
Article:
Shi, Q., Cheung, Y.-M., Zhao, Q. et al. (1 more author) (2019) Feature extraction for incomplete data via low-rank tensor decomposition with feature regularization. IEEE Transactions on Neural Networks and Learning Systems, 30 (6). pp. 1803-1817. ISSN 2162-237X

https://doi.org/10.1109/TNNLS.2018.2873655

Feature Extraction for Incomplete Data via Low-rank Tensor Decomposition with Feature Regularization

Qiquan Shi, *Student Member, IEEE,* Yiu-Ming Cheung, *Fellow, IEEE*, Qibin Zhao, *Senior Member, IEEE* and Haiping Lu, *Member, IEEE*

***Abstract*—Multi-dimensional data (i.e., tensors) with missing entries are common in practice. Extracting features from incomplete tensors is an important yet challenging problem in many fields such as machine learning, pattern recognition and computer vision. Although the missing entries can be recovered by tensor completion techniques, these completion methods focus only on missing data estimation instead of effective feature extraction. To the best of our knowledge, the problem of feature extraction from incomplete tensors has yet to be well explored in the literature. In this paper, we therefore tackle this problem within the unsupervised learning environment. Specifically, we incorporate *low-rank Tensor Decomposition* with *feature Variance Maximization* (TDVM) in a unified framework. Based on orthogonal Tucker and CP decompositions, we design two TDVM methods, TDVM-Tucker and TDVM-CP, to learn low-dimensional features viewing the core tensors of the Tucker model as features and viewing the weight vectors of the CP model as features. TDVM explores the relationship among data samples via maximizing feature variance and simultaneously estimates the missing entries via low-rank Tucker/CP approximation, leading to informative features extracted directly from observed entries. Furthermore, we generalize the proposed methods by formulating a general model that incorporates feature regularization into low-rank tensor approximation. In addition, we develop a joint optimization scheme to solve the proposed methods by integrating the alternating direction method of multipliers with the block coordinate descent method. Finally, we evaluate our methods on six real-world image and video datasets under a newly designed multi-block missing setting. The extracted features are evaluated in face recognition, object/action classification and face/gait clustering. Experimental results demonstrate the superior performance of the proposed methods compared with the state-of-the-art approaches.**

***Index Terms*—Incomplete tensor, feature extraction, orthogonal tensor decomposition, low-rank tensor completion, feature regularization, variance maximization.**

I. Introduction

Feature extraction is a fundamental and significant topic in many fields such as machine learning, pattern recognition, data mining, and computer vision. In recent decades, many methods for feature extraction have been developed, such as the classical principal component analysis (PCA) [1]. In real-world, many data such as color images, videos and 4D fMRI data are multi-dimensional, i.e., *tensors*, and have become increasingly popular and ubiquitous in many applications [2]. Tensor decomposition is a powerful computational tool for extracting valuable information from tensorial data, which can effectively perform dimensionality reduction, feature extraction, etc..

To learn features from tensorial data, many multilinear methods have been proposed based on tensor decomposition [3], [4], [5], [6]. There are two popular and fundamental decomposition models: Tucker decomposition [7], which decomposes a tensor into a core tensor multiplied by a factor matrix along each mode, and CANDECOMP/PARAFAC (CP) decomposition [8], [9], which factorizes a tensor into a weighted sum of rank-one tensors. Based on the Tucker model, for example, multilinear principal component analysis (MPCA) [3] is developed as a popular extension of PCA and can directly extract features from higher-order tensors. Furthermore, based on CP decomposition, a semi-orthogonal multilinear PCA with relaxed start (SOMPCARS)[6] improves [10] by relaxing the orthogonality constraint and initialization on factors. In addition, robust methods such as robust tensor PCA (TRPCA) [11] have been well studied for learning features from data with corruptions (e.g., noise and outliers).

