Matrix Product State for Higher-Order Tensor Compression and Classification

Johann A. Bengua, Ho N. Phien, Hoang D. Tuan and Minh N. Do

***Abstract*—This paper introduces matrix product state (MPS) decomposition as a new and systematic method to compress multidimensional data represented by higher-order tensors. It solves two major bottlenecks in tensor compression: computation and compression quality. Regardless of tensor order, MPS compresses tensors to *matrices* of moderate dimension which can be used for classification. Mainly based on a successive sequence of singular value decompositions (SVD), MPS is quite simple to implement and arrives at the global optimal matrix, bypassing local alternating optimization, which is not only computationally expensive but cannot yield the global solution. Benchmark results show that MPS can achieve better classification performance with favorable computation cost compared to other tensor compression methods.**

***Index Terms*—Higher-order tensor compression and classification, supervised learning, matrix product state (MPS), tensor dimensionality reduction.**

I. Introduction

THERE is an increasing need to handle large multidimensional datasets that cannot efficiently be analyzed or processed using modern day computers. Due to the curse of dimensionality it is urgent to develop mathematical tools which can evaluate information beyond the properties of large matrices [1]. The essential goal is to reduce the dimensionality of multidimensional data, represented by tensors, with a minimal information loss by compressing the original tensor space to a lower-dimensional tensor space, also called the feature space [1]. Tensor decomposition is the most natural tool to enable such compressions [2].

Until recently, tensor compression is merely based on Tucker decomposition (TD) [3], also known as higher-order singular value decomposition (HOSVD) when orthogonality constraints on factor matrices are imposed [4]. TD is also an important tool for solving problems related to feature extraction, feature selection and classification of large-scale multidimensional datasets in various research fields. Its well-known application in computer vision was introduced in [5] to analyze some ensembles of facial images represented by fifth-order tensors. In data mining, the HOSVD was also applied to identify handwritten digits [6]. In addition, the HOSVD has been applied in neuroscience, pattern analysis, image classification and signal processing [7], [8], [9]. The higher-order orthogonal iteration (HOOI) [10] is an alternating least squares (ALS) for finding the TD approximation of a tensor. Its application to independent component analysis (ICA) and simultaneous matrix diagonalization was investigated in [11]. Another TD-based method is multilinear principal component analysis (MPCA) [12], an extension of classical principal component analysis (PCA), which is closely related to HOOI. Meanwhile, TD suffers the following conceptual bottlenecks in tensor compression:

- *Computation.* TD compresses an Nth-order tensor in tensor space $\mathbb{R}^{I_1\times I_2\times\cdots I_N}$ of large dimension $I = \prod_{j=1}^{N} I_j$ to its Nth-order core tensor in a tensor space $\mathbb{R}^{\Delta_1\times\Delta_2\times\cdots\Delta_N}$ of smaller dimension $N_f = \prod_{j=1}^{N}\Delta_j$ by using N factor matrices of size $I_j \times \Delta_j$. Computation of these N factor matrices is computationally intractable. Instead, each factor matrix is alternatingly optimized with all other $N-1$ factor matrices held fixed, which is still computationally expensive. Practical application of the TD-based compression is normally limited to small-order tensors.
- *Compression quality.* TD is an effective representation of a tensor only when the dimension of its core tensor is fairly large [2]. Restricting dimension $N_f = \prod_{j=1}^{N}\Delta_j$ to a moderate size for tensor classification results in significant lossy compression, making TD-based compression a highly heuristic procedure for classification. It is also almost impossible to tune $\Delta_j \le I_j$ among $\prod_{j=1}^{N}\Delta_j \le \bar{N}_f$ for a prescribed $\bar{N}_f$ to have a better compression.

In this paper, we introduce the matrix product state (MPS) decomposition [13], [14], [15], [16] as a new method to compress tensors, which fundamentally circumvent all the above bottlenecks of TD-based compression. Namely,

- *Computation.* The MPS decomposition is fundamentally different from the TD in terms of its geometric structure as it is made up of local component tensors with maximum order three. Consequently, using the MPS decomposition for large higher-order tensors can potentially avoid the computational bottleneck of the TD and related algorithms. Computation for orthogonal common factors in MPS is based on successive SVDs without any recursive local optimization procedure and is very

Johann A. Bengua was with the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia. He is now with Teradata Australia and New Zealand (Email: jbengua@gmail.com).

Ho N. Phien was with the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia. He is now with Westpac Group, Australia (Email: phien10@gmail.com).

Hoang D. Tuan is with the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia (Email: tuan.hoang@uts.edu.au).

Minh N. Do is with the Department of Electrical and Computer Engineering and the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana, IL 61801 USA (Email: minhdo@illinois.edu).

efficient with low-cost.

- *Compression quality.* MPS compresses Nth-order tensors to their core matrices of size $\mathbb{R}^{N_1\times N_2}$. The dimension $N_f = N_1 N_2$ can be easily tuned to a moderate size with minimum information loss by pre-positioning the core matrix in the MPS decomposition.

MPS has been proposed and applied to study quantum many-body systems with great success, prior to its introduction to the mathematics community under the name tensor-train (TT) decomposition [17]. However, to the best of our knowledge its application to machine learning and pattern analysis has not been proposed.

Our main contribution is summarized as follows:

- Propose MPS decomposition as a new and systematic method for compressing tensors of arbitrary order to matrices of moderate dimension, which circumvents all existing bottlenecks in tensor compression;
- Develop MPS decomposition tailored for optimizing the dimensionality of the core matrices and the compression quality. Implementation issues of paramount importance for practical computation are discussed in detail. These include tensor mode permutation, tensor bond dimension control, and positioning the core matrix in MPS;
- Extensive experiments are performed along with comparisons to existing state-of-the-art tensor objection recognition (classification) methods to show its advantage.

A preliminary result of this work was presented in [18]. In the present paper, we rigorously introduce the MPS as a new and systematic approach to tensor compression for classification, with computational complexity and efficiency analysis. Furthermore, new datasets as well a new experimental design showcasing computational time and classification success rate (CSR) benchmarks are included.

The rest of the paper is structured as follows. Section II provides a rigorous mathematical analysis comparing MPS and TD in the context of tensor compression. Section III is devoted to MPS tailored for effective tensor compression, which also includes a computational complexity analysis comparing MPS to HOOI, MPCA and the CANDECOMP/PARAFAC (CP)-based algorithm [2] uncorrelated multilinear discriminant analysis with regularization (R-UMLDA) [19]. In Section IV, experimental results are shown to benchmark all algorithms in classification performance[1] and training time. Lastly, Section V concludes the paper.

II. MPS vs TD decomposition in tensor compression

We introduce some notations and preliminaries of multilinear algebra [2]. Zero-order tensors are scalars and denoted by lowercase letters, e.g., x. A first-order tensor is a vector and denoted by boldface lowercase letters, e.g., $\mathbf{x}$. A matrix is a second-order tensor and denoted by boldface capital letters, e.g., $\mathbf{X}$. A higher-order tensor (tensors of order three and above) are denoted by boldface calligraphic letters, e.g., $\boldsymbol{\mathcal{X}}$. Therefore, a general Nth-order tensor of size $I_1\times I_2\times\cdots\times I_N$ can be defined as $\boldsymbol{\mathcal{X}}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$, where each I_i is the dimension of its mode i. We also denote x_i, x_{ij} and $x_{i_1\cdots i_N}$ as the ith entry $\mathbf{x}(i)$, (i,j)th entry $\mathbf{X}(i,j)$ and $(i_1,\cdots,i_N)$th entry $\boldsymbol{\mathcal{X}}(i_1,\cdots,i_N)$ of vector $\mathbf{x}$, matrix $\mathbf{X}$ and higher-order tensor $\boldsymbol{\mathcal{X}}$, respectively.

Mode-n matricization (also known as mode-n unfolding or flattening) of $\boldsymbol{\mathcal{X}}$ is the process of unfolding or reshaping $\boldsymbol{\mathcal{X}}$ into a matrix $\mathbf{X}_{(n)}\in\mathbb{R}^{I_n\times(\prod_{i\neq n}I_i)}$ such that $\mathbf{X}_{(n)}(i_n,j)=\boldsymbol{\mathcal{X}}(i_1,\cdots,i_n,\cdots,i_N)$ for $j=1+\sum_{k=1,k\neq n}^{N}(i_k-1)\prod_{m=1,m\neq n}^{k-1}I_m$. We also define the dimension of $\boldsymbol{\mathcal{X}}$ as $\prod_{n=1}^{N}I_n$. The mode-n product of $\boldsymbol{\mathcal{X}}$ with a matrix $\mathbf{A}\in\mathbb{R}^{J_n\times I_n}$ is denoted as $\boldsymbol{\mathcal{X}}\times_n\mathbf{A}$, which is a Nth-order tensor of size $I_1\times\cdots\times I_{n-1}\times J_n\times I_{n+1}\times\cdots\times I_N$ such that

$$(\boldsymbol{\mathcal{X}}\times_n\mathbf{A})(i_1,\cdots,i_{n-1},j_n,i_{n+1},\cdots,i_N) = \sum_{i_n=1}^{I_n}\boldsymbol{\mathcal{X}}(i_1,\cdots,i_n,\cdots i_N)\mathbf{A}(j_n,i_n).$$

The Frobenius norm of $\boldsymbol{\mathcal{X}}$ is defined as $||\boldsymbol{\mathcal{X}}||_F = (\sum_{i_1=1}^{I_1}\sum_{i_2=1}^{I_2}\cdots\sum_{i_N=1}^{I_N}x^2_{i_1 i_2\cdots i_N})^{1/2}$.

We are concerned with the following problem of tensor compression for supervised learning:

Based on K training Nth-order tensors $\boldsymbol{\mathcal{X}}^{(k)}\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N}$ ($k=1,2,\ldots,K$), find common factors to compress both training tensor $\boldsymbol{\mathcal{X}}^{(k)}$ and test tensors $\boldsymbol{\mathcal{Y}}^{(\ell)}$ ($\ell=1,\cdots,L$) to a feature space of moderate dimension to enable classification.

A. TD-based tensor compression

Until now, only TD has been proposed to address this problem [7]. More specifically, the K training sample tensors are firstly concatenated along the mode $(N+1)$ to form an $(N+1)$th-order tensor $\boldsymbol{\mathcal{X}}$ as

$$\boldsymbol{\mathcal{X}}=[\boldsymbol{\mathcal{X}}^{(1)}\boldsymbol{\mathcal{X}}^{(2)}\cdots\boldsymbol{\mathcal{X}}^{(K)}]\in\mathbb{R}^{I_1\times I_2\times\cdots\times I_N\times K}. \tag{1}$$

TD-based compression such as HOOI [10] is then applied to have the approximation

$$\boldsymbol{\mathcal{X}} \approx \boldsymbol{\mathcal{R}}\times_1\mathbf{U}^{(1)}\times_2\mathbf{U}^{(2)}\cdots\times_N\mathbf{U}^{(N)}, \tag{2}$$

where each matrix $\mathbf{U}^{(j)}\in\mathbb{R}^{I_j\times\Delta_j}$ ($j=1,2,\ldots,N$) is orthogonal, i.e. $\mathbf{U}^{(j)T}\mathbf{U}^{(j)}=\mathbf{I}$ ($\mathbf{I}\in\mathbb{R}^{\Delta_j\times\Delta_j}$ denotes the identity matrix). It is called a *common factor* matrix and can be thought of as the principal components in each mode j. The parameters Δ_j satisfying

$$\Delta_j\leq\text{rank}(\mathbf{X}_{(j)}) \tag{3}$$

are referred to as the compression ranks of the TD.

