Multiscale Analysis for Higher-order Tensors

Alp Ozdemir, Ali Zare, Mark A. Iwen, and Selin Aviyente

Abstract

The widespread use of multisensor technology and the emergence of big datasets have created the need to develop tools to reduce, approximate, and classify large and multimodal data such as higher-order tensors. While early approaches focused on matrix and vector based methods to represent these higher-order data, more recently it has been shown that tensor decomposition methods are better equipped to capture couplings across their different modes. For these reasons, tensor decomposition methods have found applications in many different signal processing problems including dimensionality reduction, signal separation, linear regression, feature extraction, and classification. However, most of the existing tensor decomposition methods are based on the principle of finding a low-rank approximation in a linear subspace structure, where the definition of the rank may change depending on the particular decomposition. Since many datasets are not necessarily low-rank in a linear subspace, this often results in high approximation errors or low compression rates. In this paper, we introduce a new adaptive, multi-scale tensor decomposition method for higher order data inspired by hybrid linear modeling and subspace clustering techniques. In particular, we develop a multi-scale higher-order singular value decomposition (MS-HoSVD) approach where a given tensor is first permuted and then partitioned into several sub-tensors each of which can be represented as a low-rank tensor with increased representational efficiency. The proposed approach is evaluated for dimensionality reduction and classification for several different real-life tensor signals with promising results.

Index Terms

Higher-order singular value decomposition, tensor decomposition, multi-scale decomposition, data reduction, big data applications.

I. Introduction

Data in the form of multidimensional arrays, also referred to as tensors, arise in a variety of applications including chemometrics, hyperspectral imaging, high resolution videos, neuroimaging, biometrics and social network analysis [1]–[3]. These applications produce massive amounts of data collected in various forms with multiple aspects and high dimensionality. Tensors, which are multi-dimensional generalizations of matrices, provide a useful representation for such data. A crucial step in many applications involving higher-orders tensors is multiway reduction of the data to ensure that the reduced representation of the tensor retains certain characteristics. Early multiway data analysis approaches reformatted the tensor data as a matrix and resorted to methods developed for classical two-way analysis. However, one cannot discover hidden components within multiway data using conventional matrix decomposition methods as matrix based representations cannot capture multiway couplings focusing on standard pairwise interactions. To this end, many different types of tensor decomposition methods have been proposed in literature [4]–[9].

In contrast to the matrix case where data reduction is often accomplished via low-rank representations such as singular value decomposition (SVD), the notion of rank for higher order tensors is not uniquely defined. The CANDECOMP/PARAFAC (CP) and Tucker decompositions are two of the most widely used tensor decomposition methods for data reduction [10], [11]. For CP, the goal is to approximate the given tensor as a weighted sum of rank-1 tensors, where a rank-1 tensor refers to the outer product of n vectors with n being equal to the order of the tensor. The Tucker model allows for interactions between the factors from the different modes resulting in a typically dense, but small, core tensor. This model also introduces the notion of Tucker rank or n-rank, which refers to the n-tuple of ranks corresponding to the tensor's unfoldings along each of its modes. Therefore, low rank approximation with the Tucker model can be obtained by projections onto low-rank factor matrices. Unlike the CP decomposition, the Tucker decomposition is in general non-unique. To help obtain meaningful and unique representations by the Tucker decomposition, orthogonality, sparsity, and non-negativity constraints are often imposed on the factors yielding, e.g., the Non-Negative Tensor Factorization (NTF) and the Sparse Non-Negative Tucker Decomposition [12]–[14]. The Tucker decomposition with orthogonality constraints on the factors is known as the Higher-Order Singular Value Decomposition (HoSVD), or Multilinear SVD [11]. The HoSVD can be computed by simply flattening the tensor in each mode and calculating the n-mode singular vectors corresponding to that mode.

With the emergence of multidimensional big data, classical tensor representation and decomposition methods have become inadequate since the size of these tensors exceeds available working memory and the processing time is very long. In order to

A. Ozdemir (ozdemira@egr.msu.edu) and S. Aviyente (aviyente@egr.msu.edu) are with the Electrical and Computer Engineering, Michigan State University, East Lansing, MI, 48824, USA.

Ali Zare (zareali@msu.edu) is with the Department of Computational Mathematics, Science, and Engineering (CMSE), Michigan State University, East Lansing, MI, 48824, USA.

Mark A. Iwen (markiwen@math.msu.edu) is with the Department of Mathematics, and the Department of Computational Mathematics, Science, and Engineering (CMSE), Michigan State University, East Lansing, MI, 48824, USA.

This work was in part supported by NSF DMS-1416752 and NSF CCF-1615489.

address the problem of large-scale tensor decomposition, several block-wise tensor decomposition methods have been proposed [6]. The basic idea is to partition a big data tensor into smaller blocks and perform tensor related operations block-wise using a suitable tensor format. Preliminary approaches relied on a hierarchical tree structure and reduced the storage of d-dimensional arrays to the storage of auxiliary three-dimensional ones such as the tensor-train decomposition (T-Train), also known as the matrix product state (MPS) decomposition, [5] and the Hierarchical Tucker Decomposition (H-Tucker) [15]. In particular, in the area of large volumetric data visualization, tensor based multiresolution hierarchical methods such as TAMRESH have attracted attention [16]. However, all of these methods are interested in fitting a low-rank model to data which lies near a *linear* subspace, thus being limited to learning linear structure.

Similar to the research efforts in tensor reduction, low-dimensional subspace and manifold learning methods have also been extended for higher order data clustering and classification applications. In early work in the area, Vasilescu and Terzopoulos [17] extended the eigenface concept to the tensorface by using higher order SVD and taking different modes such as expression, illumination and pose into account. Similarly, 2D-PCA for matrices has been used for feature extraction from face images without converting the images into vectors [18]. He et al. [19] extended locality preserving projections [20] to second order tensors for face recognition. Dai and Yeung [21] presented generalized tensor embedding methods such as the extensions of local discriminant embedding methods [22], neighborhood preserving embedding methods [23], and locality preserving projection methods [20] to tensors. Li et al. [24] proposed a supervised manifold learning method for vector type data which preserves local structures in each class of samples, and then extended the algorithm to tensors to provide improved performance for face and gait recognition. Similar to vector-type manifold learning algorithms, the aim of these methods is to find an optimal *linear* transformation for the tensor-type training data samples without vectorizing them and mapping these samples to a low dimensional subspace while preserving the neighborhood information.

In this paper, we propose a novel multi-scale analysis technique to efficiently approximate tensor type data using locally linear low-rank approximations. The proposed method consists of two major steps: 1) Constructing a tree structure by partitioning the tensor into a collection of permuted subtensors, followed by 2) Constructing multiscale dictionaries by applying HoSVD to each subtensor. The contributions of the proposed framework and the novelty in the proposed approach with respect to previously published work in [25], [26] are manifold. They include: 1) The introduction of a more flexible multi-scale tensor decomposition method which allows the user to approximate a given tensor within given memory and processing power constraints; 2) the introduction of theoretical error bounds for the proposed decomposition; 3) the introduction of adaptive pruning to achieve a better trade-off between compression rate and reconstruction error for the developed factorizations; 4) the extensive evaluation of the method for both data reduction and classification applications; and 5) a detailed comparison of the proposed method to state-of-the-art tensor decomposition methods including the HoSVD, T-Train, and H-Tucker decompositions.

The remainder of the paper is organized as follows. In Section II, basic notation and tensor operations are reviewed. The proposed multiscale tensor decomposition method along with theoretical error bounds are then introduced in Section III. Sections IV and V illustrate the results of applying the proposed framework to data reduction and classification problems, respectively.

II. Background

A. Tensor Notation and Algebra

A multidimensional array with N modes $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is called a tensor, where $x_{i_1,i_2,..i_N}$ denotes the $(i_1, i_2, ..i_N)^{th}$ element of the tensor $\mathcal{X}$. The vectors in $\mathbb{R}^{I_n}$ obtained by fixing all of the indices of such a tensor $\mathcal{X}$ except for the one that corresponds to its nth mode are called its *mode-n fibers*. Let $[N] := \{1, \ldots, N\}$ for all $N \in \mathbb{N}$. Basic tensor operations are reviewed below (see, e.g., [10], [27], [28]).

Tensor addition and multiplication by a scalar: Two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ can be added using component-wise tensor addition. The resulting tensor $\mathcal{X} + \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ has its entries given by $(\mathcal{X} + \mathcal{Y})_{i_1,i_2,..i_N} = x_{i_1,i_2,..i_N} + y_{i_1,i_2,..i_N}$. Similarly, given a scalar $\alpha \in \mathbb{R}$ and a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ the rescaled tensor $\alpha\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ has its entries given by $(\alpha\mathcal{X})_{i_1,i_2,..i_N} = \alpha\ x_{i_1,i_2,..i_N}$.

Mode-n products: The mode-n product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \ldots I_n \times \ldots \times I_N}$ and a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$ is denoted as $\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}$, $(\mathcal{Y})_{i_1,i_2,\ldots,i_{n-1},j,i_{n+1},\ldots,i_N} = \sum_{i_n=1}^{I_n} x_{i_1,\ldots,i_n,\ldots,i_N} u_{j,i_n}$. It is of size $I_1 \times ... \times I_{n-1} \times J \times I_{n+1} \times ... \times I_N$. The following facts about mode-n products are useful (see, e.g., [10], [28]).

Lemma 1. *Let $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$, $\alpha, \beta \in \mathbb{R}$, and $\mathbf{U}^{(n)}, \mathbf{V}^{(n)} \in \mathbb{R}^{J_n \times I_n}$ for all $n \in [N]$. The following are true:*

(a) $(\alpha\mathcal{X} + \beta\mathcal{Y}) \times_n \mathbf{U}^{(n)} = \alpha\left(\mathcal{X} \times_n \mathbf{U}^{(n)}\right) + \beta\left(\mathcal{Y} \times_n \mathbf{U}^{(n)}\right)$.

(b) $\mathcal{X} \times_n \left(\alpha\mathbf{U}^{(n)} + \beta\mathbf{V}^{(n)}\right) = \alpha\left(\mathcal{X} \times_n \mathbf{U}^{(n)}\right) + \beta\left(\mathcal{X} \times_n \mathbf{V}^{(n)}\right)$.

(c) If $n \neq m$ then $\mathcal{X} \times_n \mathbf{U}^{(n)} \times_m \mathbf{V}^{(m)} = \left(\mathcal{X} \times_n \mathbf{U}^{(n)}\right) \times_m \mathbf{V}^{(m)} = \left(\mathcal{X} \times_m \mathbf{V}^{(m)}\right) \times_n \mathbf{U}^{(n)} = \mathcal{X} \times_m \mathbf{V}^{(m)} \times_n \mathbf{U}^{(n)}$.

(d) If $\mathbf{W} \in \mathbb{C}^{P \times J_n}$ then $\mathcal{X} \times_n \mathbf{U}^{(n)} \times_n \mathbf{W} = \left(\mathcal{X} \times_n \mathbf{U}^{(n)}\right) \times_n \mathbf{W} = \mathcal{X} \times_n \left(\mathbf{W}\mathbf{U}^{(n)}\right) = \mathcal{X} \times_n \mathbf{W}\mathbf{U}^{(n)}$.

Tensor matricization: The process of reordering the elements of the tensor into a matrix is known as matricization or unfolding. The mode-n matricization of a tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is denoted as $\mathbf{Y}_{(n)} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$ and is obtained by arranging $\mathcal{Y}$'s

mode-n fibers to be the columns of the resulting matrix. Unfolding the tensor $\mathcal{Y} = \mathcal{X} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} ... \times_N \mathbf{U}^{(N)} =: \mathcal{X} \bigtimes_{n=1}^{N} \mathbf{U}^{(n)}$ along mode-n is equivalent to

$$\mathbf{Y}_{(n)} = \mathbf{U}^{(n)} \mathbf{X}_{(n)} (\mathbf{U}^{(N)} \otimes ... \mathbf{U}^{(n+1)} \otimes \mathbf{U}^{(n-1)} ... \otimes \mathbf{U}^{(1)})^\top, \tag{1}$$

where $\otimes$ is the matrix Kronecker product. In particular, (1) implies that the matricization $\left(\mathcal{X} \times_n \mathbf{U}^{(n)}\right)_{(n)} = \mathbf{U}^{(n)} \mathbf{X}_{(n)}$.[1]

It is worth noting that trivial inner product preserving isomorphisms exist between a tensor space $\mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ and any of its matricized versions (i.e., mode-n matricization can be viewed as an isomorphism between the original tensor vector space $\mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ and its mode-n matricized target vector space $\mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$). In particular, the process of matricizing tensors is linear. If, for example, $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ then one can see that the mode-n matricization of $\mathcal{X} + \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ is $(\mathcal{X} + \mathcal{Y})_{(n)} = \mathbf{X}_{(n)} + \mathbf{Y}_{(n)}$ for all modes $n \in [N]$.

Tensor Rank: Unlike matrices, which have a unique definition of rank, there are multiple rank definitions for tensors including *tensor rank* and *tensor n-rank*. The *rank* of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times ... I_n \times ... \times I_N}$ is the smallest number of rank-one tensors that form $\mathcal{X}$ as their sum. The *n-rank* of $\mathcal{X}$ is the collection of ranks of unfoldings $\mathbf{X}_{(n)}$ and is denoted as:

$$n\text{-rank}(\mathcal{X}) = \left(\text{rank}(\mathbf{X}_{(1)}),\ \text{rank}(\mathbf{X}_{(2)}), ...,\ \text{rank}(\mathbf{X}_{(N)})\right). \tag{2}$$

Tensor inner product: The inner product of two same sized tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ is the sum of the products of their elements.

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} ... \sum_{i_N=1}^{I_N} x_{i_1, i_2, ..., i_N} y_{i_1, i_2, ..., i_N}. \tag{3}$$

It is not too difficult to see that matricization preserves Hilbert-Schmidt/Frobenius matrix inner products. That is, that $\langle \mathcal{X}, \mathcal{Y} \rangle = \left\langle \mathbf{X}_{(n)}, \mathbf{Y}_{(n)} \right\rangle_{\text{F}} = \text{Trace}\left(\mathbf{X}_{(n)}^\top \mathbf{Y}_{(n)}\right)$ holds for all $n \in [N]$. If $\langle \mathcal{X}, \mathcal{Y} \rangle = 0$, $\mathcal{X}$ and $\mathcal{Y}$ are *orthogonal*.