In practice, some entries of tensors are often missing in the acquisition process, costly experiments, etc. [12], [13]. The reasons for missing data are numerous. For example, in social science, when data are collected in surveys, it is likely that some people refuse to answer a few questions related to personal privacy or sensitive topics, thus resulting in missing values with arbitrary patterns [14]. In industrial applications, some data, such as images, are corrupted with irregular patterns due to the insufficient resolution of a device or the dysfunction of equipment [15]. Over all, missing data are common in real-world [16]. In these scenarios, the feature learning methods mentioned above cannot work well due to missing data. How to correctly handle missing data is a fundamental yet challenging problem in many fields [17], [18], [15], which is critical to many real-world applications such as classification [12], [19], [16], image inpainting [20] and clustering [21], [22]. However, to the best of our knowledge, effectively *extracting features* from *incomplete tensors* has yet to be well explored.

There are two possible approaches to solving the problem of extracting features from incomplete tensors. One natural solution is to fill in the missing values and then view the

Qiquan Shi is with Huawei Noah's Ark Lab, Hong Kong (e-mail: qiquan.shi@gmail.com). Yiu-Ming Cheung is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong (e-mail: ymc@comp.hkbu.edu.hk). Yiu-Ming Cheung is the corresponding author. Qibin Zhao is with Tensor Learning Unit, RIKEN AIP, Japan and School of Automation, Guangdong University of Technology, China (e-mail: qibin.zhao@riken.jp). Haiping Lu is with the Department of Computer Science, the University of Sheffield, U.K. (e-mail: h.lu@sheffield.ac.uk).

recovered tensors as the extracted features. Many tensor completion techniques have been extended from matrix completion cases [23], [24], which are widely used for predicting missing data given partially observed entries and have drawn much attention in many applications such as image/video recovery [25], [26]. For example, Liu *et al.* [25] defined the Tucker-based tensor nuclear norm by combining nuclear norms of all matrices unfolded along each mode and proposed a high accuracy low-rank tensor completion algorithm (HaLRTC) for estimating missing values in tensor visual data. Jain *et al.* [27] developed an alternating minimization algorithm (TenALS) for tensors with a fixed low-rank orthogonal CP decomposition, which yields good completion results for incomplete data under certain conditions (e.g., a good CP rank [28]). Furthermore, Liu *et al.* [26] proposed a nuclear norm regularized CP decomposition method (TNCP) for tensor completion by imposing the Tucker-based tensor nuclear norm on factor matrices. Although these tensor completion methods can recover data well under typical conditions, they focus only on tensor recovery without considering the relationship among data samples for effective feature extraction. In addition, by treating the recovered data as learned features, the dimension of features cannot be reduced.

Another straightforward approach is a "two-step" strategy: applying tensor completion algorithms (e.g., HaLRTC) to estimate missing entries first and then feature extraction methods (e.g., MPCA) on the recovered tensors to learn the features, i.e., *"tensor completion methods + feature extraction methods"*. For example, LRANTD [4] employs nonnegative Tucker decomposition (NTD) for incomplete tensors by incorporating low-rank representation (LRA) with nonnegative feature extraction. LRANTD requires a tensor completion algorithm to estimate the missing entries in the preceding LRA step. This approach likely amplifies the approximation error as the missing data and the features are learned in separate stages. Besides, the reconstruction error from the tensor completion step can deteriorate the performance of feature extraction in the subsequent step. Moreover, the "two-step" strategy combining two separate methods is usually not computationally efficient.

On the other hand, a few supervised methods have been proposed for classifying low-rank missing data [12], [16], and some studies have integrated a discriminant analysis criterion into low-rank matrix/tensor completion models for feature classification [29], [30]. However, these methods require labels, which are expensive and difficult to obtain in practice, especially for incomplete data.