The $(N+1)$th-order core tensor $\boldsymbol{\mathcal{R}}$ and common factor matrices $\mathbf{U}^{(j)}\in\mathbb{R}^{I_j\times\Delta_j}$ are supposed to be found from the following nonlinear least squares

$$\begin{aligned}&\min_{\substack{\boldsymbol{\mathcal{R}}\in\mathbb{R}^{\Delta_1\times\cdots\times\Delta_N\times K},\\ \mathbf{U}^{(j)}\in\mathbb{R}^{I_j\times\Delta_j},j=1,\ldots,N}} \Phi(\boldsymbol{\mathcal{R}},\mathbf{U}^{(1)},\cdots,\mathbf{U}^{(N)})\\ &\text{subject to}\quad(\mathbf{U}^{(j)})^T\mathbf{U}^{(j)}=\mathbf{I}, j=1,...,N,\end{aligned} \tag{4}$$

[1]HOOI, MPCA and R-UMLDA can be combined in series with simple classifiers for recognition tasks. This variety of algorithms allows for a good comparison to the proposed methods.

where $\Phi(\boldsymbol{\mathcal{R}}, \mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}) := ||\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{R}} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}||_F^2$ The optimization problem (4) is computationally intractable, which could be addressed only by alternating least squares (ALS) in each $\mathbf{U}^{(j)}$ (with other $\mathbf{U}^{(\ell)}$, $\ell \neq j$ held fixed) [10]:

$$\min_{\substack{\boldsymbol{\mathcal{R}}^{(j)} \in \mathbb{R}^{\Delta_1 \times \cdots \times \Delta_N \times K}, \\ \mathbf{U}^{(j)} \in \mathbb{R}^{I_j \times \Delta_j}}} \Phi^{(j)}(\boldsymbol{\mathcal{R}}^{(j)}, \mathbf{U}^{(j)}) \quad \text{subject to} \quad (\mathbf{U}^{(j)})^T \mathbf{U}^{(j)} = \mathbf{I}, \tag{5}$$

where $\Phi^{(j)}(\boldsymbol{\mathcal{R}}^{(j)}, \mathbf{U}^{(j)}) := ||\boldsymbol{\mathcal{X}} - \boldsymbol{\mathcal{R}}^{(j)} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}||_F^2$. The computation complexity per one iteration consisting of N ALS (5) is [20, p. 127]

$$\mathcal{O}(K\Delta I^N + NKI\Delta^{2(N-1)} + NK\Delta^{3(N-1)}) \tag{6}$$

for

$$I_j \equiv I \quad \text{and} \quad \Delta_j \equiv \Delta, j = 1, 2, ..., N. \tag{7}$$

The optimal $(N+1)$th-order core tensor $\boldsymbol{\mathcal{R}} \in \mathbb{R}^{\Delta_1 \times \cdots \times \Delta_N \times K}$ in (4) is seen as the concatenation of compressed $\widetilde{\boldsymbol{\mathcal{X}}}^{(k)} \in \mathbb{R}^{\Delta_1 \times \cdots \times \Delta_N}$ of the sample tensors $\boldsymbol{\mathcal{X}}^{(k)} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$, $k = 1, \cdots, K$:

$$\begin{aligned} \boldsymbol{\mathcal{R}} &= [\widetilde{\boldsymbol{\mathcal{X}}}^{(1)} \widetilde{\boldsymbol{\mathcal{X}}}^{(2)} \cdots \widetilde{\boldsymbol{\mathcal{X}}}^{(K)}] \\ &= \boldsymbol{\mathcal{X}} \times_1 (\mathbf{U}^{(1)})^T \cdots \times_N (\mathbf{U}^{(N)})^T. \end{aligned} \tag{8}$$

Accordingly, the test tensors $\boldsymbol{\mathcal{Y}}^{(\ell)}$ are compressed to

$$\widetilde{\boldsymbol{\mathcal{Y}}}^{(\ell)} = \boldsymbol{\mathcal{Y}}^{(\ell)} \times_1 (\mathbf{U}^{(1)})^T \cdots \times_N (\mathbf{U}^{(N)})^T \in \mathbb{R}^{\Delta_1 \times \cdots \times \Delta_N}. \tag{9}$$

The number

$$N_f = \prod_{j=1}^{N} \Delta_j \tag{10}$$

thus represents the dimension of the feature space $\mathbb{R}^{\Delta_1 \times \cdots \times \Delta_N}$.

Putting aside the computational intractability of the optimal factor matrices $\mathbf{U}^{(j)}$ in (4), the TD-based tensor compression by (8) and (9) is a systematic procedure only when the right hand side of (2) provides a good approximation of $\boldsymbol{\mathcal{X}}$, which is impossible for small Δ_j satisfying (3) [2]. In other words, the compression of large dimensional tensors to small dimensional tensors results in substantial lossy compression under the TD framework. Furthermore, one can see the value of (5) is lower bounded by

$$\sum_{i=1}^{r_j - \Delta_j - 1} s_i, \tag{11}$$

where $r_j := \text{rank}(\boldsymbol{\mathcal{X}}_{(j)})$ and $\{s_{r_j}, \cdots, s_1\}$ is the set of nonzero eigenvalues of the positive definite matrix $\boldsymbol{\mathcal{X}}_{(j)}(\boldsymbol{\mathcal{X}}_{(j)})^T$ in decreasing order. Since the matrix $\boldsymbol{\mathcal{X}}_{(j)} \in \mathbb{R}^{I_j \times (K \prod_{\ell \neq j} I_\ell)}$ is highly unbalanced as a result of tensor matricization along one mode versus the rest, it is almost full-row (low) rank ($r_j \approx I_j$) and its squared $\boldsymbol{\mathcal{X}}_{(j)}(\boldsymbol{\mathcal{X}}_{(j)})^T$ of size $I_j \times I_j$ is well-conditioned in the sense that its eigenvalues do not decay quickly. As a consequence, (11) cannot be small for small Δ_j so the ALS (5) cannot result in a good approximation. The information loss with the least square (5) is thus more than

$$-\sum_{i=1}^{r_j - \Delta_j - 1} \frac{s_i}{\sum_{i=1}^{r_j} s_i} \log_2 \frac{s_i}{\sum_{i=1}^{r_j} s_i}, \tag{12}$$

which is really essential in the von Neumann entropy [21] of $\boldsymbol{\mathcal{X}}_{(j)}$:

$$-\sum_{i=1}^{r_j} \frac{s_i}{\sum_{i=1}^{r_j} s_i} \log_2 \frac{s_i}{\sum_{i=1}^{r_j} s_i}. \tag{13}$$

Note that each entropy (13) quantifies only local correlation between mode j and the rest [22]. The MPCA [12] aims at (4) with

$$\boldsymbol{\mathcal{X}} = [(\boldsymbol{\mathcal{X}}^{(1)} - \bar{\boldsymbol{\mathcal{X}}}) \cdots (\boldsymbol{\mathcal{X}}^{(K)} - \bar{\boldsymbol{\mathcal{X}}})]$$

with $\bar{\boldsymbol{\mathcal{X}}} = \frac{1}{K+L}(\sum_{k=1}^{K} \boldsymbol{\mathcal{X}}^{(k)} + \sum_{\ell=1}^{L} \boldsymbol{\mathcal{Y}}^{\ell)})$. With such definition of $\boldsymbol{\mathcal{X}}$, $(N+1)$th-order core tensor $\boldsymbol{\mathcal{X}}$ is the concatenation of principal components of $\boldsymbol{\mathcal{X}}^{(k)}$, while principal components of $\boldsymbol{\mathcal{Y}}^{(\ell)}$ is defined by $(\boldsymbol{\mathcal{Y}}^{(\ell)} - \bar{\boldsymbol{\mathcal{X}}}) \times_1 (\boldsymbol{U}^{(1)})^T \cdots \times_N (\boldsymbol{U}^{(N)})^T$. Thus, MPCA suffers the similar conceptual drawbacks inherent by TD. Particularly, restricting $N_f = \prod_{j=1}^{N} \Delta_j$ to a moderate size leads to ignoring many important principle components.

B. *MPS-based tensor compression*

We now present a novel approach to extract tensor features, which is based on MPS. Firstly, *permute* all modes of the tensor $\boldsymbol{\mathcal{X}}$ and position mode K such that such that

$$\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots I_{n-1} \times K \times I_n \cdots \times I_N}, \tag{14}$$

$I_1 \geq \cdots \geq I_{n-1}$ and $I_n \leq \cdots ... \leq I_N$. The elements of $\boldsymbol{\mathcal{X}}$ can be presented in the following *mixed-canonical form* [23] of the matrix product state (MPS) or tensor train (TT) decomposition [16], [14], [15], [17]:

$$\begin{aligned} x_{i_1 \cdots k \cdots i_N} &= x^{(k)}_{i_1 \cdots i_n \cdots i_N} \\ &\approx \mathbf{B}^{(1)}_{i_1} \cdots \mathbf{B}^{(n-1)}_{i_{n-1}} \mathbf{G}^{(n)}_k \mathbf{C}^{(n+1)}_{i_n} \cdots \mathbf{C}^{(N+1)}_{i_N}, \end{aligned} \tag{15}$$

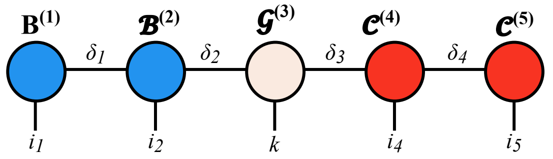
where matrices $\mathbf{B}^{(j)}_{i_j}$ and $\mathbf{C}^{(j)}_{i_{(j-1)}}$ (the upper index "(j)" denotes the position j of the matrix in the chain) of size $\Delta_{j-1} \times \Delta_j$ ($\Delta_0 = \Delta_{N+1} = 1$), are called "left" and "right" *common factors* which satisfy the following orthogonality conditions:

$$\sum_{i_j=1}^{I_j} (\mathbf{B}^{(j)}_{i_j})^T \mathbf{B}^{(j)}_{i_j} = \mathbf{I}, \quad (j = 1, \ldots, n-1) \tag{16}$$

and

$$\sum_{i_{j-1}=1}^{I_{j-1}} \mathbf{C}^{(j)}_{i_{j-1}} (\mathbf{C}^{(j)}_{i_{j-1}})^T = \mathbf{I}, \quad (j = n+1, \ldots, N+1) \tag{17}$$

respectively, where $\mathbf{I}$ denotes the identity matrix. Each matrix $\mathbf{G}^{(n)}_k$ of dimension $\Delta_{n-1} \times \Delta_n$ is the compression of the training tensor $\boldsymbol{\mathcal{X}}^{(k)}$. The parameters Δ_j are called the bond



$\mathbf{B}^{(1)}$
$\boldsymbol{\mathcal{B}}^{(2)}$
$\boldsymbol{\mathcal{G}}^{(3)}$
$\boldsymbol{\mathcal{C}}^{(4)}$
$\boldsymbol{\mathcal{C}}^{(5)}$
δ_1
δ_2
δ_3
δ_4
i_1
i_2
k
i_4
i_5

Fig. 1: The mixed-canonical MPS decomposition of $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times I_2\times K\times I_4\times I_5}$. Each circle represents a tensor, where lines protruding from the tensor represent different modes within that tensor. Lines connecting adjacent circles represent shared indices. The tensors $\mathbf{B}^{(1)}, \boldsymbol{\mathcal{B}}^{(2)}$ (in blue) are in left-orthogonal form, whereas the tensors $\boldsymbol{\mathcal{C}}^{(4)}, \mathbf{C}^{(5)}$ (in red) are right-orthogonal. The core tensor $\boldsymbol{\mathcal{G}}^{(n)}$ is neither right- or left-orthogonal.

dimensions or compression ranks of the MPS. See Fig. 1 for an example of (15) for a fifth-order tensor with $n = 3$.