Tensor norm: Norm of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ is the square root of the sum of the squares of all its elements.

$$\| \mathcal{X} \| = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle} = \sqrt{\sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} ... \sum_{i_N=1}^{I_N} x_{i_1, i_2, ..., i_N}^2}. \tag{4}$$

The fact that matricization preserves Frobenius matrix inner products also means that it preserves Frobenius matrix norms. As a result we have that $\|\mathcal{X}\| = \left\|\mathbf{X}_{(n)}\right\|_{\text{F}}$ holds for all $n \in [N]$. If $\mathcal{X}$ and $\mathcal{Y}$ are orthogonal and also have unit norm (i.e., have $\|\mathcal{X}\| = \|\mathcal{Y}\| = 1$) we will say that they are an *orthonormal* pair.

B. Some Useful Facts Concerning Mode-n Products and Orthogonality

Let $\mathbf{I} \in \mathbb{R}^{I_n \times I_n}$ be the identity matrix. Given a (low rank) orthogonal projection matrix $\mathbf{P} \in \mathbb{R}^{I_n \times I_n}$ one can decompose any given tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ into two orthogonal tensors using Lemma 1 (b)

$$\mathcal{X} = \mathcal{X} \times_n \mathbf{I} = \mathcal{X} \times_n ((\mathbf{I} - \mathbf{P}) + \mathbf{P}) = \mathcal{X} \times_n (\mathbf{I} - \mathbf{P}) + \mathcal{X} \times_n \mathbf{P}.$$

To check that the last two summands are orthogonal one can use (1) to compute that

$$\langle \mathcal{X} \times_n (\mathbf{I} - \mathbf{P}), \mathcal{X} \times_n \mathbf{P} \rangle = \left\langle (\mathbf{I} - \mathbf{P}) \mathbf{X}_{(n)}, \mathbf{P} \mathbf{X}_{(n)} \right\rangle_{\text{F}} = \text{Trace}\left(\mathbf{X}_{(n)}^\top (\mathbf{I} - \mathbf{P}) \mathbf{P} \mathbf{X}_{(n)}\right) = 0.$$

As a result one can also verify that the Pythagorean theorem holds, i.e., that $\|\mathcal{X}\|^2 = \|\mathcal{X} \times_n \mathbf{P}\|^2 + \|\mathcal{X} \times_n (\mathbf{I} - \mathbf{P})\|^2$.

If we now regard $\mathcal{X} \times_n \mathbf{P}$ as a low rank approximation to $\mathcal{X}$ then we can see that its approximation error

$$\mathcal{X} - \mathcal{X} \times_n \mathbf{P} = \mathcal{X} \times_n (\mathbf{I} - \mathbf{P})$$

is orthogonal to the low rank approximation $\mathcal{X} \times_n \mathbf{P}$, as one would expect. Furthermore, the norm of its approximation error satisfies $\|\mathcal{X} \times_n (\mathbf{I} - \mathbf{P})\|^2 = \|\mathcal{X}\|^2 - \|\mathcal{X} \times_n \mathbf{P}\|^2$. By continuing to use similar ideas in combination with lemma 1 for all modes one can prove the following more general Pythagorean result (see, e.g., theorem 5.1 in [28]).

Lemma 2. *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ and $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ be an orthogonal projection matrix for all $n \in [N]$. Then,*

$$\left\| \mathcal{X} - \mathcal{X} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} ... \times_N \mathbf{U}^{(N)} \right\|^2 =: \left\| \mathcal{X} - \mathcal{X} \bigtimes_{n=1}^{N} \mathbf{U}^{(n)} \right\|^2 = \sum_{n=1}^{N} \left\| \mathcal{X} \bigtimes_{h=1}^{n-1} \mathbf{U}^{(h)} \times_n \left(\mathbf{I} - \mathbf{U}^{(n)}\right) \right\|^2.$$

[1] Simply set $\mathbf{U}^{(m)} = \mathbf{I}$ (the identity) for all $m \neq n$ in (1). This fact also easily follows directly from the definition of the mode-n product.

C. The Higher Order Singular Value Decomposition (HoSVD)

Any tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ can be decomposed as mode products of a core tensor $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ with N orthogonal matrices $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ each of which is composed of the left singular vectors of $\mathbf{X}_{(n)}$ [11]:

$$\mathcal{X} = \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_N \mathbf{U}^{(N)} = \mathcal{C} \bigtimes_{n=1}^{N} \mathbf{U}^{(n)} \tag{5}$$

where $\mathcal{C}$ is computed as

$$\mathcal{C} = \mathcal{X} \times_1 \left(\mathbf{U}^{(1)}\right)^\top \times_2 \left(\mathbf{U}^{(2)}\right)^\top \ldots \times_N \left(\mathbf{U}^{(N)}\right)^\top. \tag{6}$$

Let $\mathcal{C}_{i_n=\alpha}$ be a subtensor of $\mathcal{C}$ obtained by fixing the nth index to α. This subtensor satisfies the following properties:

- all-orthogonality: $\mathcal{C}_{i_n=\alpha}$ and $\mathcal{C}_{i_n=\beta}$ are orthogonal for all possible values of n, α and β subject to $\alpha \neq \beta$.

$$\langle \mathcal{C}_{i_n=\alpha},\, \mathcal{C}_{i_n=\beta} \rangle = 0 \; when \; \alpha \neq \beta. \tag{7}$$

- ordering:

$$\| \mathcal{C}_{i_n=1} \| \geqslant \| \mathcal{C}_{i_n=2} \| \geqslant \ldots \geqslant \| \mathcal{C}_{i_n=I_n} \| \geqslant 0 \tag{8}$$

  for $n \in [N]$.

III. Multiscale Analysis of Higher-order Datasets

In this section, we present a new tensor decomposition method named Multiscale HoSVD (MS-HoSVD) for an Nth order tensor, $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$. The proposed method recursively applies the following two-step approach: (i) Low-rank tensor approximation, followed by (ii) Partitioning the residual (original minus low-rank) tensor into subtensors.

A tensor $\mathcal{X}$ is first decomposed using HoSVD as follows:

$$\mathcal{X} = \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \ldots \times_N \mathbf{U}^{(N)}, \tag{9}$$

where the $\mathbf{U}^{(n)}$'s are the left singular vectors of the unfoldings $\mathbf{X}_{(n)}$. The low-rank approximation of $\mathcal{X}$ is obtained by

$$\hat{\mathcal{X}}_0 = \mathcal{C}_0 \times_1 \hat{\mathbf{U}}^{(1)} \times_2 \hat{\mathbf{U}}^{(2)} \ldots \times_N \hat{\mathbf{U}}^{(N)} \tag{10}$$

where $\hat{\mathbf{U}}^{(n)} \in \mathbb{R}^{I_n \times r_n}$s are the truncated matrices obtained by keeping the first r_n columns of $\mathbf{U}^{(n)}$ and $\mathcal{C}_0 = \mathcal{X} \times_1 \left(\hat{\mathbf{U}}^{(1)}\right)^\top \times_2 \left(\hat{\mathbf{U}}^{(2)}\right)^\top \ldots \times_N \left(\hat{\mathbf{U}}^{(N)}\right)^\top$. The multilinear-rank of $\hat{\mathcal{X}}_0$, $\{r_1, \ldots, r_N\}$, can either be given a *priori*, or an energy criterion can be used to determine the minimum number of singular values to keep along each mode as:

$$r_n = \arg\min_i \sum_{l=1}^{i} \sigma_l^{(n)} \quad s.t. \quad \frac{\sum_{l=1}^{i} \sigma_l^{(n)}}{\sum_{l=1}^{I_n} \sigma_l^{(n)}} > \tau, \tag{11}$$

where $\sigma_l^{(n)}$ is the lth singular value of the matrix obtained from the SVD of the unfolding $\mathbf{X}_{(n)}$, and τ is an energy threshold. Once $\hat{\mathcal{X}}_0$ is obtained, the tensor $\mathcal{X}$ can be written as

$$\mathcal{X} = \hat{\mathcal{X}}_0 + \mathcal{W}_0, \tag{12}$$

where $\mathcal{W}_0$ is the residual tensor.

For the first scale analysis, to better encode the details of $\mathcal{X}$, we adapted an idea similar to the one presented in [25], [29]. The 0^{th} scale residual tensor, $\mathcal{W}_0$ is first decomposed into subtensors as follows. $\mathcal{W}_0 \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ is unfolded across each mode yielding $\mathbf{W}_{0,(n)} \in \mathbb{R}^{I_n \times \prod_{j \neq n} I_j}$ whose columns are the mode-n fibers of $\mathcal{W}_0$. For each mode, rows of $\mathbf{W}_{0,(n)}$ are partitioned into c_n non-overlapping clusters using a clustering algorithm such as local subspace analysis (LSA) [30] in order to encourage the formation of new subtensors which are intrinsically lower rank, and therefore better approximated via a smaller HoSVD at the next scale. The Cartesian product of the partitioning labels coming from the N modes yields $K = \prod_{i=1}^{N} c_i$ disjoint subtensors $\mathcal{X}_{1,k}$ where $k \in [K]$.

Let J_0^n be the index set corresponding to the nth mode of $\mathcal{W}_0$ with $J_0^n = [I_n]$, and let $J_{1,k}^n$ be the index set of the subtensor $\mathcal{X}_{1,k}$ for the nth mode, where $J_{1,k}^n \subset J_0^n$ for all $k \in [K]$ and $n \in [N]$. Index sets of subtensors for the nth mode satisfy $\bigcup_{k=1}^{K} J_{1,k}^n = J_0^n$ for all $n \in [N]$. The kth subtensor $\mathcal{X}_{1,k} \in \mathbb{R}^{|J_{1,k}^1| \times |J_{1,k}^2| \times \cdots \times |J_{1,k}^N|}$ is obtained by

$$\begin{gathered} \mathcal{X}_{1,k}(i_1,\, i_2, \ldots,\, i_N) = \mathcal{W}_0(J_{1,k}^1(i_1),\, J_{1,k}^2(i_2), \ldots,\, J_{1,k}^N(i_N)), \\ \mathcal{X}_{1,k} = \mathcal{W}_0(J_{1,k}^1 \times J_{1,k}^2 \times \ldots \times J_{1,k}^N), \end{gathered} \tag{13}$$

where $i_n \in \left[\left|J_{1,k}^n\right|\right]$. Low-rank approximation for each subtensor is obtained by applying HoSVD as:

$$\hat{\mathcal{X}}_{1,k} = \mathcal{C}_{1,k} \times_1 \hat{\mathbf{U}}^{(1)}_{1,k} \times_2 \hat{\mathbf{U}}^{(2)}_{1,k} ... \times_N \hat{\mathbf{U}}^{(N)}_{1,k}, \tag{14}$$

where $\mathcal{C}_{1,k}$ and $\hat{\mathbf{U}}^{(n)}_{1,k} \in \mathbb{R}^{|J^n_{1,k}| \times r^{(n)}_{1,k}}$s correspond to the core tensor and low-rank projection basis matrices of $\mathcal{X}_{1,k}$, respectively. We can then define $\hat{\mathcal{X}}_1$ as the 1^{st} scale approximation of $\mathcal{X}$ formed by mapping all of the subtensors onto $\hat{\mathcal{X}}_{1,k}$ as follows:

$$\hat{\mathcal{X}}_1(J^1_{1,k} \times J^2_{1,k} \times ... \times J^n_{1,k}) = \hat{\mathcal{X}}_{1,k}. \tag{15}$$

Similarly, 1^{st} scale residual tensor is obtained by

$$\mathcal{W}_1(J^1_{1,k} \times J^2_{1,k} \times ... \times J^n_{1,k}) = \mathcal{W}_{1,k}, \tag{16}$$

where $\mathcal{W}_{1,k} = \mathcal{X}_{1,k} - \hat{\mathcal{X}}_{1,k}$. Therefore, $\mathcal{X}$ can be rewritten as:

$$\mathcal{X} = \hat{\mathcal{X}}_0 + \mathcal{W}_0 = \hat{\mathcal{X}}_0 + \hat{\mathcal{X}}_1 + \mathcal{W}_1. \tag{17}$$

Continuing in this fashion the j^{th} scale approximation of $\mathcal{X}$ is obtained by partitioning $\mathcal{W}_{j-1,k}$s into subtensors $\mathcal{X}_{j,k}$s and fitting a low-rank model to each one of them in a similar fashion. Finally, the j^{th} scale decomposition of $\mathcal{X}$ can be written as:

$$\mathcal{X} = \sum_{i=0}^{j} \hat{\mathcal{X}}_i + \mathcal{W}_j. \tag{18}$$

Algorithm 1 describes the pseudo code for this approach and Fig. 1 illustrates 1-scale MS-HoSVD.

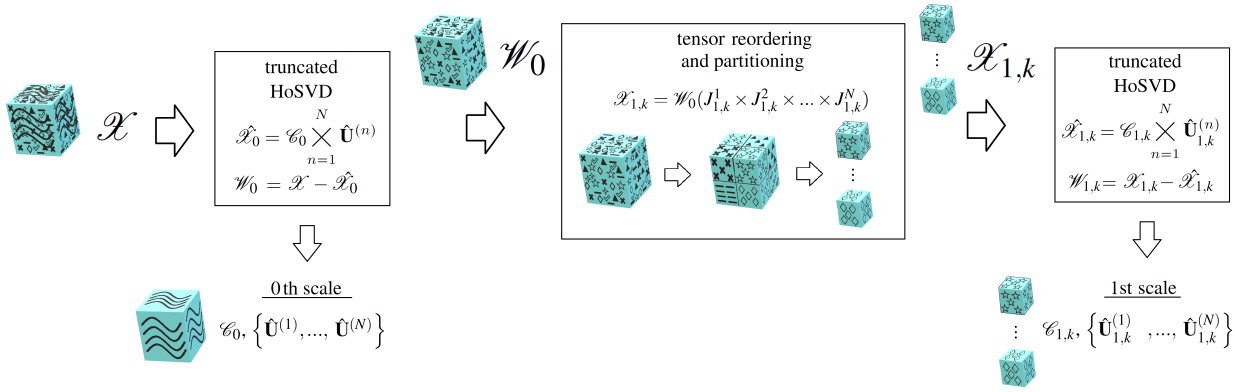


Fig. 1: Illustration of 1-scale MS-HoSVD. Higher scale decomposition can be obtained by applying the illustrated approach to the residual tensors recursively.