To solve the problem of feature extraction for incomplete tensors, we incorporate *low-rank **T**ensor **D**ecomposition* with *feature **V**ariance **M**aximization* (**TDVM**) into a unified framework. In this paper, we focus on two popular tensor decompositions for TDVM and design two methods: **TDVM-Tucker** and **TDVM-CP** based on Tucker and CP models, respectively. These two methods are essentially deployed under a general unsupervised model that incorporates low-rank Tensor Decomposition with Feature Regularization (TDFR). TDFR simultaneously estimates missing data via low-rank approximation and explores the relationship among samples via feature regularization. In other words, TDVM-Tucker and TDVM-CP specify TDFR by employing low-rank Tucker/CP decomposition for low-rank approximation and using feature variance maximization as the feature constraint. Specifically, TDVM-Tucker imposes the Tucker-based tensor nuclear norm on the core tensors of Tucker decomposition with orthonormal factor matrices (a.k.a., higher-order singular value decomposition (HOSVD) [31]) while minimizing the approximation error, and meanwhile maximizes the variance of core tensors. Here, the learned *core tensors* (analogous to the singular values of a matrix) are viewed as the *extracted features*. TDVM-CP realizes the low-rank CP approximation by minimizing the CP-based tensor nuclear norm [32] of weight vectors and the reconstruction error based on orthogonal CP decomposition, and meanwhile maximizes the variance of learned feature vectors for feature regularization. The *weight vector* of the orthogonal CP decomposition of a tensor (analogous to the vector of singular values of the SVD of a matrix) is viewed as the *feature vector*.

TDVM incorporates Tucker- and CP-based tensor nuclear norm regularization with variance maximization on features while estimating missing entries, which results in informative features extracted directly from observed entries. Moreover, TDVM-Tucker aims to learn low-dimensional tensorial features from high-dimensional incomplete tensors (i.e., tensor-to-tensor projection [10]), while TDVM-CP can extract low-dimensional vectorial features (i.e., tensor-to-vector projection [10]). The proposed methods differ from both tensor completion methods and two-step strategies as follows. 1) Tensor completion methods aim to recover the incomplete tensors only without exploring the relationship among samples for effective feature extraction. In contrast, TDVM methods focus on extracting low-dimensional features instead of estimating missing data. Moreover, TDVM utilizes a feature constraint (feature variance maximization) to capture the relationship among samples for extracting informative features; 2) unlike the "two-step" strategies, which learn the features of incomplete data via two separate stages, TDVM simultaneously estimates missing entries and learns low-dimensional features directly from the observed entries in the unified framework. Thus, TDVM can extract more informative features and reduce computational cost; 3) compared with the supervised methods, TDVM does not require label information during feature learning, which is more feasible in practice.

We employ Alternating Direction Method of Multipliers (ADMM) [33] and Block Coordinate Descent (BCD) to optimize TDVM models. After feature extraction via TDVM, we evaluate the extracted features on six image and video databases for three applications: face recognition, object/action classification and face/gait clustering. Partial work pertaining to TDVM-Tucker has been published in the conference version [34] of this paper, and the main contributions of this work are threefold:

1) We propose two unsupervised methods, TDVM-Tucker and TDVM-CP, for feature extraction of incomplete tensors. The TDVM methods explore the relationship among tensor samples via feature variance maximization while estimating missing values by low-rank approximation,

leading to informative features extracted directly from observed entries. Moreover, we discuss the generalization of TDVM by proposing the general model TDFR.

2) We develop an ADMM-BCD joint optimization scheme to solve the TDVM-CP model, in which each subproblem of TDVM-CP can be solved in a closed form although its overall objective is non-convex and non-smooth.
3) We evaluate the proposed methods on six tensor datasets with newly designed multi-block missing settings. Tensors with multi-block data missing are not only more general, as they cover the existing pixel-based and block-based missing settings, but are also more difficult and practical in real-world scenarios. More importantly, the experimental results show that the proposed methods outperform the state-of-the-art approaches with significant improvements.

The rest of the paper is organized as follows. We review related preliminaries and related works in Section II. Then, we present the proposed methods and general model in Section III. We report the empirical results in Section IV and conclude this paper in Section V.