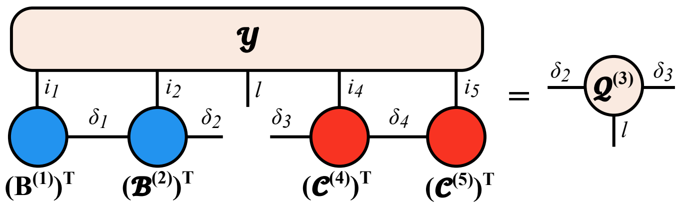
Using the common factors $\mathbf{B}^{(j)}_{i_j}$ and $\mathbf{C}^{(j)}_{i_{(j-1)}}$, we can extract the core matrices for the test tensors $\boldsymbol{\mathcal{Y}}^{(\ell)}$ as follows. We permute all $\boldsymbol{\mathcal{Y}}^{(\ell)}$, $\ell = 1, \cdots, L$ to ensure the compatibility between the training and test tensors. The compressed matrix $\mathbf{Q}^{(n)}_\ell \in \mathbb{R}^{\Delta_{n-1}\times\Delta_n}$ of the test tensor $\boldsymbol{\mathcal{Y}}^{(\ell)}$ is then given by

$$\mathbf{Q}^{(n)}_\ell = \sum_{i_1,\ldots,i_N} (\mathbf{B}^{(1)}_{i_1})^T \cdots (\mathbf{B}^{(n-1)}_{i_{n-1}})^T y^{(\ell)}_{i_1\cdots\cdots i_N} (\mathbf{C}^{(n+1)}_{i_n})^T \cdots (\mathbf{C}^{(N+1)}_{i_N})^T. \quad (18)$$

The dimension

$$N_f = \Delta_{n-1}\Delta_n \quad (19)$$

is the number of reduced features. An example of (18) for a fifth-order tensor $\boldsymbol{\mathcal{Y}}$ with $n = 3$ in Fig. 2.



$\boldsymbol{\mathcal{Y}}$
i_1
i_2
l
i_4
i_5
δ_1
δ_2
δ_3
δ_4
$(\mathbf{B}^{(1)})^T$
$(\boldsymbol{\mathcal{B}}^{(2)})^T$
$(\boldsymbol{\mathcal{C}}^{(4)})^T$
$(\boldsymbol{\mathcal{C}}^{(5)})^T$
=
$\boldsymbol{\mathcal{Q}}^{(3)}$
δ_2
δ_3
l

Fig. 2: Obtaining a core tensor $\boldsymbol{\mathcal{Q}}^{(3)}$ from a fifth-order tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{I_1\times I_2\times L\times I_4\times I_5}$.

III. Tailored MPS for tensor compression

The advantage of MPS for tensor compression is that the order N of a tensor does not affect directly the feature number N_f in Eq. (19), which is only determined strictly by the product of the aforementioned bond dimensions Δ_{n-1} and Δ_n. In order to keep Δ_{n-1} and Δ_n to a moderate size, it is important to control the bond dimensions Δ_j, and also to optimize the positions of tensor modes as we address in this section. In what follows, for a matrix $\boldsymbol{X}$ we denote $\boldsymbol{X}(i,:)$ ($\boldsymbol{X}(:,j)$, resp.) as its ith row (jth column, resp.), while for a third-order tensor $\boldsymbol{\mathcal{X}}$ we denote $\boldsymbol{\mathcal{X}}(:,\ell,:)$ as a matrix such that its (i_1, i_3)th entry is $\boldsymbol{\mathcal{X}}(i_1,\ell,i_3)$. For a Nth-order tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ we denote $\mathbf{X}_{[j]} \in \mathbb{R}^{(I_1I_2\cdots I_j)\times(I_{j+1}\cdots K\cdots I_N)}$ as its *mode*-$(1,2,\ldots,j)$ *matricization*. It is obvious that $\mathbf{X}_{[1]} = \mathbf{X}_{(1)}$.

A. Adaptive bond dimension control in MPS

To decompose the training tensor $\boldsymbol{\mathcal{X}}$ into the MPS according to Eq. (15), we apply two successive sequences of SVDs to the tensor which include left-to-right sweep for computing the left common factors $\mathbf{B}^{(1)}_{i_1}, \ldots, \mathbf{B}^{(n-1)}_{i_{n-1}}$, and right-to-left sweep for computing the right common factors $\mathbf{C}^{(n+1)}_{i_n}, \ldots, \mathbf{C}^{(N+1)}_{i_N}$ and the core matrix $\mathbf{G}^{(n)}_k$ in Eq. (15) as follows:

- *Left-to-right sweep for left factor computation:*

The left-to-right sweep involves acquiring matrices $\mathbf{B}^{(j)}_{i_j}$ ($i_j = 1,\ldots,I_j$; $j = 1,\ldots,n-1$) fulfilling orthogonality condition in Eq. (16). Start by performing the mode-1 matricization of $\boldsymbol{\mathcal{X}}$ to obtain

$$\mathbf{W}^{(1)} := \boldsymbol{\mathcal{X}}_{[1]} = \boldsymbol{\mathcal{X}}_{(1)} \in \mathbb{R}^{I_1\times(I_2\cdots K\cdots I_N)}.$$

For

$$\Delta_1 \le \text{rank}(\mathbf{X}_{[1]}), \quad (20)$$

apply SVD to $\mathbf{W}^{(1)}$ to have the QR-approximation

$$\mathbf{W}^{(1)} \approx \mathbf{U}^{(1)}\mathbf{V}^{(1)} \in \mathbb{R}^{I_1\times(I_2\cdots K\cdots I_N)}, \quad (21)$$

where $\mathbf{U}^{(1)} \in \mathbb{R}^{I_1\times\Delta_1}$ is orthogonal:

$$(\mathbf{U}^{(1)})^T\mathbf{U}^{(1)} = \mathbf{I}, \quad (22)$$

and $\mathbf{V}^{(1)} \in \mathbb{R}^{\Delta_1\times(I_2\cdots K\cdots I_N)}$. Define the most left common factors by

$$\mathbf{B}^{(1)}_{i_1} = \mathbf{U}^{(1)}(i_1,:) \in \mathbb{R}^{1\times\Delta_1}, i_1 = 1,\cdots,I_1 \quad (23)$$

which satisfy the left-canonical constraint in Eq. (16) due to (22).
Next, reshape the matrix $\mathbf{V}^{(1)} \in \mathbb{R}^{\Delta_1\times(I_2\cdots K\cdots I_N)}$ to $\mathbf{W}^{(2)} \in \mathbb{R}^{(\Delta_1I_2)\times(I_3\cdots K\cdots I_N)}$. For

$$\Delta_2 \le \text{rank}(\mathbf{W}^{(2)}) \le \text{rank}(\mathbf{X}_{[2]}), \quad (24)$$

apply SVD to $\mathbf{W}^{(2)}$ for the QR-approximation

$$\mathbf{W}^{(2)} \approx \mathbf{U}^{(2)}\mathbf{V}^{(2)} \in \mathbb{R}^{(\Delta_1I_2)\times(I_3\cdots K\cdots I_N)}, \quad (25)$$

where $\mathbf{U}^{(2)} \in \mathbb{R}^{(\Delta_1I_2)\times\Delta_2}$ is orthogonal such that

$$(\mathbf{U}^{(2)})^T\mathbf{U}^{(2)} = \mathbf{I}, \quad (26)$$

and $\mathbf{V}^{(2)} \in \mathbb{R}^{\Delta_2\times(I_3\cdots K\cdots I_N)}$. Reshape the matrix $\mathbf{U}^{(2)} \in \mathbb{R}^{(\Delta_1I_2)\times\Delta_2}$ into a third-order tensor $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{\Delta_1\times I_2\times\Delta_2}$ to define the next common factors

$$\mathbf{B}^{(2)}_{i_2} = \boldsymbol{\mathcal{U}}(:,i_2,:) \in \mathbb{R}^{\Delta_1\times\Delta_2}, i_2 = 1,\cdots,I_2, \quad (27)$$

which satisfy the left-canonical constraint due to (26).
Applying the same procedure for determining $\mathbf{B}^{(3)}_{i_3}$ by reshaping the matrix $\mathbf{V}^{(2)} \in \mathbb{R}^{\Delta_2\times(I_3\cdots K\cdots I_N)}$ to

$$\mathbf{W}^{(3)} \in \mathbb{R}^{(\Delta_2I_3)\times(I_4\cdots K\cdots I_N)},$$

performing the SVD, and so on. This procedure is iterated till obtaining the last QR-approximation

$$\begin{aligned}\mathbf{W}^{(n-1)} \approx\ & \mathbf{U}^{(n-1)}\mathbf{V}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times(KI_n\cdots I_N)},\\ & \mathbf{U}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times\Delta_{n-1}},\\ & \mathbf{V}^{(n-1)} \in \mathbb{R}^{\Delta_{n-1}\times(KI_n\cdots I_N)},\end{aligned} \quad (28)$$

with $\mathbf{U}^{(n-1)}$ orthogonal:

$$\mathbf{U}^{(n-1)}(\mathbf{U}^{(n-1)})^T = \mathbf{I} \tag{29}$$

and reshaping $\mathbf{U}^{(n-1)} \in \mathbb{R}^{(\Delta_{n-2}I_{n-1})\times\Delta_{n-1}}$ into a third-order tensor $\boldsymbol{\mathcal{U}} \in \mathbb{R}^{\Delta_{n-2}\times I_{n-1}\times\Delta_{n-1}}$ to define the last left common factors

$$\mathbf{B}^{(n-1)}_{i_{n-1}} = \boldsymbol{\mathcal{U}}(:, i_{n-1}, :) \in \mathbb{R}^{\Delta_{n-2}\times\Delta_{n-1}}, i_{n-1} = 1, \cdots, I_{n-1}, \tag{30}$$

which satisfy the left-canonical constraint due to (29).

In a nutshell, after completing the left-to-right sweep, the elements of tensor $\boldsymbol{\mathcal{X}}$ are approximated by

$$x^{(k)}_{i_1\cdots i_{n-1}i_n\cdots i_{N+1}} \approx \mathbf{B}^{(1)}_{i_1}\cdots\mathbf{B}^{(n-1)}_{i_{n-1}}\mathbf{V}^{(n-1)}(:, ki_n\cdots i_N). \tag{31}$$

The matrix $\mathbf{V}^{(n-1)} \in \mathbb{R}^{\Delta_{n-1}\times(KI_n\cdots I_N)}$ is reshaped to $\mathbf{W}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times I_N}$ for the next right-to-left sweeping process.