Algorithm 1 Multiscale HoSVD

Input: $\mathcal{X}$: tensor , $\mathbf{C} = (c_1,\ c_2,\ ...,\ c_N)$: the desired number of clusters for each mode, s_H: the highest scale of MS-HoSVD.
Output: T: Tree structure containing the MS-HoSVD decomposition of $\hat{\mathcal{X}}$.
Create an empty tree T
Create an empty list L
Add the node containing $\mathcal{X} =: \mathcal{X}_{0,1}$ to L with $\text{Parent}(0,1) = \varnothing$ (i.e., this is the root of the the tree).
while L is not empty. **do**
    Pop a node corresponding to $\mathcal{X}_{s,t}$ (the tth subtensor from sth scale) from the list L where $s \in \{0, ..., s_H\}$ and $t \in \{1, ..., K^s\}$.
    $\mathcal{C}_{s,t}, \left\{\hat{\mathbf{U}}^{(n)}_{s,t}\right\} \leftarrow \text{truncatedHOSVD}(\mathcal{X}_{s,t})$.
    Add the node containing $\mathcal{C}_{s,t}, \left\{\hat{\mathbf{U}}^{(n)}_{s,t}\right\}$ to T as a child of $\text{Parent}(s,t)$.
    if $s < s_H$ **then**
        Compute $\mathcal{W}_{s,t} = \mathcal{X}_{s,t} - \hat{\mathcal{X}}_{s,t}$.
        Create K subtensors $\mathcal{X}_{s+1,K(t-1)+k}$ with $J^n_{s+1,K(t-1)+k}$ from $\mathcal{W}_{s,t}$ where $k \in \{1,\ 2,\ ...,\ K\}$ and $n \in \{1,\ 2,\ ...,\ N\}$.
        Add K nodes containing $\mathcal{X}_{s+1,K(t-1)+k}$ and $\left\{J^n_{s+1,K(t-1)+k}\right\}$ to L with $\text{Parent}(s+1, K(t-1)+k) = (s,t)$.
    end if
end while

A. Memory Cost of the First Scale Decomposition

Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times .... \times I_N}$ be an Nth order tensor. To simplify the notation, assume that the dimension of each mode is the same, i.e. $I_1 = I_2 = ... = I_N = I$. Assume $\mathcal{X}$ is approximated by HoSVD as:

$$\hat{\mathcal{X}} = \mathcal{C}_H \times_1 \mathbf{U}_H^{(1)} \times_2 \mathbf{U}_H^{(2)} ... \times_N \mathbf{U}_H^{(N)}, \tag{19}$$

by fixing the rank of each mode matrix as $\text{rank}(\mathbf{U}_H^{(i)}) = r_H$ for $i \in \{1,\ 2, ...,\ N\}$. Let $\mathbb{F}(\cdot)$ be a function that quantifies the memory cost, then the storage cost of $\mathcal{X}$ decomposed by HoSVD is $\mathbb{F}(\mathcal{C}_H) + \sum_{i=1}^{N}(\mathbb{F}(\mathbf{U}_H^{(i)})) \approx r_H^N + NIr_H$.

For multiscale analysis at scale 1, $\hat{\mathcal{X}} = \hat{\mathcal{X}}_0 + \hat{\mathcal{X}}_1$. The cost of storing $\hat{\mathcal{X}}_0$ is $\mathbb{F}(\mathcal{C}_0) + \sum_{i=1}^{N}(\mathbb{F}(\hat{\mathbf{U}}^{(i)})) \approx r_0^N + NIr_0$ where the rank of each mode matrix is fixed at $\text{rank}(\mathbf{U}^{(i)}) = r_0$ for $i \in \{1,\ 2, ...,\ N\}$. The cost of storing $\hat{\mathcal{X}}_1$ is the sum of the storage costs for each of the $K = \prod_{i=1}^{N} c(i)$ subtensors $\hat{\mathcal{X}}_{1,k}$. Assume $c(i) = c$ for all $i \in \{1,\ 2, ...,\ N\}$ yielding c^N equally sized subtensors, and that each $\hat{\mathcal{X}}_{1,k}$ is decomposed using the HoSVD as $\hat{\mathcal{X}}_{1,k} = \mathcal{C}_{1,k} \times_1 \hat{\mathbf{U}}_{1,k}^{(1)} \times_2 \hat{\mathbf{U}}_{1,k}^{(2)} ... \times_N \hat{\mathbf{U}}_{1,k}^{(N)}$. Let the rank of each mode matrix be fixed as $\text{rank}(\hat{\mathbf{U}}_{1,k}^{(i)}) = r_1$ for all $i \in \{1,\ 2, ...,\ N\}$ and $k \in \{1,\ 2, ...,\ K\}$. Then, the memory cost for the first scale is $\sum_{k=1}^{K}\left(\mathbb{F}(\mathcal{C}_{1,k}) + \sum_{i=1}^{N}\mathbb{F}(\hat{\mathbf{U}}_{1,k}^{(i)})\right) \approx c^N\left(r_1^N + \frac{NIr_1}{c}\right)$. Choosing $r_1 \lesssim \frac{r_0}{c^{(N-1)}}$ ensures that the storage cost does not grow exponentially so that $\mathbb{F}(\hat{\mathcal{X}}_1) < \mathbb{F}(\hat{\mathcal{X}}_0)$ since the total cost becomes approximately equal to $r_0^N\left(1 + \frac{1}{c^{N^2-2N}}\right) + 2NIr_0$. Thus, picking $r_0 \approx r_H/2$ can now provide lower storage cost for the first scale analysis than for HoSVD.

B. Computational Complexity

The computational complexity of MS-HoSVD at the first scale is equal to the sum of computational complexity of computing HoSVD at the parent node, partitioning into subtensors and computing HoSVD for each one of the subtensors. Computational complexity of HoSVD of an N-way tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times ... \times I_N}$ where $I_1 = I_2 = ... = I_N = I$ is $\mathcal{O}\left(NI^{(N+1)}\right)$ [31]. By assuming that the partitioning is performed using K-means (via Lloyd's algorithm) with $c_i = c$ along each mode, the complexity partitioning along each mode is $\mathcal{O}\left(NI^N ci\right)$, where i is the number of iterations used in Lloyd's algorithm. Finally, the total complexity of applying the HoSVD to c^N equally sized subtensors is $\mathcal{O}\left(c^N N(I/c)^{(N+1)}\right)$. Therefore, first scale MS-HoSVD has a total computational complexity of $\mathcal{O}\left(NI^{(N+1)} + NI^N ci + c^N N(I/c)^{(N+1)}\right)$. Note that this complexity is similar to that of the HoSVD whenever ci is small compared to I. The runtime complexity of these multiscale methods can be reduced even further by computing the HoSVDs for different subtensors in parallel whenever possible, as well as by utilizing distributed and parallel SVD algorithms such as [32] when computing all the required HoSVD decompositions.

C. A Linear Algebraic Representation of the Proposed Multiscale HoSVD Approach

Though the tree-based representation of the proposed MS-HoSVD approach used above in Algorithm 1 is useful for algorithmic development, it is somewhat less useful for theoretical error analysis. In this subsection we will develop formulas for the proposed MS-HoSVD approach which are more amenable to error analysis. In the process, we will also formulate a criterion which, when satisfied, guarantees that the proposed fist scale MS-HoSVD approach produces an accurate multiscale approximation to a given tensor.

Preliminaries: We can construct full size first scale subtensors of the residual tensor $\mathcal{W}_0 \in \mathbb{R}^{I_1 \times I_2 \times ... I_N}$ from (12), $\mathcal{X}|_k \in \mathbb{R}^{I_1 \times I_2 \times ... I_N}$ for all $k \in [K]$, using the index sets $J_{1,k}^n$ from (13) along with diagonal restriction matrices. Let $\mathbf{R}_k^{(n)} \in \{0,\ 1\}^{I_n \times I_n}$ be the diagonal matrix with entries given by

$$\mathbf{R}_k^{(n)}(i,j) = \begin{cases} 1, & \text{if } i = j, \text{ and } j \in J_{1,k}^n \\ 0, & \text{otherwise} \end{cases} \tag{20}$$

for all $k \in [K]$, and $n \in [N]$. We then define

$$\mathcal{X}|_k := \mathcal{W}_0 \bigtimes_{n=1}^{N} \mathbf{R}_k^{(n)} = \mathcal{W}_0 \times_1 \mathbf{R}_k^{(1)} \times_2 \mathbf{R}_k^{(2)} ... \times_N \mathbf{R}_k^{(N)}. \tag{21}$$

Thus, the kth subtensor $\mathcal{X}|_k$ will only have nonzero entries, given by $\mathcal{W}_0(J_{1,k}^1 \times ... \times J_{1,k}^N)$, in the locations indexed by the sets $J_{1,k}^n$ from above. The properties of the index sets $J_{1,k}^n$ furthermore guarantee that these subtensors all have disjoint support. As a result both

$$\mathcal{W}_0 = \sum_{k=1}^{K} \mathcal{X}|_k \tag{22}$$

and

$$\langle \mathcal{X}|_k, \mathcal{X}|_j \rangle = 0 \text{ for all } j, k \in [K] \text{ with } j \neq k$$

will always hold.

Recall that we want to compute the HoSVD of the subtensors we form at each scale in order to create low-rank projection basis matrices along the lines of those in (14). Toward this end we compute the top $r_k^{(n)} \leqslant rank(\mathbf{R}_k^{(n)}) = |J_{1,k}^n|$ left singular vectors of the mode-n matricization of each $\mathcal{X}|_k$, $\mathbf{X}|_{\mathbf{k}_{(n)}} \in \mathbb{R}^{I_n \times \prod_{m \neq n} I_m}$, for all $n \in [N]$. Note that $\mathbf{X}|_{\mathbf{k}_{(n)}} = \mathbf{R}_k^{(n)} \mathbf{X}|_{\mathbf{k}_{(n)}}$ always holds for these matricizations since $\mathbf{R}_k^{(n)}$ is a projection matrix.[2] Thus, the top $r_k^{(n)}$ left singular vectors of $\mathbf{X}|_{\mathbf{k}_{(n)}}$ will only have nonzero entries in locations indexed by $J_{1,k}^n$. Let $\hat{\mathbf{U}}_k^{(n)} \in \mathbb{R}^{I_n \times r_k^{(n)}}$ be the matrix whose columns are these top singular vectors. As a result of the preceding discussion we can see that $\hat{\mathbf{U}}_k^{(n)} = \mathbf{R}_k^{(n)} \hat{\mathbf{U}}_k^{(n)}$ will hold for all $n \in [N]$ and $k \in [K]$. Our low rank projection matrices $\mathbf{Q}_k^{(n)} \in \mathbb{R}^{I_n \times I_n}$ used to produce low rank approximations of each subtensor $\mathcal{X}|_k$ can now be defined as

$$\mathbf{Q}_k^{(n)} := \hat{\mathbf{U}}_k^{(n)} \left(\hat{\mathbf{U}}_k^{(n)} \right)^\top. \tag{23}$$

As a consequence of $\hat{\mathbf{U}}_k^{(n)} = \mathbf{R}_k^{(n)} \hat{\mathbf{U}}_k^{(n)}$ holding, combined with the fact that $\left(\mathbf{R}_k^{(n)} \right)^\top = \mathbf{R}_k^{(n)}$ since each $\mathbf{R}_k^{(n)}$ matrix is diagonal, we have that

$$\mathbf{Q}_k^{(n)} := \hat{\mathbf{U}}_k^{(n)} \left(\hat{\mathbf{U}}_k^{(n)} \right)^\top = \mathbf{R}_k^{(n)} \hat{\mathbf{U}}_k^{(n)} \left(\mathbf{R}_k^{(n)} \hat{\mathbf{U}}_k^{(n)} \right)^\top = \mathbf{R}_k^{(n)} \hat{\mathbf{U}}_k^{(n)} \left(\hat{\mathbf{U}}_k^{(n)} \right)^\top \left(\mathbf{R}_k^{(n)} \right)^\top = \mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} \tag{24}$$

holds for all $n \in [N]$ and $k \in [K]$. Using (24) combined with the fact that $\mathbf{R}_k^{(n)}$ is a projection matrix we can further see that

$$\mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} = \mathbf{R}_k^{(n)} \left(\mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} \right) = \mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} = \mathbf{Q}_k^{(n)} = \mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} = \left(\mathbf{R}_k^{(n)} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} \right) \mathbf{R}_k^{(n)} = \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} \tag{25}$$

also holds for all $n \in [N]$ and $k \in [K]$.

