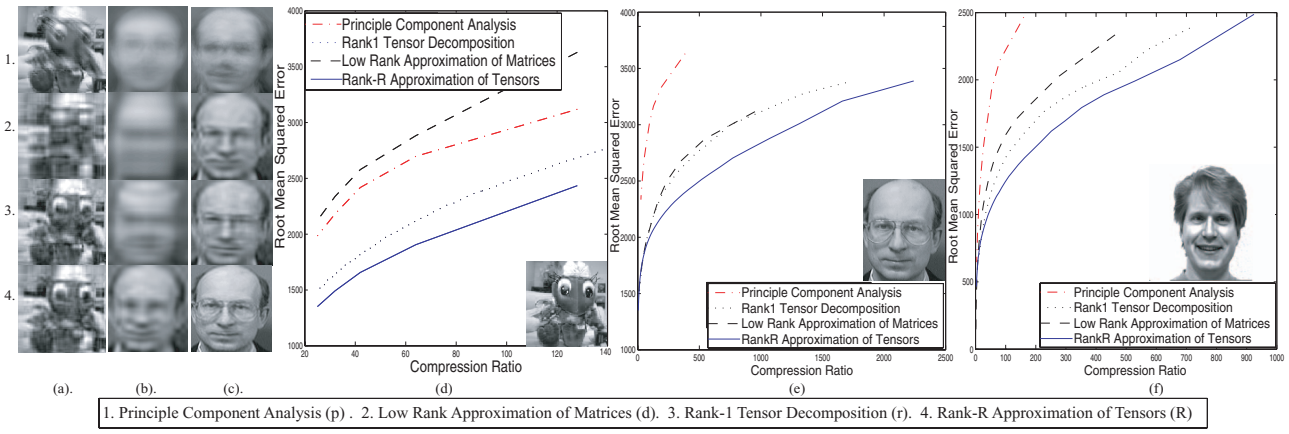

Figure 2. Comparison of the quality of reconstructions achieved using Rank-R approach and other methods. The parameters used for each method were chosen so as to achieve the same compression ratio. (a): Reconstruction results for the toy sequence $(p = 1, d = 10, r = 43, R = 20, \alpha = 128 : 1)$. Original image is in (d); (b): Reconstruction results for ORL $(p = 1, d = 5, r = 18, R = 14, \alpha = 385 : 1)$; (c): Reconstruction results for ORL $(p = 5, d = 11, r = 88, R = 32, \alpha = 77 : 1)$. Original image is in (e); Reconstruction error for (d) toy sequence; (e) ORL; (f) Yale.

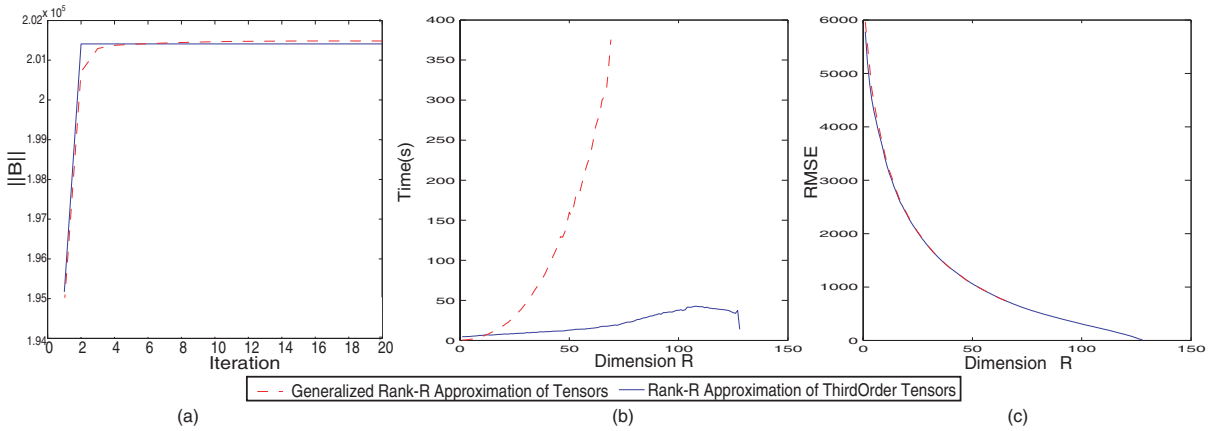

Figure 3. Comparison of Generalized Rank-R Approximation of Tensors vs. Rank-R Approximation of Third-Order Tensors. (a). Convergence; (b). Execution time; (c). Reconstruction error.

memory requirements of Algorithm 1 (Figure 3(b)). From Figure 3(b,c), we see that $R > 70$ is not applicable to Algorithm 1 for this data since the machine does not have the required memory for $R > 70$. The experiments on other datasets such as Yale and ORL databases give similar results.

The results we present here are based on a direct implementation of Algorithm 1. Better implementations will help to improve its efficiency. For example, we could exploit the fact that the SVD of $\tilde{U}^{(n)}\tilde{U}^{(n)^T}$ is $U^{(n)}S^2U^{(n)^T}$ to obtain U^n [1], if $\tilde{U}^{(n)} \in \Re^{m \times n}$ is an unfolded tensor with $m \ll n$, which will reduce the computational time.