- *Right-to-left sweep for right factor computation:*

Similar to left-to-right sweep, we perform a sequence of SVDs starting from the right to the left of the MPS to get the matrices $\mathbf{C}^{(j)}_{i_{j-1}}$ ($i_{j-1} = 1, \ldots, I_{j-1}$; $j = N+1, \ldots, n+1$) fulfilling the right-canonical condition in Eq. (17). To start, we apply the SVD to the matrix $\mathbf{W}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times I_N}$ obtained previously in the left-to-right sweep to have the RQ-approximation

$$\mathbf{W}^{(N)} \approx \mathbf{U}^{(N)}\mathbf{V}^{(N)}, \tag{32}$$

where $\mathbf{U}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times\Delta_N}$ and $\mathbf{V}^{(N)} \in \mathbb{R}^{\Delta_N\times I_N}$ is orthogonal:

$$\mathbf{V}^{(N)}(\mathbf{V}^{(N)})^T = \mathbf{I} \tag{33}$$

for

$$\Delta_N \leq \text{rank}(\mathbf{W}^{(N)}) \leq \text{rank}(\boldsymbol{\mathcal{X}}_{[N-1]}). \tag{34}$$

Define the most right common factors

$$\mathbf{C}^{(N+1)}_{i_N} = \mathbf{V}^{(N)}(:, i_N) \in \mathbb{R}^{\Delta_N\times 1}, i_N = 1, \cdots, I_N,$$

which satisfy the right-canonical constraint (17) due to (33). Next, reshape $\mathbf{U}^{(N)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-1})\times\Delta_N}$ into $\mathbf{W}^{(N-1)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-2})\times(I_{N-1}\Delta_N)}$ and apply the SVD to have the RQ-approximation

$$\mathbf{W}^{(N-1)} \approx \mathbf{U}^{(N-1)}\mathbf{V}^{(N-1)}, \tag{35}$$

where $\mathbf{U}^{(N-1)} \in \mathbb{R}^{(\Delta_{n-1}K\cdots I_{N-2})\times\Delta_{N-1}}$ and $\mathbf{V}^{(N-1)} \in \mathbb{R}^{\Delta_{N-1}\times(I_{N-1}\Delta_N)}$ is orthogonal:

$$\mathbf{V}^{(N-1)}(\mathbf{V}^{(N-1)})^T = \mathbf{I} \tag{36}$$

for

$$\Delta_{N-1} \leq \text{rank}(\mathbf{W}^{(N-1)}) \leq \text{rank}(\boldsymbol{\mathcal{X}}_{[N-2]}). \tag{37}$$

Reshape the matrix $\mathbf{V}^{(N-1)} \in \mathbb{R}^{\Delta_{N-1}\times(I_{N-1}\Delta_N)}$ into a third-order tensor $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{\Delta_{N-1}\times I_{N-1}\times\Delta_N}$ to define the next common factor

$$\mathbf{C}^{(N)}_{i_{N-1}} = \boldsymbol{\mathcal{V}}(:, i_{N-1}, :) \in \mathbb{R}^{\Delta_{N-1}\times\Delta_N} \tag{38}$$

which satisfy Eq. (17) due to (36).
This procedure is iterated till obtaining the last RQ-approximation

$$\begin{aligned}\mathbf{W}^{(n)} \quad &\approx \quad \mathbf{U}^{(n)}\mathbf{V}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K)\times(I_n\Delta_{n+1})},\\ &\quad\ \mathbf{U}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K)\times\Delta_n}, \mathbf{V}^{(n)} \in \mathbb{R}^{\Delta_n\times(I_n\Delta_{n+1})},\end{aligned} \tag{39}$$

with $\mathbf{V}^{(n)}$ orthogonal:

$$\mathbf{V}^{(n)}(\mathbf{V}^{(n)})^T = \mathbf{I} \tag{40}$$

for

$$\Delta_n \leq \text{rank}(\mathbf{W}^{(n)}) \leq \text{rank}(\boldsymbol{\mathcal{X}}_{[n-1]}). \tag{41}$$

Reshape $\mathbf{V}^{(n)} \in \mathbb{R}^{(\Delta_n)\times(I_n\Delta_{n+1})}$ into a third-order tensor $\boldsymbol{\mathcal{V}} \in \mathbb{R}^{\Delta_n\times I_n\times\Delta_{n+1}}$ to define the last right common factors

$$\mathbf{C}^{(n+1)}_{i_n} = \boldsymbol{\mathcal{V}}(:, i_n, :) \in \mathbb{R}^{\Delta_{n-1}\times\Delta_n}, i_n = 1, \cdots, I_n, \tag{42}$$

which satisfy (17) due to (40).

By reshaping $\mathbf{U}^{(n)} \in \mathbb{R}^{(\Delta_{n-1}K)\times\Delta_n}$ into a third-order tensor $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{\Delta_{n-1}\times K\times\Delta_n}$ to define $\mathbf{G}^{(n)}_k = \boldsymbol{\mathcal{G}}(:, k, :)$, $k = 1, \cdots .K$, we arrive at Eq. (15).

Note that the MPS decomposition described by Eq. (15) can be performed exactly or approximately depending on the bond dimensions Δ_j ($j = 1, \ldots, N$). The bond dimension truncation is of crucial importance to control the final feature number $N_f = \Delta_{n-1}\Delta_n$. To this end, we rely on thresholding the singular values of $\mathbf{W}^{(j)}$. With a threshold ϵ being defined in advance, we control Δ_j such that Δ_j largest singular values $s_1 \geq s_2 \geq ... \geq s_{\Delta_j}$ satisfy

$$\frac{\sum_{i=1}^{\Delta_j} s_i}{\sum_{i=1}^{r_j} s_j} \geq \epsilon, \tag{43}$$

for $r_j = \text{rank}(\mathbf{W}^{(j)})$. The information loss from the von Neumann entropy (13) of $\mathbf{W}^{(j)}$ by this truncation is given by (12). The entropy of each $\mathbf{W}^{(j)}$ provides the correlation degree between two sets of modes $1, \cdots, j$ and $j+1, \cdots, N$ [22]. Therefore, the N entropies $\mathbf{W}^{(j)}$, $j = 1, \cdots, N$ provide the mean of the tensor's global correlation. Furthermore, rank r_j of each $\mathbf{W}^{(j)}$ is upper bounded by

$$\min\{I_1\cdots I_j, I_{j+1}\cdots I_N\} \tag{44}$$

making the truncation (43) highly favorable in term of compression loss to matrices of higher rank due to balanced row and column numbers.
A detailed outline of our MPS approach to tensor feature extraction is presented in Algorithm 1.

B. Tensor mode pre-permutation and pre-positioning mode K for MPS

One can see from (44) that the efficiency of controlling the bond dimension Δ_j is dependent on its upper bound (44). Particularly, the efficiency of controlling the bond dimensions Δ_{n-1} and Δ_n that define the feature number (19) is dependent on

$$\min\{I_1\cdots I_{n-1}, I_n\cdots I_N\} \tag{45}$$

Algorithm I: MPS for tensor feature extraction

```
Input:   $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times K \cdots \times I_N}$,
         $\epsilon$: SVD threshold
Output:  $\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}$, $k = 1, \cdots, K$
         $\mathbf{B}_{i_j}^{(j)}$ $(i_j = 1, \ldots, I_j, j = 1, \ldots, n-1)$
         $\mathbf{C}_{i_{(j-1)}}^{(j)}$ $(i_{(j-1)} = 1, \ldots, I_{(j-1)}, j = n+1, \ldots, N+1)$
1:  Set $\mathbf{W}^{(1)} = \mathbf{X}_{(1)}$          % Mode-1 matricization of $\boldsymbol{\mathcal{X}}$
2:  for $j = 1$ to $n-1$                % Left-to-right sweep
3:      $\mathbf{W}^{(j)} = \mathbf{USV}$              % SVD of $\mathbf{W}^{(j)}$
4:      $\mathbf{W}^{j} \approx \mathbf{U}^{(j)}\mathbf{W}^{(j+1)}$      % Thresholding $\mathbf{S}$ for QR-approximation
5:      Reshape $\mathbf{U}^{(j)}$ to $\boldsymbol{\mathcal{U}}$
6:      $\mathbf{B}_{i_j}^{(j)} = \boldsymbol{\mathcal{U}}(:, i_j, :)$        % Set common factors
7:  end
8:  Reshape $\mathbf{V}^{(n-1)}$ to $\mathbf{W}^{N} \in \mathbb{R}^{(\Delta_{n-1} K \cdots I_N) \times I_N}$
9:  for $j = N$ down to $n$             % right-to-left sweep
10:     $\mathbf{W}^{(j)} = \mathbf{USV}$              % SVD of $\mathbf{W}^{(j)}$
11:     $\mathbf{W}^{(j)} \approx \mathbf{W}^{(j-1)}\mathbf{V}^{(j)}$     % Thresholding $\mathbf{S}$ for RQ-approximation
13:     Reshape $\mathbf{V}^{(j)}$ to $\boldsymbol{\mathcal{V}}$
14:     $\mathbf{C}_{i_{j-1}}^{(j+1)} = \boldsymbol{\mathcal{V}}(:, i_{j-1}, :)$      % Set common factors
15: end
16: Reshape $\mathbf{U}^{(n)}$ into $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{\Delta_{n-1} \times K \times \Delta_n}$
17: Set $\mathbf{G}_k^{(n)} = \boldsymbol{\mathcal{G}}(:, k, :)$          % Training core matrix
```

Texts after symbol "%" are comments.

Therefore, it is important to pre-permute the tensors modes such that the ratio

$$\frac{\min\{\prod_{i=1}^{n-1} I_i, \prod_{i=n}^{N} I_i\}}{\max\{\prod_{i=1}^{n-1} I_i, \prod_{i=n}^{N} I_i\}} \tag{46}$$

is near to 1 as possible, while $\{I_1, \cdots, I_{n-1}\}$ is in decreasing order

$$I_1 \geq \cdots \geq I_{n-1} \tag{47}$$

and $\{I_n, \cdots, I_N\}$ in increasing order

$$I_n \leq \cdots \leq I_N \tag{48}$$

to improve the ratio

$$\frac{\min\{\prod_{i=1}^{j} I_j, \prod_{i=j+1}^{N} I_i\}}{\max\{\prod_{i=1}^{j} I_j, \prod_{i=j+1}^{N} I_i\}} \tag{49}$$

for balancing $\mathbf{W}^{(j)}$.
The mode K is then pre-positioned in n-th mode as in (14).

C. Complexity analysis

In the following complexity analysis it is assumed $I_n = I$ $\forall n$ for simplicity. The dominant computational complexity of MPS is $\mathcal{O}(KI^{(N+1)})$ due to the first SVD of the matrix obtained from the mode-1 matricization of $\boldsymbol{\mathcal{X}}$. On the other hand, the computational complexity of HOOI requires several iterations of an ALS method to obtain convergence. In addition, it usually employs the HOSVD to initialize the tensors which involves the cost of order $\mathcal{O}(NKI^{N+1})$, and thus very expensive with large N compared to MPS.