1-scale Analysis of MS-HoSVD: Using this linear algebraic formulation we are now able to re-express the the 1^{st} scale approximation of $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$, $\hat{\mathcal{X}}_1 \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$, as well as the 1^{st} scale residual tensor tensor, $\mathcal{W}_1 \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$, as follows (see (15) – (17)). We have that

$$\begin{aligned}
\hat{\mathcal{X}}_1 = \sum_{k=1}^{K} \left(\mathcal{X}|_k \bigtimes_{n=1}^{N} \mathbf{Q}_k^{(n)} \right) &= \sum_{k=1}^{K} \left(\mathcal{W}_0 \bigtimes_{n=1}^{N} \mathbf{Q}_k^{(n)} \mathbf{R}_k^{(n)} \right) && \text{(Using Lemma 1 and (21))} \\
&= \sum_{k=1}^{K} \left(\mathcal{W}_0 \bigtimes_{n=1}^{N} \mathbf{Q}_k^{(n)} \right) && \text{(Using the properties in (25))} \\
&= \sum_{k=1}^{K} \left(\left(\mathcal{X} - \hat{\mathcal{X}}_0 \right) \bigtimes_{n=1}^{N} \mathbf{Q}_k^{(n)} \right) && \text{(Using (12))}
\end{aligned} \tag{26}$$

holds. Thus, we see that the residual error $\mathcal{W}_1$ from (17) satisfies

$$\mathcal{X} = \hat{\mathcal{X}}_0 + \sum_{k=1}^{K} \left(\left(\mathcal{X} - \hat{\mathcal{X}}_0 \right) \bigtimes_{n=1}^{N} \mathbf{Q}_k^{(n)} \right) + \mathcal{W}_1. \tag{27}$$

Having derived (27) it behooves us to consider when using such a first scale approximation of $\mathcal{X}$ is actually better than, e.g., just using a standard HoSVD-based 0^{th} scale approximation of $\mathcal{X}$ along the lines of (12). As one might expect, this depends entirely on (i) how well the 1^{st} scale partitions (i.e., the restriction matrices utilized (20)) are chosen, as well as on (ii) how well restriction matrices of the type used in (20) interact with the projection matrices used to create the standard HoSVD-based approximation in question. Toward understanding these two conditions better, recall that $\hat{\mathcal{X}}_0 \in \mathbb{R}^{I_1 \times I_2 \times \ldots I_N}$ in (27) is defined as

$$\hat{\mathcal{X}}_0 = \mathcal{X} \bigtimes_{n=1}^{N} \mathbf{P}^{(n)} = \mathcal{X} \times_1 \mathbf{P}^{(1)} \times_2 \mathbf{P}^{(2)} \cdots \times_N \mathbf{P}^{(N)} \tag{28}$$

where the orthogonal projection matrices $\mathbf{P}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ are given by $\mathbf{P}^{(n)} = \hat{\mathbf{U}}^{(n)} \left(\hat{\mathbf{U}}^{(n)} \right)^\top$ for the matrices $\hat{\mathbf{U}}^{(n)} \in \mathbb{R}^{I_n \times r_n}$ used in (10). For simplicity let the ranks of the $\mathbf{P}^{(n)}$ projection matrices momentarily satisfy $r_1 = r_2 = \cdots = r_N =: r_0$ (i.e., let them all be rank $r_0 < \max_n \{\text{rank}(\mathbf{X}_{(n)})\}$). Similarly, let all the ranks, $r_k^{(n)}$, of the 1^{st} scale projection matrices $\mathbf{Q}_k^{(n)}$ in (23) be r_1 for the time being.

Motivated by, e.g., the memory cost analysis of Section III-A above, one can now ask when the multiscale approximation error, $\|\mathcal{W}_1\|$, resulting from (27) will be less than a standard HoSVD-based approximation error, $\|\mathcal{X} - \bar{\mathcal{X}}_0\|$, where

$$\bar{\mathcal{X}}_0 := \mathcal{X} \bigtimes_{n=1}^{N} \bar{\mathbf{P}}^{(n)} = \mathcal{X} \times_1 \bar{\mathbf{P}}^{(1)} \times_2 \bar{\mathbf{P}}^{(2)} \cdots \times_N \bar{\mathbf{P}}^{(N)}, \tag{29}$$

[2]Here we are implicitly using (1).

and each orthogonal projection matrix $\bar{\mathbf{P}}^{(n)}$ is of rank $\bar{r}_n = r_H \geqslant 2r_0 \geqslant r_0 + c^{N-1}r_1$ $\big($i.e., where each $\bar{\mathbf{P}}^{(n)}$ projects onto the top r_H left singular vectors of $\mathbf{X}_{(n)}\big)$. In this situation having both $\|\mathcal{W}_1\| < \|\mathcal{X} - \bar{\mathcal{X}}_0\|$ and $r_H \geqslant 2r_0 \geqslant r_0 + c^{N-1}r_1$ hold at the same time would imply that one could achieve smaller approximation error using MS-HoSVD than using HoSVD while simultaneously achieving better compression (recall Section III-A). In order to help facilitate such analysis we prove error bounds in Appendix A that are implied by the choice of a good partitioning scheme for the residual tensor $\mathcal{W}_0$ in (20) – (22).

In particular, with respect to the question concerning how well the 1^{st}-scale approximation error, $\|\mathcal{W}_1\|$, from (27) might compare to the HoSVD-based approximation error $\|\mathcal{X} - \bar{\mathcal{X}}_0\|$ we can use the following notion of an *effective partition of* $\mathcal{W}_0$. The partition of $\mathcal{W}_0$ formed by the restriction matrices $\mathbf{R}_k^{(n)}$ in (20) – (22) will be called *effective* if there exists another *pessimistic* partitioning of $\mathcal{W}_0$ via (potentially different) restriction matrices $\left\{\tilde{\mathbf{R}}_k^{(n)}\right\}_{k=1}^{K}$ together with a bijection $f : [K] \to [K]$ such that

$$\sum_{n=1}^{N} \left\| \mathcal{X}|_k \times_n \left(\mathbf{I} - \mathbf{Q}_k^{(n)}\right)\right\|^2 \leqslant \sum_{n=1}^{N} \left\| \mathcal{W}_0 \times_n \tilde{\mathbf{R}}_{f(k)}^{(n)} \left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right) \bigtimes_{h\neq n}^{N} \tilde{\mathbf{R}}_{f(k)}^{(h)} \right\|^2 \tag{30}$$

holds for each $k \in [K]$. In (30) the $\left\{\tilde{\mathbf{P}}^{(n)}\right\}$ are the orthogonal projection matrices obtained from the HoSVD of $\mathcal{W}_0$ with ranks $\tilde{r}_n = r_H$ $\big($i.e., where each $\tilde{\mathbf{P}}^{(n)}$ projects onto the top $\tilde{r}_n = r_H$ left singular vectors of the matricization $\mathbf{W}_{0,(n)}\big)$. In Appendix A, we show that (30) holding for $\mathcal{W}_0$ implies that the error $\|\mathcal{W}_1\|$ resulting from our 1^{st}-scale approximation in (27) is less than an upper bound of the type often used for HoSVD-based approximation errors of the form $\|\mathcal{X} - \bar{\mathcal{X}}_0\|$ (see, e.g., [28]). In particular, we prove the following result.

Theorem 1. *Suppose that* (30) *holds. Then, the first scale approximation error given by MS-HoSVD in* (27) *is bounded by*

$$\|\mathcal{W}_1\|^2 = \left\|\mathcal{X} - \hat{\mathcal{X}}_0 - \hat{\mathcal{X}}_1\right\|^2 \leqslant \sum_{n=1}^{N} \left\|\mathcal{X} \times_n \left(\mathbf{I} - \bar{\mathbf{P}}^{(n)}\right)\right\|^2,$$

where $\left\{\bar{\mathbf{P}}^{(n)}\right\}$ *are low-rank projection matrices of rank* $\bar{r}_n = \tilde{r}_n = r_H$ *obtained from the truncated HoSVD of* $\mathcal{X}$ *as per* (29)*.*

Proof. See Appendix A. □

Theorem 1 implies that $\|\mathcal{W}_1\|$ *may be less* than $\|\mathcal{X} - \bar{\mathcal{X}}_0\|$ when (30) holds. It does not, however, actually prove that $\|\mathcal{W}_1\| \leqslant \|\mathcal{X} - \bar{\mathcal{X}}_0\|$ holds whenever (30) does. In fact, directly proving that $\|\mathcal{W}_1\| \leqslant \|\mathcal{X} - \bar{\mathcal{X}}_0\|$ whenever (30) holds does not appear to be easy. It also does not appear to be easy to prove the error bound in theorem 1 without an assumption along the lines of (30) which simultaneously controls both (i) how well the restriction matrices utilized to partition $\mathcal{W}_0$ in (21) are chosen, as well as (ii) how poorly (worst case) restriction matrices interact with the projection matrices used to create standard HoSVD-based approximations of $\mathcal{W}_0$ and/or $\mathcal{X}$. The development of simpler and/or weaker conditions than (30) which still yield meaningful error guarantees along the lines of theorem 1 is left as future work. See Appendix A for additional details and comments, and Appendix B below for an example illustrating Theorem 1 on an idealized tensor..

D. Adaptive Pruning in Multiscale HoSVD for Improved Performance

In order to better capture the local structure of the tensor, it is important to look at higher scale decompositions. However, as the scale increases, the storage cost and computational complexity will increase making any gain in reconstruction error potentially not worth the additional memory cost. For this reason, it is important to carefully select the subtensors adaptively at higher scales. To help avoid the redundancy in decomposition structure we propose an adaptive pruning method across scales.

In adaptive pruning, the tree is pruned by minimizing the following cost function $\mathbb{H} = Error + \lambda \cdot Compression$ similar to the rate-distortion criterion commonly used by compression algorithms where λ is the trade-off parameter [33]. To minimize this function we employ a greedy procedure similar to sequential forward selection [34]. First, the root node which stores $\hat{\mathcal{X}}_0$ is created and scale-1 subtensors $\hat{\mathcal{X}}_{1,k}$ are obtained from the 0th order residual tensor $\hat{\mathcal{W}}_0$ as discussed in Section III. These subtensors are stored in a list and the subtensor which decreases the cost function the most is then added to the tree structure under its parent node. Next, scale-2 subtensors belonging to the added node are created and added to the list. All of the scale-1 and scale-2 subtensors in the list are again evaluated to find the subtensor that minimizes the cost function. This procedure is repeated until the cost function $\mathbb{H}$ converges or the decrease is minimal. A pseudocode of the algorithm is given in Algorithm 2. It is important to note that this algorithm is suboptimal similar to other greedy search methods.

IV. Data Reduction

In this section we demonstrate the performance of MS-HoSVD for tensor type data reduction on several real 3-mode and 4-mode datasets as compared with three other tensor decompositions: HoSVD, H-Tucker, and T-Train. The performance of tensor decomposition methods are evaluated in terms of reconstruction error and compression rate. In the tables and figures

Algorithm 2 Multiscale HoSVD with Adaptive Pruning

Input: $\mathcal{X}$: tensor , $\mathbf{C} = (c_1,\ c_2,\ ...,\ c_N)$: the desired number of clusters for each modes, s_H: the highest scale of MS-HoSVD.
Output: T: Tree structure containing the MS-HoSVD decomposition of $\hat{\mathcal{X}}$.
Create an empty tree T.
Create an empty list L.
Add node containing $\mathcal{X}$ to L.
while There is a node in L that decreases the cost function $\mathbb{H}(T)$. **do**
    Find the node corresponding to $\mathcal{X}_{s,t}$ (the tth subtensor from sth scale) in the list L that decreases $\mathbb{H}$ the most where $s \in \{0, ..., s_H\}$ and $t \in \{1, ..., K^s\}$.
    $\mathcal{C}_{s,t}, \left\{\hat{\mathbf{U}}_{s,t}^{(n)}\right\} \leftarrow$ truncatedHOSVD($\mathcal{X}_{s,t}$).
    Add the node containing $\mathcal{C}_{s,t}, \left\{\hat{\mathbf{U}}_{s,t}^{(n)}\right\}$ to T.
    if $s < s_H$ **then**
        Compute $\mathcal{W}_{s,t} = \mathcal{X}_{s,t} - \hat{\mathcal{X}}_{s,t}$.
        Create K subtensors $\mathcal{X}_{s+1,K(t-1)+k}$ with $J^n_{s+1,K(t-1)+k}$ from $\mathcal{W}_{s,t}$ where $k \in \{1,\ 2,\ ...,\ K\}$ and $n \in \{1,\ 2,\ ...,\ N\}$.
        Add K nodes containing $\mathcal{X}_{s+1,K(t-1)+k}$ and $\left\{J^n_{s+1,K(t-1)+k}\right\}$ to L.
    end if
end while

below the error rate refers to the normalized tensor approximation error $\frac{\|\mathcal{X}-\hat{\mathcal{X}}\|_F}{\|\mathcal{X}\|_F}$ and the compression rate is computed as $\frac{\#\text{ total bits to store }\hat{\mathcal{X}}}{\#\text{ total bits to store }\mathcal{X}}$. Moreover, we show the performance of the proposed adaptive tree prunning strategy for data reduction.

A. Datasets

1) PIE dataset: A 3-mode tensor $\mathcal{X} \in \mathbb{R}^{244\times320\times138}$ is created from PIE dataset [35]. The tensor contains 138 images from 6 different yaw angles and varying illumination conditions collected from a subject where each image is converted to gray scale. Fig. 2 illustrates the images from different frames of the PIE dataset.

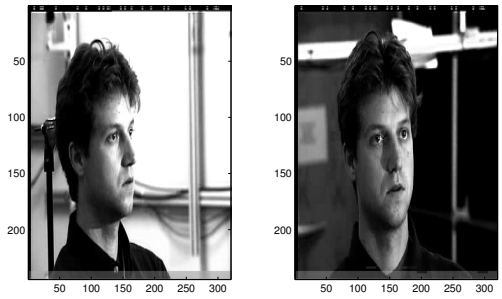


Fig. 2: Sample frames from PIE dataset corresponding to the 30th (left) and 80th (right) frames.

2) COIL-100 dataset: The COIL-100 database contains 7200 images collected from 100 objects where the images of each object were taken at pose intervals of 5°. A 4-mode tensor $\mathcal{X} \in \mathbb{R}^{128\times128\times72\times100}$ is created from COIL-100 dataset [36]. The constructed 4-mode tensor contains 72 images of size 128×128 from 100 objects where each image is converted to gray scale. In Fig. 3, sample images of four objects taken from different angles can be seen.

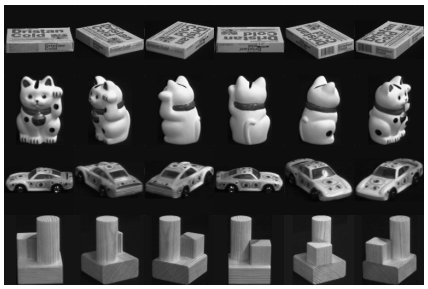


Fig. 3: Image samples of four different objects from COIL-100 dataset from varying pose angles (from 0° to 240° with 60° increments).