6.3 Appearance-Based Recognition

To evaluate the quality of the representation obtained by our algorithm, we applied it to appearance-based recognition in face images in ORL and Yale face databases. For the Yale database (containing 11 different images of each of 15 distinct persons), we used 11-fold cross validation, i.e., we randomly selected ten images per person, and used the remaining one for testing. For the ORL face database (containing 10 different images of each of 40 distinct persons), we applied 10-fold cross validation. Similarly, we randomly selected nine images from each class as a training set, and used

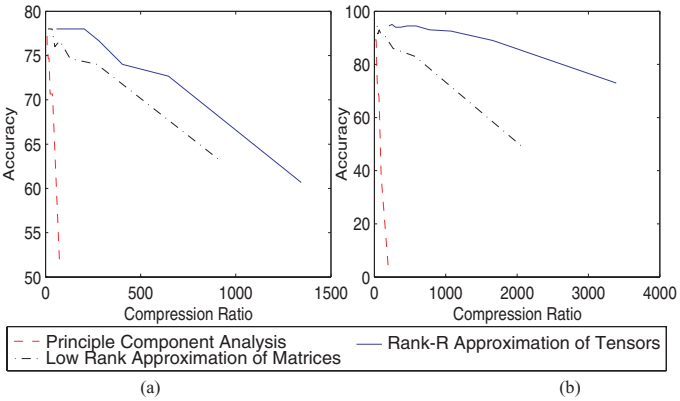

Figure 4. Comparing the quality of representation of the different methods on face recognition for the Yale and ORL face datasets. (a). accuracy vs. compression ratio on Yale. (b). accuracy vs. compression ratio on ORL.

the remaining image for testing. For each database, we repeated the process ten times. Features can be obtained using Equation 3. The face is correctly classified if its feature has the minimum Frobenius distance from the features of the same person. The reported final accuracy is the average of the ten runs.

We compared our algorithm with LRAM and PCA. As shown in Figure 4, comparing the performance of different methods for a fixed compression ratio, our method yields the highest accuracy in every case (Figure 4). This shows that our algorithm has the best reduced dimensionality representation.

7. Conclusion

We have introduced a novel approximation of tensors based on multilinear algebra. The method is designed to capture the spatial and temporal redundancies along each dimension of the tensor. Experiments show superior performance of rank-R approximation of tensors in terms of quality of data representation and object classification accuracy for a fixed compression ratio.

The support of the Office of Naval Research under grant N00014-03-1-0107 is gratefully acknowledged. Thanks go as well to the anonymous reviewers for their informative comments that strengthened this paper.

References

[1] G. Golub and C. V. Loan. *Matrix Computation*. The Johns Hopkins University Press, Baltimore, MD, USA, third edition, 1996.

[2] E. Kofidis and P. A. Regalia. On the besrt rank-1 approximation of higher-order supersymmetric tensors. *SIAM J. Matrix Anal. Appl.*, 23(3):863–884, 2002.

[3] P. Kroonenberg. *Three-mode principal component analysis*. DSWO Press, Leiden, The Netherlands, 1983.

[4] L. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

[5] L. Lathauwer, B. D. Moor, and J. Vandewalle. On the best rank-1 and rank-$(R_1, R_2, \ldots, R_N)$ approximation of high-order tensors. *SIAM J. Matrix Anal. Appl.*, 21(4):1324–1342, 2000.

[6] A. Shashua and A. Levin. Linear image regression and classification using the tensor-rank principle. *CVPR*, 2001.

[7] L. Sirovich and M. Kirby. Low-dimensional procedure for the characterization of human faces. *Journal of Optical Society of America*, 4(3):519–524, 1987.

[8] M. Turk and A. Pentland. Eigen faces for recognition. *J. of Cognitive Neuroscience*, 1991.

[9] M. A. O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. *ECCV'02*, pages 447–460, 2002.

[10] M. A. O. Vasilescu and D. Terzopoulos. Multilinear subspace analysis of image ensembles. *CVPR'03*, June 2003.

[11] M. A. O. Vasilescu and D. Terzopoulos. Tensortextures: Multilinear image-based rendering. *Proc. ACM SIGGRAPH 2004*, pages 336–342, August 2004.

[12] H. Wang and N. Ahuja. Facial expression decomposition. *Int. Conf. on Computer Vision, ICCV'03*, pages 958–965, 2003.

[13] H. Wang and N. Ahuja. Compact representation of multidimensional data using tensor rank-one decomposition. *Int. Conf. on Pattern Recognition, ICPR'04*, 1:44–47, 2004.

[14] J. Yang, D. Zhang, A. F. Frangi, and J. Yang. Two-dimensional pca: A new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(1), January 2004.

[15] J. Ye. Generalized low rank approximations of matrices. *International Conference on Machine Learning, ICML'04*, July 2004.

[16] J. Ye, R. Janardan, and Q. Li. GPCA: An efficient dimension reduction scheme for image compression and retrieval. *ACM KDD'04*, pages 354–363, August 2004.