MPCA is computationally upper bounded by $\mathcal{O}(NKI^{N+1})$, however, unlike HOOI, MPCA doesn't require the formation of the $(N+1)$th order core tensor at every iteration and convergence can usually happen in one iteration [12].[2]

[2]This does not mean that MPCA is computationally efficient but in contrast this means that alternating iterations of MPCA prematurely terminate, yielding a solution that is far from the optimal one of a NP-hard problem.

The computational complexity of R-UMLDA is approximately $\mathcal{O}(K\sum_{n=2}^{N} I^n + (C+K)I^2 + (p-1)[IK + 2I^2 + (p-1)^2 + (2I(p-1)] + 4I^3)$, where C is the number of classes, p is the number of projections, which determines the core vector size [19]. Therefore, R-UMLDA would perform poorly for many samples and classes.

D. MPS-based tensor object classification

This subsection presents two methods for tensor objection classification based on Algorithm 1. For each method, an explanation of how to reduce the dimensionality of tensors to core matrices, and subsequently to feature vectors for application to linear classifiers is given.

1) Principal component analysis via tensor-train (TTPCA): The TTPCA algorithm is an approach where Algorithm 1 is applied directly on the training set, with no preprocessing such as data centering. Specifically, given a set of Nth-order tensor samples $\mathcal{X}^{(k)} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, then the core matrices are obtained as

$$\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}. \tag{50}$$

Vectorizing each k sample results in

$$\mathbf{g}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1}\Delta_n}. \tag{51}$$

Using (43), $\Delta_{n-1}\Delta_n$ features of k is significantly less in comparison to $N_f = \prod_{n=1}^{N} I_n$ of $\mathcal{X}^{(k)}$, which allows for PCA to be easily applied, followed by a linear classifier.

2) MPS: The second algorithm is simply called MPS, where in this case we first perform data centering on the set of training samples $\{\boldsymbol{\mathcal{X}}^{(k)}\}$, then apply Algorithm 1 to obtain the core matrices

$$\mathbf{G}_k^{(n)} \in \mathbb{R}^{\Delta_{n-1} \times \Delta_n}. \tag{52}$$

Vectorizing the K samples results in (51), and subsequent linear classifiers such as LDA or nearest neighbors can be utilized. In this method, MPS can be considered a multidimensional analogue to PCA because the tensor samples have been data centered and are projected to a new orthogonal space using Algorithm 1, resulting in the core matrices.

IV. Experimental results

In this section, we conduct experiments on the proposed TTPCA and MPS algorithms for tensor object classification. An extensive comparison is conducted based on CSR and training time with tensor-based methods MPCA, HOOI, and R-UMLDA.

Four datasets are utilized for the experiment. The Columbia Object Image Libraries (COIL-100) [24], [25], Extended Yale Face Database B (EYFB) [26], BCI Jiaotong dataset (BCI) [27], and the University of South Florida HumanID "gait challenge" dataset (GAIT) version 1.7 [28] . All simulations are conducted in a Matlab environment.

A. Parameter selection

TTPCA, MPS and HOOI rely on the threshold ϵ defined in (43) to reduce the dimensionality of a tensor, while keeping its most relevant features. To demonstrate how the classification success rate (CSR) varies, we utilize different ϵ for each dataset. It is trivial to see that a larger ϵ would result in a longer training time due to its computational complexity, which was discussed in subsection III-C. Furthermore, TTPCA utilizes PCA, and a range of principal components p is used for the experiments. MPS and HOOI also include a truncation parameter $\tilde{\Delta}_l$ (l is the lth mode of a tensor), and HOOI is implemented with a maximum of 10 ALS iterations. MPCA relies on fixing an initial quality factor Q, which is determined through numerical simulations, and a specified number of elementary multilinear projections (EMP), we denote as m_p, must be initialized prior to using the R-UMLDA algorithm. A range of EMP's is determined through numerical simulations and the regularization parameter is fixed to $\gamma = 10^{-6}$.

Results are reported by mean and standard deviation for the algorithms with only a single parameter (MPCA, R-UMLDA and PCA). For algorithms that include two parameters (MPS, TTPCA and HOOI), for each ϵ (first parameter), the mean and standard deviation is taken on the second parameter ($\tilde{\Delta}_l$ for MPS and HOOI, p for TTPCA).

B. Tensor object classification

1) COIL-100: For this dataset we strictly compare MPS and the HOSVD-based algorithm HOOI to analyse how adjusting ϵ affects the approximation of the original tensors, as well as the reliability of the extracted features for classification. The COIL-100 dataset has 7200 color images of 100 objects (72 images per object) with different reflectance and complex geometric characteristics. Each image is initially a 3rd-order tensor of dimension $128 \times 128 \times 3$ and then is downsampled to the one of dimension $32 \times 32 \times 3$. The dataset is divided into training and test sets randomly consisting of K and L images, respectively according to a certain holdout (H/O) ratio r, i.e. $r = \frac{L}{L+K}$. Hence, the training and test sets are represented by four-order tensors of dimensions $32 \times 32 \times 3 \times K$ and $32 \times 32 \times 3 \times L$, respectively. In Fig. 3 we show how a few objects of the training set ($r = 0.5$ is chosen) change after compression by MPS and HOOI with two different values of threshold, $\epsilon = 0.9, 0.65$. We can see that with $\epsilon = 0.9$, the images are not modified significantly due to the fact that many features are preserved. However, in the case that $\epsilon = 0.65$, the images are blurred. That is because fewer features are kept. However, we can observe that the shapes of objects are still preserved. Especially, in most cases MPS seems to preserve the color of the images better than HOOI. This is because the bond dimension corresponding to the color mode $I_3 = 3$ has a small value, e.g. $\Delta_3 = 1$ for $\epsilon = 0.65$ in HOOI. This problem arises due to the the unbalanced matricization of the tensor corresponding to the color mode. Specifically, if we take a mode-3 matricization of tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{32\times32\times3\times K}$, the resulting matrix of size $3 \times (1024K)$ is extremely unbalanced. Therefore, when taking SVD with some small threshold ϵ, the information corresponding to this color mode may be lost due to dimension reduction. On the contrary, we can efficiently avoid this problem in MPS by permuting the tensor such that $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{32\times K\times3\times32}$ before applying the tensor decomposition.

The peak signal-to-noise ratio (PSNR) is calculated for each of the ten images for $\epsilon = 0.9, 0.65$ in Fig. 3 and illustrated in Fig. 4. For $\epsilon = 0.9$, MPS generally has a higher PSNR, however for $\epsilon = 0.65$, HOOI has a generally higher PSNR. Although visually, HOOI has lost color compared to MPS, it retains the structural properties of the images better than MPS for this case. This can be due to the Tucker core tensor being of the same order as the original tensor.

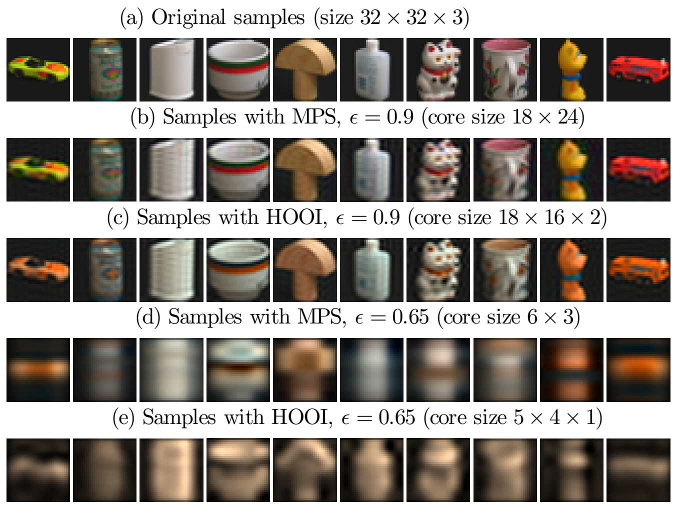

(a) Original samples (size $32 \times 32 \times 3$)
(b) Samples with MPS, $\epsilon = 0.9$ (core size 18×24)
(c) Samples with HOOI, $\epsilon = 0.9$ (core size $18 \times 16 \times 2$)
(d) Samples with MPS, $\epsilon = 0.65$ (core size 6×3)
(e) Samples with HOOI, $\epsilon = 0.65$ (core size $5 \times 4 \times 1$)

Fig. 3: Modification of ten objects in the training set of COIL-100 are shown after applying MPS and HOOI corresponding to $\epsilon = 0.9$ and 0.65 to compress tensor objects.

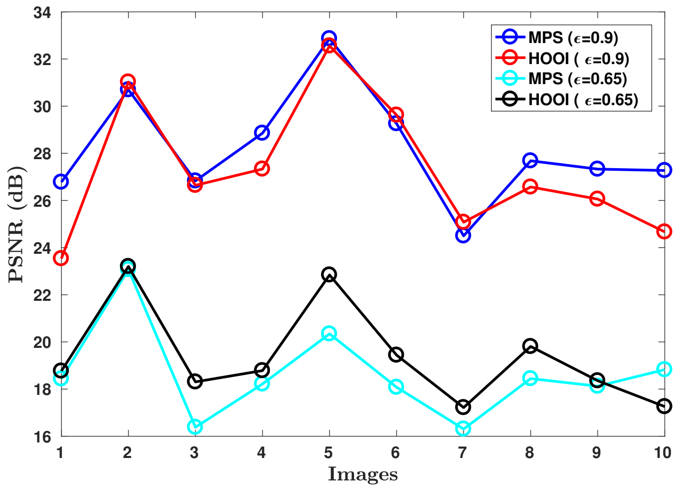


Fig. 4: PSNR of the ten images from Fig. 3.

K nearest neighbors with K=1 (KNN-1) is used for classification. For each H/O ratio, the CSR is averaged over 10 iterations of randomly splitting the dataset into training and test sets. Comparison of performance between MPS and HOOI is shown in Fig. 5 for four different H/O ratios, i.e. $r = (50\%, 80\%, 90\%, 95\%)$. In each plot, we show the CSR with respect to threshold ϵ. We can see that MPS performs

TABLE I: COIL-100 classification results. The best CSR corresponding to different H/O ratios obtained by MPS and HOOI.

Algorithm	CSR	N_f	ϵ	CSR	N_f	ϵ
$r = 50\%$				$r = 80\%$		
HOOI	98.87 ± 0.19	198	0.80	94.13 ± 0.42	112	0.75
MPS	$\mathbf{99.19 \pm 0.19}$	120	0.80	$\mathbf{95.37 \pm 0.31}$	18	0.75
$r = 90\%$				$r = 95\%$		
HOOI	87.22 ± 0.56	112	0.75	77.76 ± 0.90	112	0.65
MPS	$\mathbf{89.38} \pm \mathbf{0.40}$	59 ± 5	0.75	$\mathbf{83.17 \pm 1.07}$	18	0.65

quite well when compared to HOOI. Especially, with small ϵ, MPS performs much better than HOOI. Besides, we also show the best CSR corresponding to each H/O ratio obtained by different methods in Table I. It can be seen that MPS always gives better results than HOOI even in the case of small value of ϵ and number of features N_f defined by (10) and (19) for HOOI and MPS, respectively.