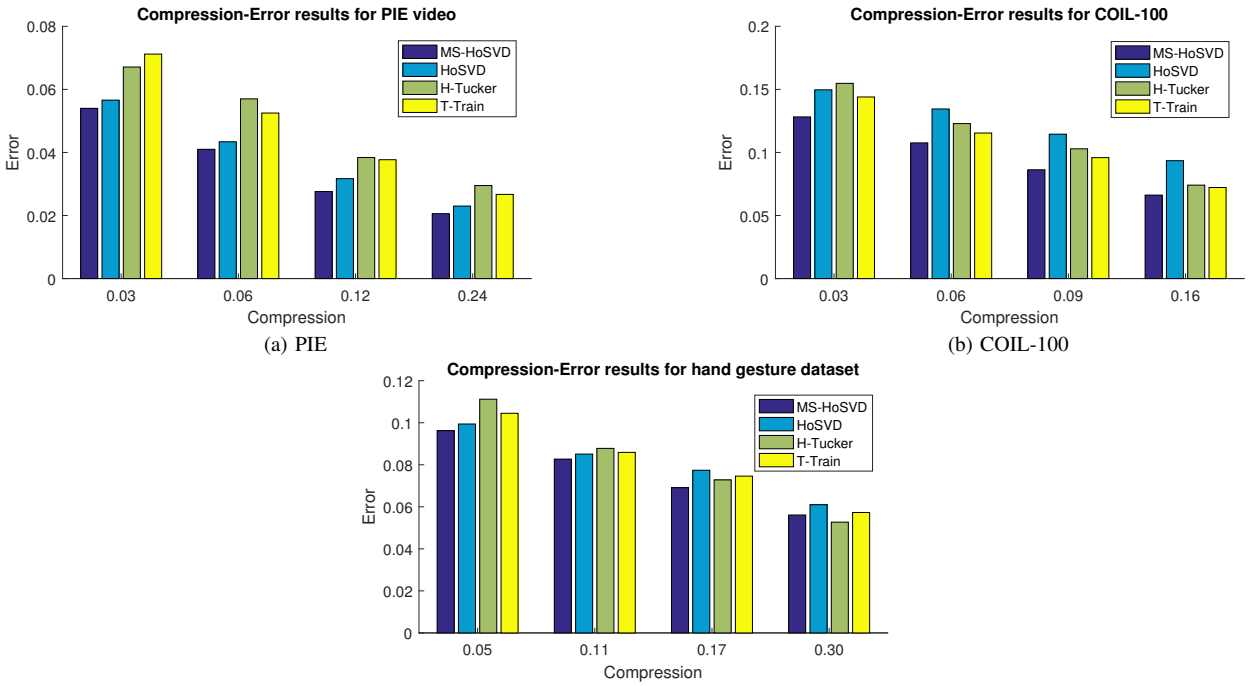
3) The Cambridge Hand Gesture Dataset: The Cambridge hand gesture database consists of 900 image sequences of nine gesture classes of three primitive hand shapes and three primitive motions where each class contains 100 image sequences (5 different illuminations $\times$ 10 arbitrary motions $\times$ 2 subjects). In Fig. 4, sample image sequences collected for nine hand gestures can be seen. The created 4-mode tensor $\mathcal{X} \in \mathbb{R}^{60\times80\times30\times900}$ contains 900 image sequences of size $60 \times 80 \times 30$ where each image is converted to gray scale.

Fig. 4: Illustration of nine different classes in Cambridge Hand Gesture Dataset.

B. Data Reduction Experiments

In this section, we evaluate the performance of MS-HoSVD for 1 and 2-scale decompositions compared to HoSVD, H-Tucker and T-Train decompositions. In the following experiments, tensor partitioning is performed by LSA and the cluster number along each mode is chosen as $c_i = 2$. The rank used in HoSVD is selected adaptively using the energy criterion as per Section III's (11). In our experiments, we performed MS-HoSVD with $\tau = 0.7$ and $\tau = 0.75$ and we kept τ the same for each scale. For the same compression rates as the MS-HoSVD, the reconstruction error of HoSVD, H-Tucker and T-Train models are computed.

Fig. 5 explores the interplay between compression rate and approximation error for MS-HoSVD in comparison to HoSVD, H-Tucker and T-Train for PIE, COIL-100 and hand gesture datasets. Starting from the left in Figs. 5(a), 5(b) and 5(c), the first two compression rates correspond to 1-scale MS-HoSVD with $\tau = 0.7$ and $\tau = 0.75$, respectively while the last two are obtained from 2-scale approximation with $\tau = 0.7$ and $\tau = 0.75$, respectively. As seen in Fig. 5, MS-HoSVD outperforms other approaches with respect to reducing PIE, COIL-100 and hand gesture tensors at varying compression rates. Moreover, adding 2nd scale increases the storage requirements while decreasing the error of MS-HoSVD. Fig. 6 illustrates the influence of scale on the visual quality of the reconstructed images. As expected, introducing additional finer scales into a multiscale approximation of video data improves image detail in each frame. Moreover, the data reduction performance of T-Train is seen to be slightly better than H-Tucker in most of the experiments.



(a) PIE

(b) COIL-100

(c) Hand Gesture

Fig. 5: Compression rate versus Normalized Reconstruction Error for MS-HoSVD (dark blue), HoSVD (light blue), H-Tucker (green) and T-Train (yellow) for a) PIE, b) COIL-100 and c) Hand Gesture datasets. Starting from the left for all (a), (b) and (c), the first two compression rates correspond to 1-scale MS-HoSVD with $\tau = 0.7$ and $\tau = 0.75$ while the last two are obtained from 2-scale approximation with $\tau = 0.7$ and $\tau = 0.75$, respectively. MS-HoSVD provides lower error than HoSVD, H-Tucker and T-Train.

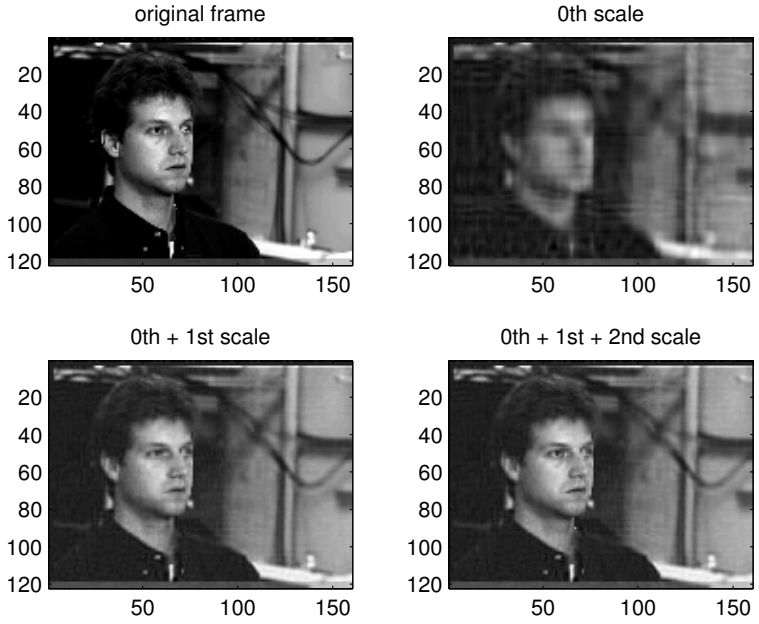


Fig. 6: A single frame of the PIE dataset showing increasing accuracy with scale for MS-HoSVD.

C. Data Reduction with Adaptive Tree Prunning

In this section, we evaluate the performance of adaptive tree pruning multiscale decompositions. In the pruning experiments, clustering is performed by LSA and the cluster number along each mode is chosen as $c_i = 2$. The rank used in HoSVD is

selected adaptively based on the energy threshold $\tau = 0.7$. A pruned version of 2-scale MS-HoSVD that greedily minimizes the cost function $\mathbb{H} = Error + \lambda \cdot Compression$ for is implemented for PIE, COIL-100 and Hand Gesture datasets with varying λ values as reported in Tables I, II and III. As λ increases, reducing the compression rate becomes more important and the algorithm prunes the leaf nodes more. For example, a choice of $\lambda = 0.75$ prunes all of the nodes corresponding to the second scale subtensors for PIE data (see Table I).

As can be seen from Tables I, II, and III, the best tradeoffs achieved between reconstruction error and compression rate occur at different λ values for different datasets. For example, for PIE data, increasing λ value does not provide much change in reconstruction error while increasing the compression. On the other hand, for COIL-100, $\lambda = 0.75$ provides a good tradeoff between reconstruction error and compression rate. Small changes in λ yield significant effects on pruning the subtensors of 2-scale decomposition of hand gesture data. Fig. 7 illustrates the performance of the pruning algorithm on the PIE dataset. Applying pruning with $\lambda = 0.25$ increases the reconstruction error from 0.0276 to 0.0506 while reducing the compression rate by a factor of 4 (Table I). As seen in Fig. 7, the 2nd scale approximation obtained by the adaptive pruning algorithm preserves most of the facial details in the image.

TABLE I: Reconstruction error and compression rate computed for pruned tree structure obtained by applying MS-HoSVD with 2-scales to PIE data.

λ	0	0.22	0.25	0.30	0.75
Normalized error	0.0276	0.0395	0.0506	0.0530	0.0540
Compression	0.1241	0.0809	0.0377	0.0284	0.0261
Scales of subtensors	0+1+2	0+1+2	0+1+2	0+1+2	0+1

TABLE II: Reconstruction error and compression rate computed for pruned tree structure obtained by applying MS-HoSVD with 2-scales to COIL-100 dataset.

λ	0	0.25	0.50	0.75	0.80
Normalized error	0.0857	0.0867	0.0913	0.1060	0.1207
Compression	0.0863	0.0840	0.0734	0.0526	0.0347
Scales of subtensors	0+1+2	0+1+2	0+1+2	0+1+2	0+1+2

TABLE III: Reconstruction error and compression rate computed for pruned tree structure obtained by applying MS-HoSVD with 2-scales to Hand Gesture dataset.

λ	0	0.25	0.26	0.27	0.28
Normalized error	0.0691	0.0869	0.0913	0.0946	0.0999
Compression	0.1694	0.1056	0.0827	0.0698	0.0514
Scales of subtensors	0+1+2	0+1+2	0+1+2	0+1+2	0+1

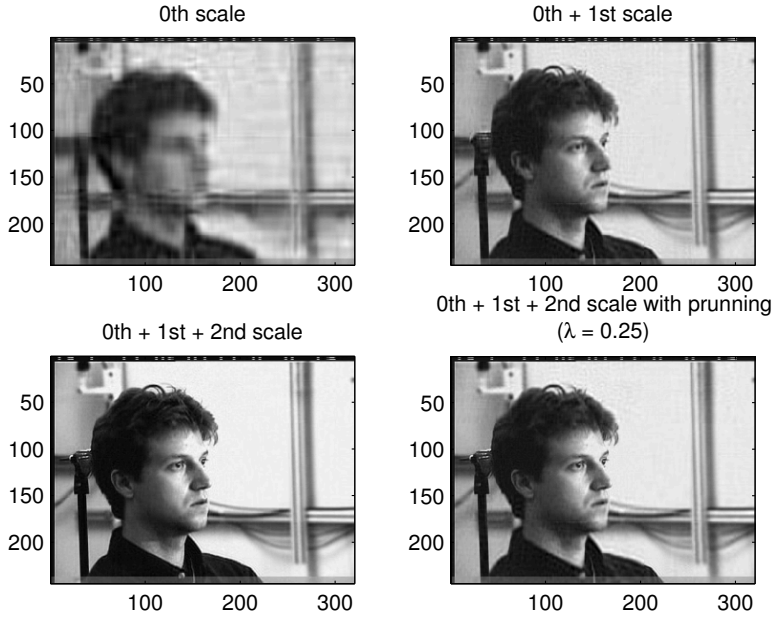
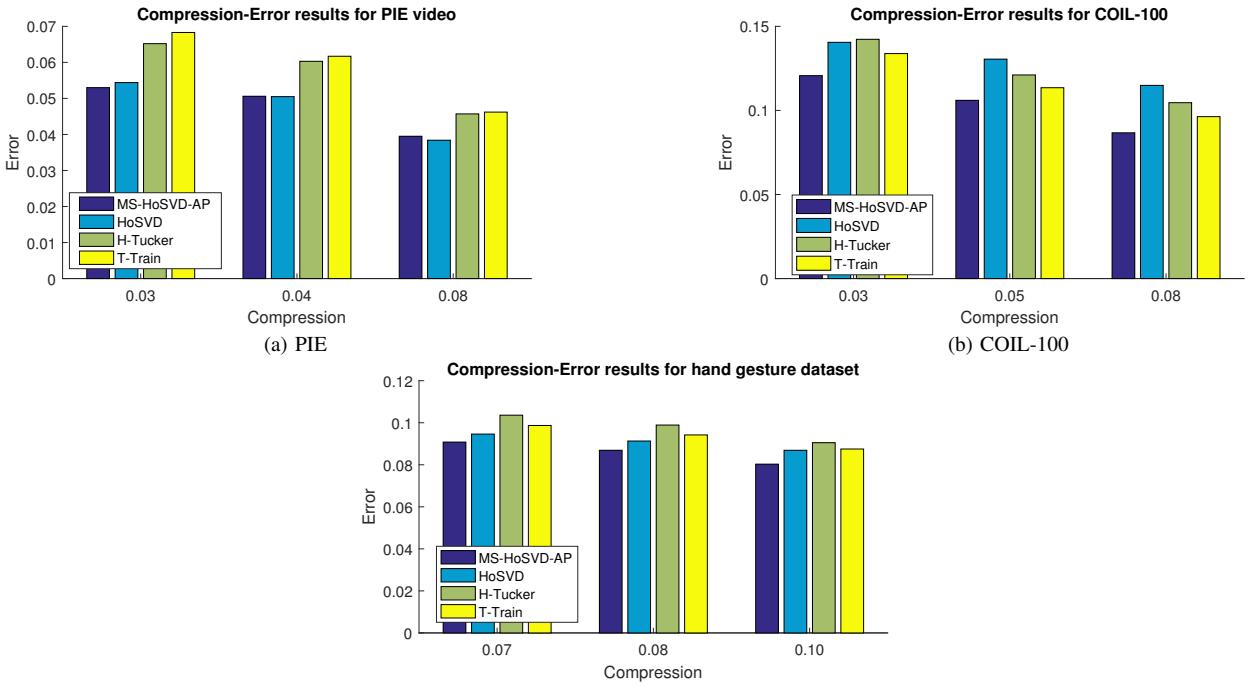


Fig. 7: Reconstruction error and compression rate computed for the pruned tree structure obtained by applying MS-HoSVD with 2-scales to the PIE dataset. The top-left and right image is the sample frame obtained by reconstructing the tensor using only the 0th scale, and 0th and 1st scales, respectively. The bottom-left image is a sample frame reconstructed using the 2-scale approximation with all the sub-tensors, and the bottom-right image is the reconstruction using the 2 scale analysis with the pruning approach where $\lambda = 0.25$.