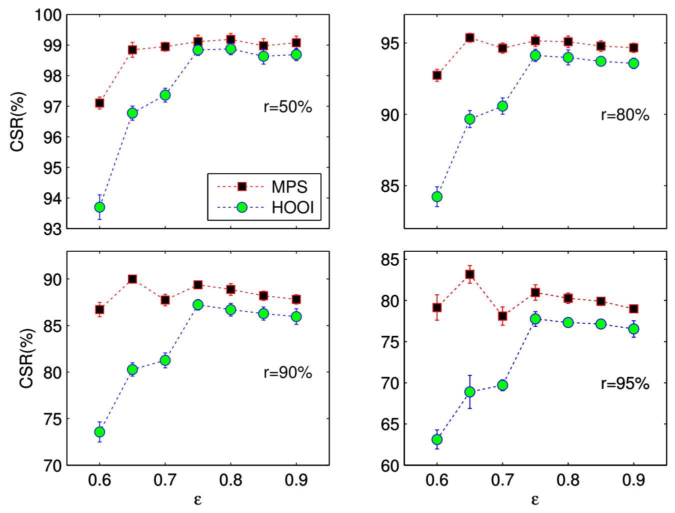


Fig. 5: Error bar plots of CSR versus thresholding rate ϵ for different H/O ratios.

2) *Extended Yale Face Database B:* The EYFB dataset contains 16128 grayscale images with 28 human subjects, under 9 poses, where for each pose there is 64 illumination conditions. Similar to [29], to improve computational time each image was cropped to keep only the center area containing the face, then resized to 73 x 55. Some examples are given in Fig. 6.

The training and test datasets are not selected randomly but partitioned according to poses. More precisely, the training and test datasets are selected to contain poses 0, 2, 4, 6 and 8 and 1, 3, 5, and 7, respectively. For a single subject the training tensor has size $5 \times 73 \times 55 \times 64$ and $4 \times 73 \times 55 \times 64$ is the size of the test tensor. Hence for all 28 subjects we have fourth-order tensors of sizes $140 \times 73 \times 55 \times 64$ and $112 \times 73 \times 55 \times 64$ for the training and test datasets, respectively.

Fig. 6: Examples of the resultant 73 x 55 images of EYFB.

In this experiment, the core tensors remains very large even with a small threshold used, e.g., for $\epsilon = 0.75$, the core size of each sample obtained by TTPCA/MPS and HOOI are $18 \times 201 = 3618$ and $14 \times 15 \times 13 = 2730$, respectively, because of slowly decaying singular values, which make them too large for classification. Therefore, we need to further reduce the sizes of core tensors before feeding them to classifiers for a better performance. In our experiment, we simply apply a further truncation to each core tensor by keeping the first few dimensions of each mode of the tensor. Intuitively, this can be done as we have already known that the space of each mode is orthogonal and ordered in such a way that the first dimension corresponds to the largest singular value, the second one corresponds to the second largest singular value and so on. Subsequently, we can independently truncate the dimension of each mode to a reasonably small value (which can be determined empirically) without changing significantly the meaning of the core tensors. It then gives rise to core tensors of smaller size that can be used directly for classification. More specifically, suppose that the core tensors obtained by MPS and HOOI have sizes $Q \times \Delta_1 \times \Delta_2$ and $Q \times \Delta_1 \times \Delta_2 \times \Delta_3$, where Q is the number K (L) of training (test) samples, respectively. The core tensors are then truncated to be $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2$ and $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2 \times \tilde{\Delta}_3$, respectively such that $\tilde{\Delta}_l < \Delta_l$ ($l = 1, 2, 3$). Note that each $\tilde{\Delta}_l$ is chosen to be the same for both training and test core tensors. In regards to TTPCA, each core matrix is vectorized to have $\Delta_1 \Delta_2$ features.

Classification results for different threshold values ϵ is shown in Table II for TTPCA, MPS and HOOI using two different classifiers, i.e. KNN-1 and LDA. Results from MPCA and R-UMLDA is also included. The core tensors obtained by MPS and HOOI are reduced to have sizes of $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2$ and $Q \times \tilde{\Delta}_1 \times \tilde{\Delta}_2 \times \tilde{\Delta}_3$, respectively such that $\tilde{\Delta}_1 = \tilde{\Delta}_2 = \Delta \in (10, 11, 12, 13, 14)$ and $\tilde{\Delta}_3 = \min(1, \Delta_3)$. Therefore, the reduced core tensors obtained by both methods have the same size for classification. With MPS and HOOI, each value of CSR in Table II is computed by taking the average of the ones obtained from classifying different reduced core tensors due to different Δ. In regards to TTPCA, for each ϵ, a range of principal components $p = \{50, \ldots, 70\}$ is used. We utilize $Q = \{70, 75, 80, 85, 90\}$ for MPCA, and the range $m_p = \{10, \ldots, 20\}$ for R-UMLDA. PCA is also included for the range $p = \{20, 30, 50, 80, 100\}$ principal components, where we vectorize each train or test tensor sample. The

TABLE II: EYFB classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
KNN-1				
HOOI	90.71 ± 1.49	90.89 ± 1.60	91.61 ± 1.26	88.57 ± 0.80
MPS	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$	$\mathbf{94.29 \pm 0.49}$
TTPCA	86.05 ± 0.44	86.01 ± 0.86	87.33 ± 0.46	86.99 ± 0.53
MPCA	90.89 ± 1.32			
R-UMLDA	71.34 ± 2.86			
PCA	32.50 ± 1.02			
LDA				
HOOI	96.07 ± 0.80	95.89 ± 0.49	96.07 ± 0.49	96.07 ± 0.49
MPS	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$	$\mathbf{97.32 \pm 0.89}$
TTPCA	95.15 ± 0.45	95.15 ± 0.45	95.15 ± 0.45	94.86 ± 0.74
MPCA	90.00 ± 2.92			
R-UMLDA	73.38 ± 1.78			
PCA	31.07 ± 2.04			

TABLE III: BCI Jiaotong classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
Subject 1				
HOOI	84.39 ± 1.12	83.37 ± 0.99	82.04 ± 1.05	84.80 ± 2.21
MPS	$\mathbf{87.24 \pm 1.20}$	$\mathbf{87.55 \pm 1.48}$	$\mathbf{87.24 \pm 1.39}$	$\mathbf{87.65 \pm 1.58}$
TTPCA	78.57 ± 3.95	78.43 ± 3.73	79.43 ± 4.12	79.14 ± 2.78
MPCA	82.14 ± 3.50			
R-UMLDA	63.18 ± 0.37			
CSP	80.14 ± 3.73			
PCA	50.85 ± 1.28			
Subject 2				
HOOI	83.16 ± 1.74	82.35 ± 1.92	82.55 ± 1.93	79.39 ± 1.62
MPS	$\mathbf{90.10 \pm 1.12}$	$\mathbf{90.10 \pm 1.12}$	$\mathbf{90.00 \pm 1.09}$	$\mathbf{91.02 \pm 0.70}$
TTPCA	80.57 ± 0.93	81.14 ± 1.86	81.29 ± 1.78	80 ± 2.20
MPCA	81.29 ± 0.78			
R-UMLDA	70.06 ± 0.39			
CSP	81.71 ± 8.96			
PCA	61.29 ± 0.93			
Subject 3				
HOOI	60.92 ± 1.83	61.84 ± 1.97	61.12 ± 1.84	60.51 ± 1.47
MPS	61.12 ± 1.36	61.22 ± 1.53	61.12 ± 1.54	60.71 ± 1.54
TTPCA	67.43 ± 2.56	68.29 ± 2.56	67.71 ± 2.28	66.43 ± 2.02
MPCA	56.14 ± 2.40			
R-UMLDA	57.86 ± 0.00			
CSP	$\mathbf{77.14 \pm 2.26}$			
PCA	52.43 ± 2.51			
Subject 4				
HOOI	48.27 ± 1.54	47.55 ± 1.36	49.98 ± 1.29	47.96 ± 1.27
MPS	52.35 ± 2.82	52.55 ± 3.40	52.55 ± 3.69	51.84 ± 3.11
TTPCA	50.29 ± 2.97	49.71 ± 3.77	49.14 ± 3.48	52.00 ± 3.48
MPCA	51.00 ± 3.96			
R-UMLDA	46.36 ± 0.93			
CSP	$\mathbf{59.86 \pm 1.98}$			
PCA	52.00 ± 3.62			
Subject 5				
HOOI	$\mathbf{60.31 \pm 1.08}$	$\mathbf{60.82 \pm 0.96}$	$\mathbf{59.90 \pm 2.20}$	$\mathbf{60.41 \pm 1.36}$
MPS	59.39 ± 2.08	59.18 ± 2.20	58.57 ± 1.60	59.29 ± 1.17
TTPCA	53.43 ± 2.79	54.29 ± 3.19	53.86 ± 3.83	54.86 ± 2.49
MPCA	50.43 ± 1.48			
R-UMLDA	55.00 ± 0.55			
CSP	59.14 ± 2.11			
PCA	53.57 ± 2.08			

average CSR's are computed with TTPCA, MCPA, PCA and R-UMLDA according to their respective range of parameters in Table II. We can see that the MPS gives rise to better results for all threshold values using different classifiers. More importantly, MPS with the smallest ϵ can produce the highest CSR. The CSR is identical for each ϵ in this case because the features in the range $\Delta \in (10, 11, 12, 13, 14)$ show no variation. Incorporating more features, e.g. $\Delta \in (10, \ldots, 30)$, will more likely lead to variation in average CSR and standard deviation for each ϵ. The LDA classifier gives rise to the best result, i.e. $\mathbf{97.32 \pm 0.89}$.

3) BCI Jiaotong: The BCIJ dataset consists of single trial recognition for BCI electroencephalogram (EEG) data involving left/right motor imagery (MI) movements. The dataset includes five subjects and the paradigm required subjects to control a cursor by imagining the movements of their right or left hand for 2 seconds with a 4 second break between trials. Subjects were required to sit and relax on a chair, looking at a computer monitor approximately 1m from the subject at eye level. For each subject, data was collected over

TABLE IV: GAIT classification results

Algorithm	CSR ($\epsilon = 0.9$)	CSR ($\epsilon = 0.85$)	CSR ($\epsilon = 0.80$)	CSR ($\epsilon = 0.75$)
Probe A				
HOOI	63.71 ± 3.36	63.90 ± 3.40	64.16 ± 3.39	64.33 ± 3.20
MPS	70.03 ± 0.42	70.03 ± 0.38	70.01 ± 0.36	69.99 ± 0.38
TTPCA	$\mathbf{75.31 \pm 0.29}$	$\mathbf{76.03 \pm 0.38}$	$\mathbf{76.38 \pm 0.78}$	$\mathbf{77.75 \pm 0.92}$
MPCA	55.77 ± 1.08			
R-UMLDA	46.62 ± 2.13			
PCA	2.89 ± 0.52			
Probe C				
HOOI	36.67 ± 2.84	36.73 ± 2.79	36.70 ± 3.07	36.87 ± 3.68
MPS	$\mathbf{41.46 \pm 0.64}$	$\mathbf{41.36 \pm 0.64}$	$\mathbf{41.29 \pm 0.63}$	41.46 ± 0.59
TTPCA	39.17 ± 0.90	40.83 ± 0.41	41.61 ± 1.02	$\mathbf{44.40 \pm 1.54}$
MPCA	29.35 ± 2.29			
R-UMLDA	20.87 ± 0.76			
PCA	1.10 ± 0.32			
Probe D				
HOOI	19.73 ± 0.91	19.96 ± 1.15	20.32 ± 0.93	20.29 ± 1.11
MPS	$\mathbf{23.82 \pm 0.42}$	$\mathbf{23.84 \pm 0.43}$	$\mathbf{23.84 \pm 0.45}$	$\mathbf{23.84 \pm 0.40}$
TTPCA	21.92 ± 0.54	22.14 ± 0.20	22.84 ± 0.42	21.92 ± 0.59
MPCA	21.11 ± 3.43			
R-UMLDA	7.88 ± 1.00			
PCA	2.11 ± 0.99			
Probe F				
HOOI	$\mathbf{20.77 \pm 0.92}$	$\mathbf{20.71 \pm 0.72}$	20.15 ± 0.65	19.96 ± 0.67
MPS	20.50 ± 0.40	20.52 ± 0.34	$\mathbf{20.50 \pm 0.29}$	$\mathbf{20.56 \pm 0.46}$
TTPCA	14.78 ± 0.60	14.74 ± 0.77	15.29 ± 0.75	15.40 ± 0.55
MPCA	17.12 ± 2.79			
R-UMLDA	9.67 ± 0.58			
PCA	1.40 ± 0.17			