Performance of the pruning algorithm reported in Tables I, II and III is also compared with HoSVD, H-Tucker and T-Train decompositions in Fig. 8. As seen in Fig. 8 (b) and (c), MS-HoSVD outperforms other approaches for compressing COIL-100 and Hand Gesture datasets at varying compression rates. However, for PIE data, the performance of MS-HoSVD and HoSVD are very close to each other while both approaches outperform H-Tucker and T-Train, as can be seen in Fig. 8 (a). In Fig. 9, sample frames of PIE data reconstructed by T-Train (top-left), H-Tucker (top-right), HoSVD (bottom-left) and pruned MS-HoSVD with 2-scales (bottom-right) are shown. It can be easily seen that the reconstructed images by H-Tucker and T-Train are more blurred than the ones obtained by HoSVD and MS-HoSVD. One can also see the facial details captured by MS-HoSVD are clearer than HoSVD although the performances of both algorithms are very similar to each other. The reason for capturing facial details better by MS-HoSVD is that the higher scale subtensors encode facial details.



(a) PIE (b) COIL-100

(c) Hand Gesture

Fig. 8: Compression rate versus Normalized Reconstruction Error for MS-HoSVD with adaptive pruning (dark blue), HoSVD (light blue), H-Tucker (green) and T-Train (yellow) for a) PIE, b) COIL-100 and c) Hand Gesture datasets. 2-scale MS-HoSVD tensor approximations are obtained using $\tau = 0.7$ for each scale and varying pruning trade-off parameter λ.

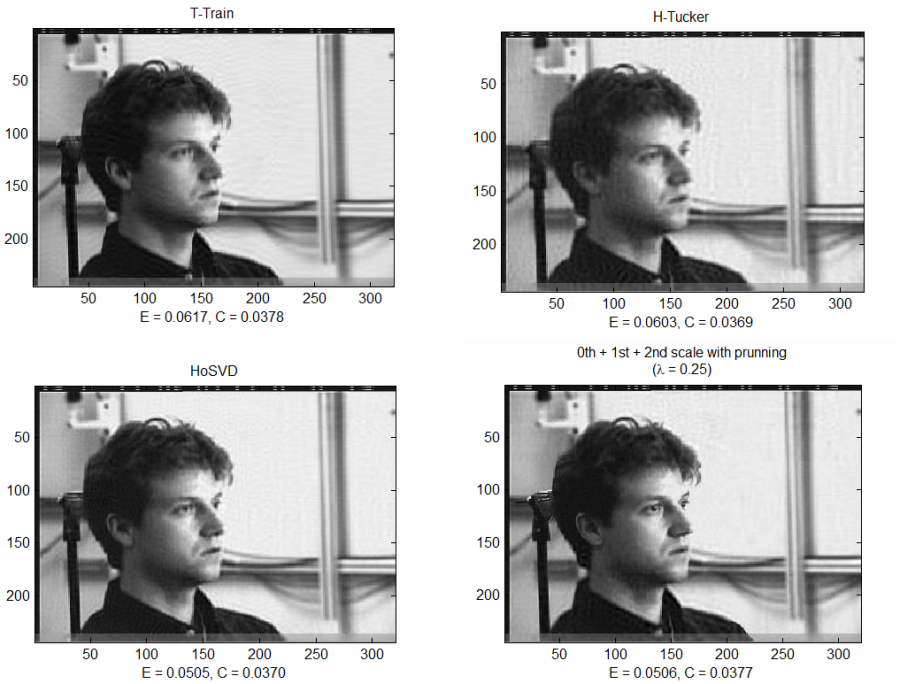


Fig. 9: Reconstructed frame samples from PIE data compressed by T-Train (top-left), H-Tucker (top-right), HoSVD (bottom-left) and pruned MS-HoSVD with 2-scales (bottom-right). 2-scale MS-HoSVD tensor approximation is obtained using $\tau = 0.7$ for each scale and $\lambda = 0.25$.

V. Feature Extraction and Classification

In this section, we evaluate the features extracted from MS-HoSVD for classification of 2-mode and 3-mode tensors containing object images and hand gesture videos.

The classification accuracy of MS-HosVD features are compared to the features extracted by HoSVD and T-Train using three different classifiers: 1-NN, Adaboost and Naive Bayes.

A. COIL-100 Image Dataset

For computational efficiency, each image was downsampled to a gray-scale image of 32×32 pixels. Number of images per object used for training data was gradually increased from 18 to 54 and selected randomly. A 3-mode tensor $\mathcal{X}^{tr} \in \mathbb{R}^{32\times32\times I_{tr}}$ is constructed from training images where $I_{tr} \in 100 \times \{18,\ 36,\ 54\}$ and the rest of the images are used to create the testing tensor $\mathcal{X}^{te} \in \mathbb{R}^{32\times32\times I_{te}}$ where $I_{te} = 7200 - I_{tr}$.

B. The Cambridge Hand Gesture Dataset

For computational efficiency, each image was downsampled to a gray-scale image of 30×40 pixels. Number of image sequences used for training data gradually increased from 25 to 75 per gesture and selected randomly. A 4-mode tensor $\mathcal{X}^{tr} \in \mathbb{R}^{30\times40\times30\times I_{tr}}$ is constructed from training image sequences where $I_{tr} \in 9 \times \{25,\ 50,\ 75\}$ and the rest of the image sequences are used to create the testing tensor $\mathcal{X}^{te} \in \mathbb{R}^{30\times40\times15\times I_{te}}$ where $I_{te} = 900 - I_{tr}$.

C. Classification Experiments

1) Training: For MS-HoSVD, the training tensor $\mathcal{X}^{tr}$ is decomposed using 1-scale MS-HoSVD as follows. Tensor partitioning is performed by LSA and the cluster number along each mode is chosen as $c = \{2,\ 3,\ 1\}$ yielding 6 subtensors for COIL-100 dataset and $c = \{2,\ 2,\ 3,\ 1\}$ yielding 12 subtensors for hand gesture dataset. We did not partition the tensor along the last mode that corresponds to the classes to make the comparison with other methods fair. The rank used in 0th scale is selected based on the energy criterion with $\tau = 0.7$, while the full rank decomposition is used for the 1st scale. The 0th scale approximation:

$$\hat{\mathcal{X}}_0^{tr} = \mathcal{C}_0^{tr} \times_1 \hat{\mathbf{U}}^{tr,(1)} \times_2 \hat{\mathbf{U}}^{tr,(2)} ... \times_N \hat{\mathbf{U}}^{tr,(N)} \tag{31}$$

provides 0th scale core tensor $\mathcal{C}_0^{tr}$, factor matrices $\hat{\mathbf{U}}^{tr,(i)}$ and residual tensor $\mathcal{W}_0^{tr} = \mathcal{X}^{tr} - \hat{\mathcal{X}}_0^{tr}$. Next, 0th scale feature tensor $\mathcal{S}_0^{tr}$ for the training data is created by projecting $\mathcal{X}^{tr}$s onto the first $N-1$ factor matrices $\mathbf{U}^{tr,(i)}$ as:

$$\mathcal{S}_0^{tr} = \mathcal{X}^{tr} \times_1 \left(\hat{\mathbf{U}}^{tr,(1)}\right)^\top \times_2 \left(\hat{\mathbf{U}}^{tr,(2)}\right)^\top ... \times_{N-1} \left(\hat{\mathbf{U}}^{tr,(N-1)}\right)^\top. \tag{32}$$

Subtensors of $\mathcal{W}_0^{tr}$ obtained by $\mathcal{X}_{1,k}^{tr} = \mathcal{W}_0^{tr}(J_{1,k}^{tr,1} \times J_{1,k}^{tr,2} \times ... \times J_{1,k}^{tr,N})$ are used to extract 1st order core tensors $\mathcal{C}_{1,k}$ and factor matrices $\mathbf{U}_{1,k}^{tr,(i)}$ as:

$$\mathcal{X}_{1,k}^{tr} = \mathcal{C}_{1,k}^{tr} \times_1 \mathbf{U}_{1,k}^{tr,(1)} \times_2 \mathbf{U}_{1,k}^{tr(2)} ... \times_N \mathbf{U}_{1,k}^{tr(N)}. \tag{33}$$

1st order feature tensors are then created by projecting $\mathcal{X}_{1,k}^{tr}$s onto the first $N-1$ factor matrices $\mathbf{U}_{1,k}^{tr,(i)}$ as:

$$\mathcal{S}_{1,k}^{tr} = \mathcal{X}_{1,k}^{tr} \times_1 \left(\mathbf{U}_{1,k}^{tr,(1)}\right)^\top \times_2 \left(\mathbf{U}_{1,k}^{tr,(2)}\right)^\top ... \times_{N-1} \left(\mathbf{U}_{1,k}^{tr,(N-1)}\right)^\top. \tag{34}$$

Unfolding the feature tensors $\mathcal{S}_0^{tr}$ and $\mathcal{S}_{1,k}^{tr}$ along the sample mode N and concatenating them to each other yields a high dimensional feature vector for each of the training samples. From these vectors, N_f features with the highest Fisher Score [37] are selected to form the lower-dimensional feature vectors $\mathbf{x}^{tr} \in \mathbb{R}^{N_f \times 1}$ for each training sample where the number of features (N_f) is determined 100 for COIL-100 and 200 for hand gesture dataset emprically. For HoSVD and T-Train, full rank decompositions are computed and feature vectors are created by selecting N_f features with the highest Fisher Score from the core tensors as described above. For T-Train, the procedure described in [38] is used without reducing the dimensionality.

2) Testing: To create 0th order feature tensor $\mathcal{S}_0^{te}$ for testing samples, first, the testing tensor $\mathcal{X}_{te}$ is projected onto $\hat{\mathbf{U}}^{tr,(i)}$ where $i \in [N-1]$ as:

$$\mathcal{S}_0^{te} = \mathcal{X}^{te} \times_1 \left(\hat{\mathbf{U}}^{tr,(1)}\right)^\top \times_2 \left(\hat{\mathbf{U}}^{tr,(2)}\right)^\top ... \times_{N-1} \left(\hat{\mathbf{U}}^{tr,(N-1)}\right)^\top. \tag{35}$$

0th order residual tensor $\mathcal{W}_0^{te}$ of testing data is computed as $\mathcal{W}_0^{te} = \mathcal{X}^{te} - \mathcal{X}^{te} \times_{n=1}^{n=N} \left(\hat{\mathbf{U}}^{tr,(n)}\right)^\top$. Then 1st order subtensors are created from $\mathcal{W}_0^{te}$ using the same partitioning as the 0th order training residual tensor $\mathcal{W}_0^{tr}$ as $\mathcal{X}_{1,k}^{te} = \mathcal{W}_0^{te}(J_{1,k}^{tr,1} \times J_{1,k}^{tr,2} \times ... \times J_{1,k}^{tr,N})$. 1st order feature tensors $\mathcal{S}_{1,k}^{te}$ for the testing samples are then obtained by

$$\mathcal{S}_{1,k}^{te} = \mathcal{X}_{1,k}^{te} \times_1 \left(\mathbf{U}_{1,k}^{tr,(1)}\right)^\top \times_2 \left(\mathbf{U}_{1,k}^{tr,(2)}\right)^\top ... \times_{N-1} \left(\mathbf{U}_{1,k}^{tr,(N-1)}\right)^\top. \tag{36}$$

Similar to the training step, unfolding the feature tensors $\mathcal{S}_0^{te}$ and $\mathcal{S}_{1,k}^{te}$ along the sample mode N and concatenating them with each other yields high dimensional feature vectors for the testing samples. The features corresponding to the features selected from the training step are used to form the feature vectors for testing samples $\mathbf{x}^{te} \in \mathbb{R}^{N_f \times 1}$. A similar two-step

procedure, i.e projecting the testing tensor onto training factor matrices followed by selecting N_f features, is used to create testing feature vectors for HoSVD and T-Train. Discrimination performance of the feature vectors are evaluated using different classifiers including 1-NN, Adaboost and Naive Bayes.

Tables IV and V summarize the classification accuracy for the three methods using three different classifiers for COIL-100 and Hand gesture data sets, respectively. As it can be seen from these Tables, for both data sets and all classifiers MS-HoSVD performs the best except for a Naive Bayes Classifier trained by 25% of the data to classify hand gesture dataset. As seen in Tables IV and V, the performance of HoSVD, T-Train and MS-HoSVD become close to each other as the size of the training dataset increases, as expected. The reason for the superior performance of MS-HoSVD is that MS-HoSVD captures the variations and nonlinearities across the modes such as rotation or translation better than the other methods. In both of the datasets used in this section, the images are rotated across the different frames. Since these nonlinearities are encoded in the higher scale (1st scale) features while the average characteristics, which are the same as HoSVD, are captured by the lower scale (0th scale) MS-HoSVD features, the classification performance of the MS-HoSVD is slightly better than HoSVD. It is also seen that T-Train features are not as good as MS-HoSVD and HoSVD features for capturing rotations and translations in the data and requires larger training set to reach the performance of MS-HoSVD and HoSVD.

TABLE IV: Classification results for COIL-100 dataset over 20 trials with $N_f = 100$.

Training Size	Method	1-NN mean $\pm$ std.	Adaboost mean $\pm$ std.	Bayes mean $\pm$ std.
25%	MS-HoSVD	**93.71 $\pm$ 1.28**	**88.90 $\pm$ 1.24**	**89.88 $\pm$ 2.08**
	HoSVD	93.07 $\pm$ 1.33	87.47 $\pm$ 1.53	87.36 $\pm$ 3.32
	T-Train	92.29 $\pm$ 2.23	87.26 $\pm$ 1.84	88.43 $\pm$ 1.36
50%	MS-HoSVD	**97.41 $\pm$ 0.69**	**92.54 $\pm$ 1.58**	**91.62 $\pm$ 1.21**
	HoSVD	97.10 $\pm$ 0.83	91.29 $\pm$ 2.16	90.43 $\pm$ 1.60
	T-Train	96.99 $\pm$ 1.09	91.36 $\pm$ 1.44	91.21 $\pm$ 1.66
75%	MS-HoSVD	**98.38 $\pm$ 0.65**	**93.66 $\pm$1.35**	**92.60 $\pm$ 1.79**
	HoSVD	98.16 $\pm$ 0.72	92.23 $\pm$1.46	92.38 $\pm$ 1.92
	T-Train	98.25 $\pm$ 0.41	93.10 $\pm$1.33	92.33 $\pm$ 1.52

TABLE V: Classification results for hand gesture dataset over 20 trials with $N_f = 200$.

Training Size	Method	1-NN mean $\pm$ std.	Adaboost mean $\pm$ std.	Bayes mean $\pm$ std.
25%	MS-HoSVD	**75.40 $\pm$ 3.87**	**77.81 $\pm$ 1.87**	82.89 $\pm$ 1.84
	HoSVD	75.01 $\pm$ 3.99	77.37 $\pm$ 2.51	**83.11 $\pm$ 2.89**
	T-Train	69.20 $\pm$ 2.63	67.53 $\pm$ 2.27	72.99 $\pm$ 4.91
50%	MS-HoSVD	**83.86 $\pm$ 3.12**	**85.46 $\pm$ 1.66**	**85.86 $\pm$ 2.25**
	HoSVD	83.15 $\pm$ 2.90	85.14 $\pm$ 1.45	84.00 $\pm$ 1.68
	T-Train	78.97 $\pm$ 2.25	78.82 $\pm$ 2.87	81.46 $\pm$ 1.37
75%	MS-HoSVD	**87.47 $\pm$ 2.07**	**88.15 $\pm$ 2.20**	**86.75 $\pm$ 2.63**
	HoSVD	86.93 $\pm$ 2.31	88.08 $\pm$ 2.51	85.04 $\pm$ 2.41
	T-Train	85.64 $\pm$ 2.57	82.60 $\pm$ 3.16	83.64 $\pm$ 2.31

VI. Conclusions

In this paper, we proposed a new multi-scale tensor decomposition technique for better approximating the local nonlinearities in generic tensor data. The proposed approach constructs a tree structure by considering similarities along different fibers of the tensor and decomposes the tensor into lower dimensional subtensors hierarchically. A low-rank approximation of each subtensor is then obtained by HoSVD. We also introduced a pruning strategy to find the optimum tree structure by keeping the important nodes and eliminating redundancy in the data. The proposed approach is applied to a set of 3-way and 4-way tensors to evaluate its performance on both data reduction and classification applications. As it is illustrated in the Results section, any application involving tensor data reduction and classification would benefit from the proposed method. Some examples include hyper-spectral image compression, high-dimensional video clustering and functional connectivity network analysis in neuroscience.

Although this paper focused on the integration of a single existing tensor factorization technique (i.e., the HoSVD) into a clustering-enhanced multiscale approximation framework, we would like to emphasize that the ideas presented herein are significantly more general. In principal, for example, there is nothing impeding the development of multiscale variants of other tensor factorization approaches (e.g., PARAFAC, T-Train, H-Tucker, etc.) in essentially the same way. In this paper it is demonstrated that the use of the HoSVD as part of a multiscale approximation approach leads to improved compression and classification performance over standard HoSVD approaches. However, this paper should additionally be considered as evidence that similar improvements are also likely possible for other tensor factorization-based compression and classification schemes, as well as for other related applications.

Future work will consider automatic selection of parameters such as the number of clusters and the appropriate rank along each mode. The computational efficiency of the proposed method can also be improved through parallelization of the

algorithm by, e.g., constructing the disjoint subtensors at each scale in parallel, as well as by utilizing distributed and parallel SVD algorithms such as [32] when computing their required HoSVD decompositions (see also, e.g., [39] for other related parallel implementations). Such efficient implementations will enable the computation of finer scale decompositions for higher order and higher dimensional tensors.

REFERENCES

[1] Letexier *et al.*, "Nonorthogonal tensor matricization for hyperspectral image filtering," *Geoscience and Remote Sensing Letters, IEEE*, vol. 5, no. 1, pp. 3–7, 2008.

[2] Kim and Cipolla, "Canonical correlation analysis of video volume tensors for action categorization and detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 31, no. 8, pp. 1415–1428, 2009.

[3] Miwakeichi *et al.*, "Decomposing eeg data into space–time–frequency components using parallel factor analysis," *NeuroImage*, vol. 22, no. 3, pp. 1035–1045, 2004.

[4] Cichocki *et al.*, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, 2015.

[5] Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[6] De Lathauwer, "Decompositions of a higher-order tensor in block terms-part ii: definitions and uniqueness," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1033–1066, 2008.

[7] Merhi *et al.*, "Face Recognition Using M-Band Wavelet Analysis," in *Proc. World Academy of Science, Engineering and Technology*, vol. 68, 2012.

[8] Cheng *et al.*, "Probabilistic tensor canonical polyadic decomposition with orthogonal factors," *IEEE Transactions on Signal Processing*, vol. 65, no. 3, pp. 663–676, 2015.

[9] Fu *et al.*, "Joint tensor factorization and outlying slab suppression with applications," *IEEE Transactions on Signal Processing*, vol. 63, no. 23, pp. 6315–6328, 2015.

[10] Kolda and Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[11] De Lathauwer *et al.*, "A multilinear singular value decomposition," *SIAM journal on Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.

[12] Shashua and Hazan, "Non-negative tensor factorization with applications to statistics and computer vision," in *Proceedings of the 22nd international conference on Machine learning*. ACM, 2005, pp. 792–799.

[13] Cichocki *et al.*, "Non-negative tensor factorization using alpha and beta divergences," in *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 3. IEEE, 2007, pp. III–1393.

[14] Cichocki *et al.*, "Novel multi-layer non-negative tensor factorization with sparsity constraints," in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2007, pp. 271–280.

[15] Grasedyck, "Hierarchical singular value decomposition of tensors," *SIAM Journal on Matrix Analysis and Applications*, vol. 31, no. 4, pp. 2029–2054, 2010.

[16] Suter *et al.*, "Tamresh–tensor approximation multiresolution hierarchy for interactive volume visualization," in *Computer Graphics Forum*, vol. 32, no. 3pt2. Wiley Online Library, 2013, pp. 151–160.

[17] Vasilescu and Terzopoulos, "Multilinear image analysis for facial recognition," in *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, vol. 2. IEEE, 2002, pp. 511–514.

[18] Yang *et al.*, "Two-dimensional pca: a new approach to appearance-based face representation and recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 26, no. 1, pp. 131–137, 2004.

[19] He *et al.*, "Tensor subspace analysis," in *Advances in neural information processing systems*, 2005, pp. 499–506.

[20] Niyogi, "Locality preserving projections," in *Neural information processing systems*, vol. 16. MIT, 2004, p. 153.

[21] Dai and Yeung, "Tensor embedding methods," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, no. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006, p. 330.

[22] Chen *et al.*, "Local discriminant embedding and its variants," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, vol. 2. IEEE, 2005, pp. 846–853.

[23] He *et al.*, "Neighborhood preserving embedding," in *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, vol. 2. IEEE, 2005, pp. 1208–1213.

[24] Li *et al.*, "Discriminant locally linear embedding with high-order tensor data," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 38, no. 2, pp. 342–352, 2008.

[25] Özdemir *et al.*, "Multiscale tensor decomposition," in *Signals, Systems and Computers, 2016 50th Asilomar Conference on*. IEEE, 2016, pp. 625–629.

[26] Ozdemir *et al.*, "Multi-scale higher order singular value decomposition (ms-hosvd) for resting-state fmri compression and analysis," in *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017, pp. 6299–6303.

[27] De Silva and Lim, "Tensor rank and the ill-posedness of the best low-rank approximation problem," *SIAM Journal on Matrix Analysis and Applications*, vol. 30, no. 3, pp. 1084–1127, 2008.

[28] Vannieuwenhoven *et al.*, "A new truncation strategy for the higher-order singular value decomposition," *SIAM Journal on Scientific Computing*, vol. 34, no. 2, pp. A1027–A1052, 2012.

[29] Ozdemir *et al.*, "Locally linear low-rank tensor approximation," in *Signal and Information Processing (GlobalSIP), 2015 IEEE Global Conference on*. IEEE, 2015, pp. 839–843.

[30] Yan and Pollefeys, "A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate," in *Computer Vision–ECCV 2006*. Springer, 2006, pp. 94–106.

[31] Karami *et al.*, "Compression of hyperspectral images using discerete wavelet transform and tucker decomposition," *IEEE journal of selected topics in applied earth observations and remote sensing*, vol. 5, no. 2, pp. 444–450, 2012.

[32] Iwen and Ong, "A distributed and incremental svd algorithm for agglomerative data analysis on large networks," *SIAM Journal on Matrix Analysis and Applications*, vol. 37, no. 4, pp. 1699–1718, 2016.

[33] Ramchandran and Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 160–175, 1993.

[34] Pudil *et al.*, "Floating search methods in feature selection," *Pattern recognition letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[35] Sim *et al.*, "The cmu pose, illumination, and expression database," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 25, no. 12, pp. 1615–1618, 2003.

[36] Nene *et al.*, "Columbia object image library (coil-20)," 1996.

[37] Gu *et al.*, "Generalized fisher score for feature selection," *arXiv preprint arXiv:1202.3725*, 2012.

[38] Bengua *et al.*, "Matrix product state for higher-order tensor compression and classification," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 4019–4030, 2016.

[39] Liavas and Sidiropoulos, "Parallel algorithms for constrained tensor factorization via alternating direction method of multipliers," *IEEE Transactions on Signal Processing*, vol. 63, no. 20, pp. 5450–5463, 2015.

APPENDIX

A. Effective Partitioning, and Error Analysis

In order to facilitate error analysis for the 1-scale MS-HoSVD that is similar to the types of error analysis available for various HoSVD-based low-rank approximation strategies (see, e.g., [28]), we will engage in a more in depth discussion of condition (30) herein. Recall that the partition of $\mathcal{W}_0$ formed by the restriction matrices $\mathbf{R}_k^{(n)}$ in (20) – (22) is called *effective* if there exists another *pessimistic* partitioning of $\mathcal{W}_0$ via restriction matrices $\left\{\tilde{\mathbf{R}}_k^{(n)}\right\}_{k=1}^{K}$ together with a bijection $f:[K]\to[K]$ such that

$$\sum_{n=1}^{N}\left\|\mathcal{X}|_k\times_n\left(\mathbf{I}-\mathbf{Q}_k^{(n)}\right)\right\|^2\leqslant\sum_{n=1}^{N}\left\|\mathcal{W}_0\times_n\tilde{\mathbf{R}}_{f(k)}^{(n)}\left(\mathbf{I}-\tilde{\mathbf{P}}^{(n)}\right)\bigtimes_{h\neq n}^{N}\tilde{\mathbf{R}}_{f(k)}^{(h)}\right\|^2 \tag{37}$$

holds for each $k\in[K]$. In (37) the $\left\{\tilde{\mathbf{P}}^{(n)}\right\}$ are the orthogonal projection matrices obtained from the HoSVD of $\mathcal{W}_0$ with ranks $\tilde{r}_n\geqslant\bar{r}_n\geqslant r_n$ (i.e., where each $\tilde{\mathbf{P}}^{(n)}$ projects onto the top $\tilde{r}_n$ left singular vectors of the matricization $\mathbf{W}_{0,(n)}$). Below we will show that (37) holding for $\mathcal{W}_0$ implies that the error $\|\mathcal{W}_1\|$ resulting from our 1^{st}-scale approximation in (27) is less than an upper bound of the type given for a high-rank standard HoSVD-based approximation (29) in [28].

Considering condition (37) above, we note that experiments show that it is regularly satisfied on real datasets when (i) the effective restriction matrices $\left\{\mathbf{R}_k^{(n)}\right\}_{k=1}^{K}$ in (20) – (22) are first formed by clustering the rows of each unfolding of $\mathcal{W}_0$ using, e.g., local subspace analysis (LSA), after which (ii) pessimistic restriction matrices $\left\{\tilde{\mathbf{R}}_k^{(n)}\right\}_{k=1}^{K}$ are randomly generated in order to create another (random) partition of $\mathcal{W}_0$ into K different disjoint subtensors for comparison. The bijection f can then be created by, e.g., (i) sorting the left-hand side errors in (37) for each $k\in[k]$, (ii) sorting the right-hand side errors in (37) for each $k\in[K]$, and then (iii) matching the largest left-hand and right-hand errors for comparision, the second largest left-hand and right-hand errors for comparision, etc.. When checked in this way the sorted right-hand side errors often dominate (entrywise) the sorted left-hand side errors for various reasonable ranks $\bar{r}_n=\tilde{r}_n=r_H=r_0+c^{N-1}r_1$ (as a function of r_0 and r_1 with, e.g., $c=2$) on every dataset considered in Section IV above, thereby verifying that (37) does indeed regularly hold.

We will now begin to prove Theorem 1 with a lemma that shows our subtensor-based approximation of $\mathcal{W}_0$ is accurate whenever (37) is satisfied.