TABLE V: Seven experiments in the USF GAIT dataset

Probe set	A(GAL)	B(GBR)	C(GBL)	D(CAR)	E(CBR)	F(CAL)	G(CBL)
Size	71	41	41	70	44	70	44
Differences	View	Shoe	Shoe, view	Surface	Surface, shoe	Surface, view	Surface, view, shoe

two sessions with a 15 minute break in between. The first session contained 60 trials (30 trials for left, 30 trials for right) and were used for training. The second session consisted of 140 trials (70 trials for left, 70 trials for right). The EEG signals were sampled at 500Hz and preprocessed with a filter at 8-30Hz, hence for each subject the data consisted of a multidimensional tensor $channel \times time \times Q$. The common spatial patterns (CSP) algorithm [30] is a popular method for BCI classification that works directly on this tensor, and provides a baseline for the proposed and existing tensor-based methods. For the tensor-based methods, we preprocess the data by transforming the tensor into the time-frequency domain using complex Mortlet wavelets with bandwidth parameter $f_b = 6$Hz (CMOR6-1) to make classification easier [31], [32]. The wavelet center frequency $f_c = 1$Hz is chosen. Hence, the size of the concatenated tensors are 62 $channels \times$ 23 $frequency\ bins \times 50\ time\ frames \times Q$.

We perform the experiment for all subjects. After applying the feature extraction methods MPS and HOOI, the core tensors still have high dimension, so we need to further reduce their sizes before using them for classification. For instance, the reduced core sizes of MPS and HOOI are chosen to be $Q \times 12 \times \Delta$ and $Q \times 12 \times \Delta \times \tilde{\Delta}_3$, where $\Delta \in (8, \ldots, 14)$, respectively. With TTPCA, the principal components $p = \{10, 50, 100, 150, 200\}$, $Q = \{70, 75, 80, 85, 90\}$ for MPCA, $m_p = \{10, \ldots, 20\}$ for R-UMLDA and $p = \{10, 20, 30, 40, 50\}$ with PCA. With CSP, we average CSR for a range of spatial components $s_c = \{2, 4, 6, 8, 10\}$.

The LDA classifier is utilized and the results are shown in Table III for different threshold values of TTPCA, MPS and HOOI. The results of PCA, MPCA, R-UMLDA and CSP are also included. MPS outperforms the other methods for Subjects 1 and 2, and is comparable to HOOI in the results for Subject 5. CSP has the highest CSR for Subjects 3 and 4, followed by MPS or TTPCA, which demonstrates the proposed methods being effective at reducing tensors to relevant features, more precisely than current tensor-based methods.

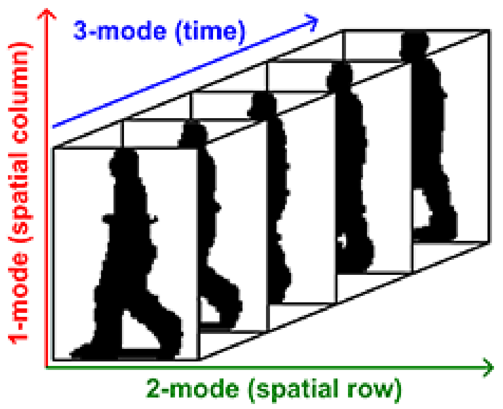
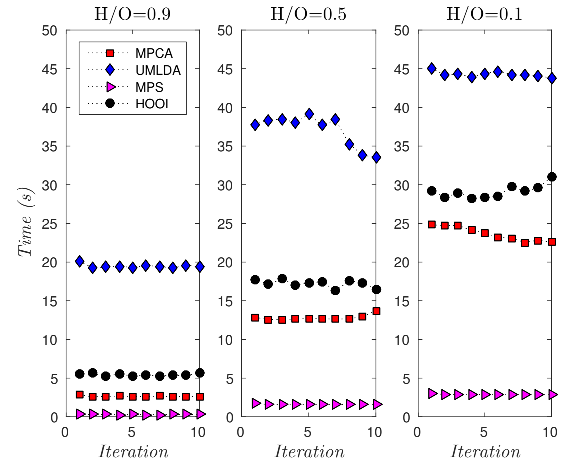
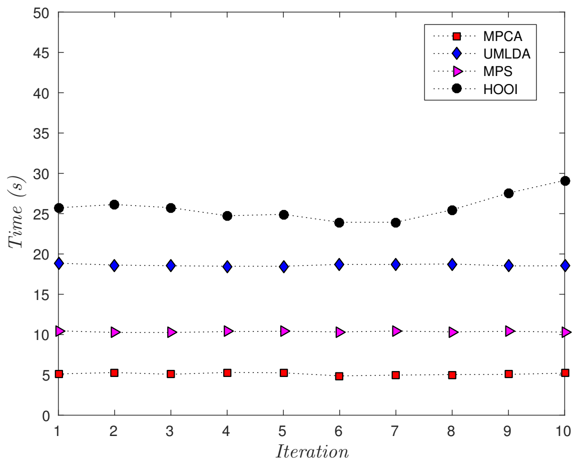


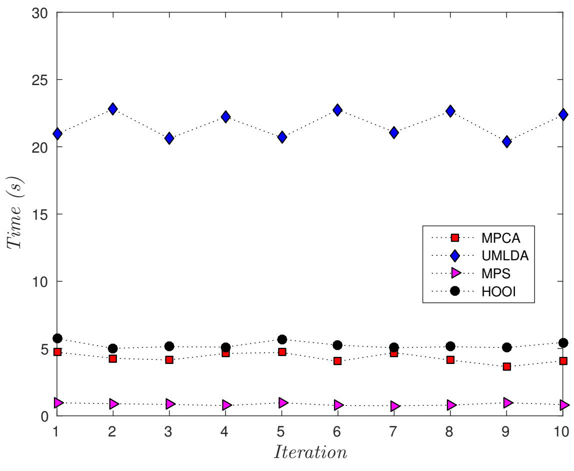
Fig. 7: The gait silhouette sequence for a third-order tensor.



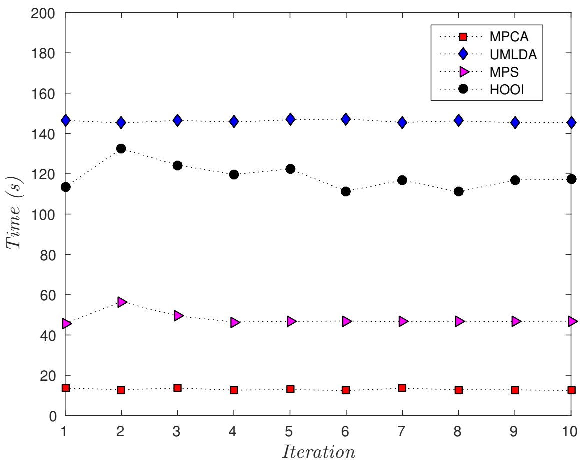
(a) COIL-100.



(b) EYFB.



(c) BCI Subject 1.



(d) GAIT Probe A.

Fig. 8: Training time of datasets for MPS, MPCA, HOOI and R-UMLDA.

4) USF GAIT challenge: The USFG database consists of 452 sequences from 74 subjects who walk in elliptical paths in front of a camera. There are three conditions for each subject: shoe type (two types), viewpoint (left or right), and the surface type (grass or concrete). A gallery set (training set) contains 71 subjects and there are seven types of experiments known as probe sets (test sets) that are designed for human identification. The capturing conditions for the probe sets is summarized in Table V, where G, C, A, B, L and R stand for grass surface, cement surface, shoe type A, shoe type B, left view and right view, respectively. The conditions in which the gallery set was captured is grass surface, shoe type A and right view (GAR). The subjects in the probe and gallery sets are unique and there are no common sequences between the gallery and probe sets. Each sequence is of size 128×88 and the time mode is 20, hence each gait sample is a third-order tensor of size $128 \times 88 \times 20$, as shown in Fig. 7. The gallery set contains 731 samples, therefore the training tensor is of size $128 \times 88 \times 20 \times 731$. The test set is of size $128 \times 88 \times 20 \times P_s$, where P_s is the sample size for the probe set that is used for a benchmark, refer to Table V. The difficulty of the classification task increases with the amount and and type of variables, e.g. Probe A only has the viewpoint, whereas Probe F has surface and viewpoint, which is more difficult. For the experiment we perform tensor object classification with Probes A, C, D and F (test sets).

The classification results of individual gait samples are based on using the LDA classifier, which is shown in Table IV. The threshold ϵ still retains many features in the core tensors of MPS and HOOI. Therefore, further reduction of the core tensors is chosen to be $Q \times 20 \times \Delta$ and $Q \times 20 \times \Delta \times \tilde{\Delta}_3$, where $\Delta \in (8, \ldots, 14)$, respectively. The principal components for TTPCA is the range $p = \{150, 200, 250, 300\}$, $Q = \{70, 75, 80, 85, 90\}$ for MPCA, $m_p = \{10, \ldots, 20\}$ for R-UMLDA and $p = \{5, 10, 20, 30, 50\}$ with PCA. The proposed algorithms achieve the highest performance for Probes A, C, and D. MPS and HOOI are similar for the most difficult test set Probe F. PCA performed very poorly, which is mostly due to its inability to capture the $128 \times 88 \times 20$ tensor gait samples correlations because they have been vectorized.

It is important to highlight the different measure of accuracy used in [12] for its gait results with MPCA, which uses cumulative match characteristic (CMC). CSR is equivalent to the measure of accuracy known as character recognition rate (CRR), which is used throughout in more recent paper on R-

UMLDA [19] by the same authors.

The purpose of Section IV is to demonstrate how the proposed methods work fundamentally against other similar methods for tensor feature extraction. The results in terms of CSR can possibly be improved using cross validation, state-of-the-art classifiers (e.g. deep learning [33]) and/or ensembling [34], and is subject to future work.

C. Training time benchmark

An additional experiment on training time for MPS[3], HOOI, MPCA and R-UMLDA is provided to understand the computational complexity of the algorithms. The classification experiments in Section IV-B involved iterations on different parameters for each algorithm (e.g. Q for MPCA, p for TTPCA, m_p for R-UMLDA) for the purpose of obtaining the highest average CSR. Therefore, since each algorithm performs feature extraction quite differently, we report on the training time, rather than the test time, which would generally be quick after dimensionality reduction and application of KNN or LDA. For the COIL-100 dataset, we measure the elapsed training time for the training tensor of size $32 \times 32 \times 3 \times K$ ($K = 720, 3600, 6480$) for H/O= $\{0.9, 0.5, 0.1\}$, according to 10 random partitions of train/test data (*iterations*). MPCA, HOOI and R-UMLDA reduces the tensor to 32 features, and MPS to 36 (due to a fixed dimension Δ^2). In Fig. 8a, we can see that as the number of training images increases, the MPS algorithms computational time only slightly increases, while MCPA and HOOI increases gradually, with UMLDA having the slowest performance overall.

The EYFB benchmark reduces the training tensor features to 36 (for MPS), 32 (MPCA and HOOI), and 16 (UMLDA, since the elapsed time for 32 features is too long). For this case, Fig. 8b demonstrates that MPCA provides the fastest computation time due to its advantage with small sample sizes (SSS). MPS performs the next best, followed by HOOI, then UMLDA with the slowest performance.

The BCI experiment involves reducing the training tensor to 36 (MPS) or 32 (MPS, HOOI and UMLDA) features and the elapsed time is shown for Subject 1 in Fig. 8c. For this case MPS performs the quickest compared to the other algorithms, with UMLDA again performing the slowest.

Lastly, the USFG benchmark tests Probe A by reducing the MPS training tensor to 36 features, MPCA and HOOI to 32 features, and UMLDA to 16 features. Fig. 8d shows that MPCA provides the quickest time to extract the features, followed by MPS, HOOI and lastly UMLDA.

V. Conclusion

In this paper, a rigorous analysis of MPS and Tucker decomposition proves the efficiency of MPS in terms of retaining relevant correlations and features, which can be used directly for tensor object classification. Subsequently, two new approaches to tensor dimensionality reduction based on compressing tensors to matrices are proposed. One method reduces a tensor to a matrix, which then utilizes PCA. And the other is a new multidimensional analogue to PCA known as MPS. Furthermore, a comprehensive discussion on the practical implementation of the MPS-based approach is provided, which emphasizes tensor mode permutation, tensor bond dimension control, and core matrix positioning. Numerical simulations demonstrates the efficiency of the MPS-based algorithms against other popular tensor algorithms for dimensionality reduction and tensor object classification.

For the future outlook, we plan to explore this approach to many other problems in multilinear data compression and tensor super-resolution.

[3]TTPCA would be equivalent in this experiment.

References

[1] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "A survey of multilinear subspace learning for tensor data," *Pattern Recognition*, vol. 44, no. 7, pp. 1540 – 1551, 2011.

[2] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.

[3] L. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[4] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM Journal on Matrix Analysis and Applications*, vol. 21, no. 4, 2000.

[5] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: Tensorfaces," *Proceedings of the 7th European Conference on Computer Vision, Lecture Notes in Comput. Sci.*, vol. 2350, pp. 447–460, 2002.

[6] B. Savas and L. Eldén, "Handwritten digit classification using higher order singular value decomposition," *Pattern Recognition*, vol. 40, no. 3, pp. 993 – 1003, 2007.

[7] A. H. Phan and A. Cichocki, "Tensor decompositions for feature extraction and classification of high dimensional datasets," *IEICE Nonlinear Theory and Its Applications*, vol. 1, no. 1, pp. 37–68, 2010.

[8] L. Kuang, F. Hao, L. Yang, M. Lin, C. Luo, and G. Min, "A tensor-based approach for big data representation and dimensionality reduction," *IEEE Trans. Emerging Topics in Computing*, vol. 2, no. 3, pp. 280–291, Sept 2014.

[9] A. Cichocki, D. Mandic, L. De Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, and A. H. Phan, "Tensor decompositions for signal processing applications: From two-way to multiway component analysis," *IEEE Signal Processing Magazine*, vol. 32, no. 2, pp. 145–163, March 2015.

[10] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "On the best rank-1 and rank-(r1,r2,. . .,rn) approximation of higher-order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1324–1342, Mar. 2000.

[11] L. D. Lathauwer and J. Vandewalle, "Dimensionality reduction in higher-order signal processing and rank-(R1,R2,,RN) reduction in multilinear algebra," *Linear Algebra and its Applications*, vol. 391, no. 0, pp. 31 – 55, 2004.

[12] H. Lu, K. Plataniotis, and A. Venetsanopoulos, "MPCA: Multilinear principal component analysis of tensor objects," *IEEE Trans. Neural Networks*, vol. 19, no. 1, pp. 18–39, Jan 2008.

[13] F. Verstraete, D. Porras, and J. I. Cirac, "Density matrix renormalization group and periodic boundary conditions: A quantum information perspective," *Phys. Rev. Lett.*, vol. 93, no. 22, p. 227205, Nov 2004.

[14] G. Vidal, "Efficient classical simulation of slightly entangled quantum computation," *Phys. Rev. Lett.*, vol. 91, no. 14, p. 147902, Oct 2003.

[15] ——, "Efficient simulation of one-dimensional quantum many-body systems," *Phys. Rev. Lett.*, vol. 93, no. 4, p. 040502, Jul 2004.

[16] D. Pérez-García, F. Verstraete, M. Wolf, and J. Cirac, "Matrix product state representations," *Quantum Information and Computation*, vol. 7, no. 5, pp. 401–430, 2007.

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

[18] J. A. Bengua, H. N. Phien, and H. D. Tuan, "Optimal feature extraction and classification of tensors via matrix product state decomposition," in *2015 IEEE International Congress on Big Data*, June 2015, pp. 669–672.

[19] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear discriminant analysis with regularization and aggregation for tensor object recognition," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 103–123, Jan 2009.

[20] M. Ishteva, P.-A. Absil, S. van Huffel, and L. de Lathauwer, "Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme." *SIAM J. Matrix Anal. Appl.*, vol. 32, no. 1, pp. 115–135, 2011.

[21] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*. Cambridge, England: Cambridge University Press, 2000.

[22] J. A. Bengua, H. N. Phien, H. D. Tuan, and M. N. Do, "Efficient tensor completion for color image and video recovery: Low-rank tensor train," *IEEE Transactions on Image Processing*, vol. PP, no. 99, 2017.

[23] U. Schollwöck, "The density-matrix renormalization group in the age of matrix product states," *Annals of Physics*, vol. 326, no. 1, pp. 96 – 192, 2011.

[24] S. A. Nene, S. K. Nayar, and H. Murase, "Columbia object image library (coil-100)," *Technical Report CUCS-005-96*, Feb 1996.

[25] M. Pontil and A. Verri, "Support vector machines for 3d object recognition," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 20, no. 6, pp. 637–646, Jun 1998.

[26] A. Georghiades, P. Belhumeur, and D. Kriegman, "From few to many: illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun 2001.

[27] (2013) Data set for single trial 64-channels eeg classification in bci. [Online]. Available: http://bcmi.sjtu.edu.cn/resource.html

[28] S. Sarkar, P. J. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W. Bowyer, "The humanid gait challenge problem: data sets, performance, and analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 2, pp. 162–177, Feb 2005.

[29] Q. Li and D. Schonfeld, "Multilinear discriminant analysis for higher-order tensor data classification," *IEEE Trans. Patt. Anal. and Mach. Intell.*, vol. 36, no. 12, pp. 2524–2537, Dec 2014.

[30] Y. Wang, S. Gao, and X. Gao, "Common spatial pattern method for channel selelction in motor imagery based brain-computer interface," in *2005 IEEE Engineering in Medicine and Biology 27th Annual Conference*, Jan 2005, pp. 5392–5395.

[31] Q. Zhao and L. Zhang, "Temporal and spatial features of single-trial eeg for brain-computer interface," *Computational Intelligence and Neuroscience*, vol. 2007, pp. 1–14, Jun 2007.

[32] A. H. Phan, "NFEA: Tensor toolbox for feature extraction and application," Lab for Advanced Brain Signal Processing, BSI, RIKEN, Tech. Rep., 2011.

[33] Y. Chen, Z. Lin, X. Zhao, G. Wang, and Y. Gu, "Deep learning-based classification of hyperspectral data," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 6, pp. 2094–2107, June 2014.

[34] L. Rokach, "Ensemble-based classifiers," *Artificial Intelligence Review*, vol. 33, no. 1, pp. 1–39, 2010.

Johann A. Bengua received the B.E. degree in telecommunications engineering from the University of New South Wales, Kensington, Australia, in 2012 and Ph.D. degree in electrical engineering from the University of Technology Sydney, Ultimo, Australia, in 2017. He is now a data scientist for Teradata Australia and New Zealand. His current research interests include tensor networks, machine learning, image and video processing.

Phien N Ho received the B.S. degree from Hue University, Vietnam in 200e, the M.Sc. degree in physics from Hanoi Institute of Physics and Electronics, Vietnam in 2007, and the Ph.D. degree in computational physics from the University of Queensland, Australia in 2015. He has been a postdoctoral fellow at the Faculty of Engineering and Information Technology, University of Technology, Sydney, Australia from 2015 to 2016. He is now a quantitative analyst with Westpac Group, Australia. His interests include mathematical modelling and numerical simulations for multi-dimensional data systems.

Hoang Duong Tuan received the Diploma (Hons.) and Ph.D. degrees in applied mathematics from Odessa State University, Ukraine, in 1987 and 1991, respectively. He spent nine academic years in Japan as an Assistant Professor in the Department of Electronic-Mechanical Engineering, Nagoya University, from 1994 to 1999, and then as an Associate Professor in the Department of Electrical and Computer Engineering, Toyota Technological Institute, Nagoya, from 1999 to 2003. He was a Professor with the School of Electrical Engineering and Telecommunications, University of New South Wales, from 2003 to 2011. He is currently a Professor with the Faculty of Engineering and Information Technology, University of Technology Sydney. He has been involved in research with the areas of optimization, control, signal processing, wireless communication, and biomedical engineering for more than 20 years.

Minh N. Do (M'01-SM'07-F'14) received the B.Eng. degree in computer engineering from the University of Canberra, Australia, in 1997, and the Dr.Sci. degree in communication systems from the Swiss Federal Institute of Technology Lausanne (EPFL), Switzerland, in 2001. Since 2002, he has been on the faculty at the University of Illinois at Urbana-Champaign (UIUC), where he is currently a Professor in the Department of ECE, and holds joint appointments with the Co- ordinated Science Laboratory, the Beckman Institute for Advanced Science and Technology, and the Department of Bioengineering. His research interests include signal processing, computational imaging, geometric vision, and data analytics. He received a CAREER Award from the National Science Foundation in 2003, and a Young Author Best Paper Award from IEEE in 2008. He was named a Beckman Fellow at the Center for Advanced Study, UIUC, in 2006, and received of a Xerox Award for Faculty Research from the College of Engineering, UIUC, in 2007. He was a member of the IEEE Signal Processing Theory and Methods Technical Committee, Image, Video, and Multidimensional Signal Processing Technical Committee, and an Associate Editor of the IEEE Transactions on Image Processing.