Lemma 3. *Let $\mathcal{W}_0=\mathcal{X}-\hat{\mathcal{X}}_0\in\mathbb{R}^{I_1\times I_2\times\ldots I_N}$. Suppose that $\left\{\mathbf{R}_k^{(n)}\right\}$ is a collection of effective restriction matrices that form an effective partition of $\mathcal{W}_0$ with respect to a pessimistic partition formed via pessimistic restriction matrices $\left\{\tilde{\mathbf{R}}_k^{(n)}\right\}$ as per* (37) *above. Similarly, let $\tilde{\mathbf{P}}^{(n)}$ be the rank $\tilde{r}_n\geqslant\bar{r}_n$ $\forall n$ orthogonal projection matrices from* (37) *obtained via the truncated HoSVD of $\mathcal{W}_0$ as above. Then,*

$$\left\|\mathcal{W}_0-\hat{\mathcal{X}}_1\right\|^2=\left\|\left(\mathcal{X}-\hat{\mathcal{X}}_0\right)-\sum_{k=1}^{K}\left(\left(\mathcal{X}-\hat{\mathcal{X}}_0\right)\bigtimes_{n=1}^{N}\mathbf{Q}_k^{(n)}\right)\right\|^2\leqslant\sum_{n=1}^{N}\left\|\left(\mathcal{X}-\hat{\mathcal{X}}_0\right)\times_n\left(\mathbf{I}-\tilde{\mathbf{P}}^{(n)}\right)\right\|^2.$$

Proof. We have that

$$\begin{aligned}
\left\|\mathcal{W}_0-\hat{\mathcal{X}}_1\right\|^2&=\left\|\mathcal{W}_0-\sum_{k=1}^{K}\mathcal{W}_0\bigtimes_{n=1}^{N}\mathbf{Q}_k^{(n)}\right\|^2 && \text{(Using (12) and (26))}\\
&=\left\|\sum_{k=1}^{K}\mathcal{W}_0\bigtimes_{n=1}^{N}\mathbf{R}_k^{(n)}-\sum_{k=1}^{K}\mathcal{W}_0\bigtimes_{n=1}^{N}\mathbf{Q}_k^{(n)}\mathbf{R}_k^{(n)}\right\|^2 && \text{(Using (21), (22), and (25))}\\
&=\left\|\sum_{k=1}^{K}\mathcal{W}_0\bigtimes_{n=1}^{N}\left(\mathbf{R}_k^{(n)}-\mathbf{Q}_k^{(n)}\mathbf{R}_k^{(n)}\right)\right\|^2 && \text{(Using Lemma 1)}\\
&=\sum_{k=1}^{K}\left\|\mathcal{X}|_k\bigtimes_{n=1}^{N}\left(\mathbf{I}-\mathbf{Q}_k^{(n)}\right)\right\|^2. && \text{(Using Lemma 1, (21), (25), and support disjointness)} \quad (38)
\end{aligned}$$

Applying lemmas 1 and 2 to (38) we can now see that

$$\begin{aligned}
\left\|\mathcal{W}_0-\hat{\mathcal{X}}_1\right\|^2&=\sum_{k=1}^{K}\sum_{n=1}^{N}\left\|\mathcal{X}|_k\bigtimes_{h=1}^{n-1}\mathbf{Q}_k^{(h)}\times_n\left(\mathbf{I}-\mathbf{Q}_k^{(n)}\right)\right\|^2\\
&\leqslant\sum_{k=1}^{K}\sum_{n=1}^{N}\left\|\mathcal{X}|_k\times_n\left(\mathbf{I}-\mathbf{Q}_k^{(n)}\right)\right\|^2
\end{aligned}$$

since the $\mathbf{Q}_k^{(n)}$ matrices are orthogonal projections. Using assumption (37) we now get that

$$\begin{aligned}\left\|\mathcal{W}_0 - \hat{\mathcal{X}}_1\right\|^2 &\leqslant \sum_{k=1}^{K}\sum_{n=1}^{N}\left\|\mathcal{W}_0 \times_n \tilde{\mathbf{R}}_k^{(n)}\left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right)\bigtimes_{h\neq n}^{N}\tilde{\mathbf{R}}_k^{(h)}\right\|^2 \\ &= \sum_{n=1}^{N}\left\|\mathcal{W}_0 \times_n \left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right)\right\|^2\end{aligned}$$

where we have used the fact that the pessimistic restriction matrices $\tilde{\mathbf{R}}_k^{(n)}$ partition $\mathcal{W}_0$ in the last line. □

Lemma 3 indicates that the error in approximating $\mathcal{W}_0$ via low-rank approximations of its effective subtensors is potentially smaller than the error obtained by approximating $\mathcal{W}_0$ via (higher rank) truncated HoSVDs whenever (37) holds.[3] The following theorem shows that this good error behavior extends to the entire 1^{st} scale approximation provided by (27) whenever (37) holds.

Theorem 2 (Restatement of Theorem 1)**.** *Let $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \ldots \times I_N}$. Suppose that* (37) *holds. Then, the first scale approximation error given by MS-HoSVD* (27) *is bounded by*

$$\|\mathcal{W}_1\|^2 = \left\|\mathcal{X} - \hat{\mathcal{X}}_0 - \hat{\mathcal{X}}_1\right\|^2 \leqslant \sum_{n=1}^{N}\left\|\mathcal{X} \times_n \left(\mathbf{I} - \bar{\mathbf{P}}^{(n)}\right)\right\|^2$$

where $\left\{\bar{\mathbf{P}}^{(n)}\right\}$ are low-rank projection matrices of rank $\bar{r}_n \geqslant r_n$ obtained from the truncated HoSVD of $\mathcal{X}$ as per (29)*.*

Proof. Using (12) and (17) together with lemma 3 we can see that

$$\begin{aligned}\|\mathcal{W}_1\|^2 = \left\|\mathcal{X} - \hat{\mathcal{X}}_0 - \hat{\mathcal{X}}_1\right\|^2 = \left\|\mathcal{W}_0 - \hat{\mathcal{X}}_1\right\|^2 &\leqslant \sum_{n=1}^{N}\left\|\left(\mathcal{X} - \hat{\mathcal{X}}_0\right) \times_n \left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right)\right\|^2 \\ &\leqslant \sum_{n=1}^{N}\left\|\left(\mathcal{X} - \hat{\mathcal{X}}_0\right) \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\right\|^2 \qquad (39)\end{aligned}$$

where $\bar{\mathbf{Q}}^{(n)} \in \mathbb{R}^{I_n \times I_n}$ is the orthogonal projection matrix of rank $\tilde{r}_n$ which projects onto the subspace spanned by the top $\tilde{r}_n$ left singular vectors of $\mathbf{X}_{(n)}$. Here (39) holds because the orthogonal projection matrices $\tilde{\mathbf{P}}^{(n)}$ are chosen in (37) so that $\tilde{\mathbf{P}}^{(n)}\mathbf{W}_{0,(n)}$ is a best possible rank $\tilde{r}_n$ approximation to $\mathbf{W}_{0,(n)}$. As a result, we have that

$$\left\|\left(\mathcal{X} - \hat{\mathcal{X}}_0\right) \times_n \left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right)\right\|^2 = \left\|\left(\mathbf{I} - \tilde{\mathbf{P}}^{(n)}\right)\mathbf{W}_{0,(n)}\right\|_{\mathrm{F}}^2 \leqslant \left\|\left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\mathbf{W}_{0,(n)}\right\|_{\mathrm{F}}^2 = \left\|\left(\mathcal{X} - \hat{\mathcal{X}}_0\right) \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\right\|^2$$

must hold for each $n \in [N]$.

Continuing from (39) we can use the definition of $\hat{\mathcal{X}}_0$ in (28) to see that

$$\begin{aligned}\|\mathcal{W}_1\|^2 &\leqslant \sum_{n=1}^{N}\left\|\left(\mathcal{X} - \mathcal{X}\bigtimes_{h=1}^{N}\mathbf{P}^{(h)}\right) \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\right\|^2 \\ &= \sum_{n=1}^{N}\left\|\mathcal{X} \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right) - \mathcal{X}\bigtimes_{h=1}^{N}\mathbf{P}^{(h)} \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\right\|^2 \qquad (40)\end{aligned}$$

by lemma 1. Due to the definition of $\bar{\mathbf{Q}}^{(n)}$ together with the fact that its rank is $\tilde{r}_n \geqslant r_n$ we can see that $\left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\mathbf{P}^{(n)} = \mathbf{0}$. As a consequence, lemma 1 implies that $\mathcal{X} \times_{h=1}^{N} \mathbf{P}^{(h)} \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right) = \mathbf{0}$ for all $n \in [N]$. Continuing from (40) we now have that

$$\|\mathcal{W}_1\|^2 \leqslant \sum_{n=1}^{N}\left\|\mathcal{X} \times_n \left(\mathbf{I} - \bar{\mathbf{Q}}^{(n)}\right)\right\|^2.$$

Again appealing to the definition of both $\bar{\mathbf{Q}}^{(n)}$ and $\bar{\mathbf{P}}^{(n)}$ in (29), combined with the fact that $\tilde{r}_n \geqslant \bar{r}_n$, finally yields the desired result. □

We refer the reader to the strong empirical performance of MS-HoSVD in Section IV for additional evidence supporting the utility of (27) as a means of improving the compression performance of standard HoSVD-based compression techniques. In addition, we further refer the reader to Section V where it is empirically demonstrated that MS-HoSVD is also capable of selecting more informative features than HoSVD-based methods for the purposes of classification. These two facts together provide strong evidence that combining the use of clustering-enhanced multiscale approximation with existing tensor factorization techniques can lead to improved performance in multiple application domains.

[3]That is, the upper bound on the error provided by Lemma 3 is less than or equal to the upper bound on the error for truncated HoSVDs provided by, e.g., [28] when/if (37) holds.

B. Experiment for Error Analysis

In this experiment we evaluate the error obtained by the 1^{st} scale MS-HoSVD analysis of a tensor along the lines of the model described in Section III. Herein we consider a three-way tensor $\mathcal{X} \in \mathbb{R}^{20\times 20\times 20}$ that is the sum of two tensors as $\mathcal{X} = \mathcal{X}_0 + \mathcal{X}_1$ where $\mathcal{X}_0 \in \mathbb{R}^{20\times 20\times 20}$ has n-rank $(2,\ 2,\ 2)$, and $\mathcal{X}_1 \in \mathbb{R}^{20\times 20\times 20}$ is formed by concatenating 8 subtensors $\mathcal{X}_k \in \mathbb{R}^{10\times 10\times 10}$ each also with n-rank $(2,\ 2,\ 2)$. Low-rank approximations for $\mathcal{X}$ and its subtensors are always obtained via the truncated HoSVD. The 1-scale MS-HoSVD is applied with the ground truth partitions $\mathbf{R}_k^{(n)}$, partitions provided by Local Subspace Analysis (LSA) clustering $\hat{\mathbf{R}}_k^{(n)}$, and also with randomly chosen partitions $\tilde{\mathbf{R}}_k^{(n)}$ of the 0^{th}-scale residual error $\mathcal{W}_0$ into 8 different $10 \times 10 \times 10$ subtensors. For the LSA clustering the cluster numbers are selected as 2 along each mode also yielding 8 subtensors.

The 0^{th} scale n-rank for MS-HoSVD is selected as $(2,\ 2,\ 2)$, and the 1^{st} scale ranks are varied in the experiments as shown in Table VI. The normalized reconstruction error computed for these varying 1^{st} scale n-ranks can also be seen in Table VI. As seen there, using ground truth partition provides lower-rank subtensors, and using clustering as part of the 1-scale MS-HoSVD leads to much better approximations than HoSVD does in general.

TABLE VI: Mean and standard deviation for reconstruction error of low-rank approximations of $\mathcal{X}$ over 20 trials.

	Reconstruction Error		
0th scale rank	(2, 2, 2)	(2, 2, 2)	(2, 2, 2)
1st scale rank	(2, 2, 2)	(4, 4, 4)	(6, 6, 6)
Ground Truth Partitioning	0.2502 ± 0.0263	0.0304 ± 0.0077	- -
Clustering by LSA	0.3587 ± 0.0583	0.1099 ± 0.0391	0.0254 ± 0.0152
Random Partitioning	0.6095 ± 0.0398	0.3588 ± 0.0298	0.1855 ± 0.0251
rank	(4, 4, 4)	(8, 8, 8)	(12, 12, 12)
truncated HoSVD	0.5457 ±0.0449	0.2127 ± 0.0195	0.0733 ± 0.0129

In addition, the left (LHS) and right (RHS) sides in Theorem 1 are computed where the first scale projections $\mathbf{Q}_k^{(n)}$ each have rank $r_1 = 2$ and are obtained via both ground truth partitioning and clustering after the 0^{th} scale $\mathbf{P}^{(n)}$ are obtained from the truncated HoSVD of $\mathcal{X}$ with $r_0 = 2$. For comparison the $\bar{\mathbf{P}}^{(n)}$ are also computed from the truncated HoSVD of $\mathcal{X}$ with varying ranks $\bar{r}_n = \kappa r_0$ for $\kappa \in \{2\ ,3\ ,4\}$. In Table VII, we report the mean value and standard deviation of both the right hand side (RHS) and left hand side (LHS) of the error bound in Theorem 1. As seen in Table VII, Theorem 1 holds for the 1^{st} scale approximation of $\mathcal{X}$ via MS-HoSVD since the RHS errors based on the $\bar{\mathbf{P}}^{(n)}$ projections are larger than the LHS errors no matter whether the $\mathbf{Q}_k^{(n)}$ are obtained via ground truth partitioning or LSA clustering.

TABLE VII: Computed reconstruction error (mean and std.) corresponding to Theorem 1 for simulated data over 20 trials.

LHS		RHS		
0th scale rank	(2, 2, 2)	rank		
1st scale rank	(2, 2, 2)	(4, 4, 4)	(6, 6, 6)	(8, 8, 8)
Ground Truth Partitioning	0.2618 ± 0.0236	2.9928 ±0.5081	1.1835 ±0.3010	0.4228 ±0.0978
Clustering by LSA	0.3469 ± 0.0513			