II. Preliminaries and Related Work

A. Notations and Operations

The number of dimensions of a tensor is the *order* and each dimension is a *mode* of it. A vector is denoted by a bold lower-case letter $\mathbf{x} \in \mathbb{R}^I$ and a matrix is denoted by a bold capital letter $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$. A higher-order ($N \geq 3$) tensor is denoted by a calligraphic letter $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$. The ith entry of a vector $\mathbf{a} \in \mathbb{R}^I$ is denoted by $\mathbf{a}(i)$, and the (i,j)th entry of a matrix $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ is denoted by $\mathbf{X}(i,j)$. The $(i_1, \cdots, i_N)$th entry of an Nth-order tensor $\mathcal{X}$ is denoted by $\mathcal{X}(i_1, \cdots, i_N)$, where $i_n \in \{1, \cdots, I_n\}$ and $n \in \{1, \cdots, N\}$. The Frobenius norm of a tensor $\mathcal{X}$ is defined by $\|\mathcal{X}\|_F = \langle \mathcal{X}, \mathcal{X} \rangle^{1/2}$. $\mathbf{\Omega} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is a binary index set: $\mathbf{\Omega}(i_1, \cdots, i_N) = 1$ if $\mathcal{X}(i_1, \cdots, i_N)$ is observed, and $\mathbf{\Omega}(i_1, \cdots, i_N) = 0$ otherwise. $\mathcal{P}_{\mathbf{\Omega}}$ is the associated sampling operator which acquires only the entries indexed by $\mathbf{\Omega}$:

$$(\mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}))(i_1, \cdots, i_N) = \begin{cases} \mathcal{X}(i_1, \cdots, i_N), & \text{if}(i_1, \cdots, i_N) \in \mathbf{\Omega} \\ 0, & \text{if}(i_1, \cdots, i_N) \in \mathbf{\Omega}^c \end{cases}, \tag{1}$$

where $\mathbf{\Omega}^c$ is the complement of $\mathbf{\Omega}$.

Definition 1. Mode-n Product. *A mode-n product of $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and $\mathbf{U} \in \mathbb{R}^{I_n \times J_n}$ is denoted by $\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}^\top \in \mathbb{R}^{I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N}$, with entries given by $\mathcal{Y}_{i_1 \cdots i_{n-1} j_n i_{n+1} \cdots i_N} = \sum_{i_n} \mathcal{X}_{i_1 \cdots i_{n-1} i_n i_{n+1} \cdots i_N} \mathbf{U}_{i_n, j_n}$, and $\mathbf{Y}_{(n)} = \mathbf{U}^T \mathbf{X}_{(n)}$[10].*

Definition 2. Mode-n Unfolding. *a.k.a., matricization, is the process of reordering the elements of a tensor into matrices along each mode [2]. A mode-n unfolding matrix of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is denoted as $\mathbf{X}^{(n)} \in \mathbb{R}^{I_n \times \Pi_{n^* \neq n} I_{n^*}}$.*

B. Tucker and CP Decomposition

1) Tucker Decomposition: It represents a tensor $\mathcal{X}_m \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ as a core tensor with factor matrices [2]:

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

where $\{\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, n = 1, 2 \cdots N,$ and $R_n < I_n\}$ are factor matrices with orthonormal columns and $\mathcal{C}_m \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ is the core tensor with lower dimension. The

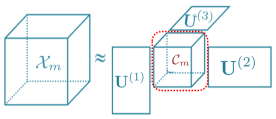


Fig. 1. The Tucker decomposition of a third-order tensor sample $\mathcal{X}_m$, where the core tensor $\mathcal{C}_m$ consists of extracted features from $\mathcal{X}_m$.

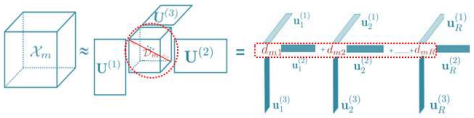


Fig. 2. The CP decomposition of a third-order tensor sample $\mathcal{X}_m$, where the core tensor $\mathcal{D}_m$ is super-diagonal and its elements $\{d_{m1}, d_{m2}, \cdots, d_{mR}\}$ (i.e. feature vector $\mathbf{d}_m$) are viewed as extracted features from $\mathcal{X}_m$.

Tucker-rank of an Nth-order tensor $\mathcal{X}$ is an N-dimensional vector, denoted as $\mathbf{r} = [R_1, \cdots, R_n, \cdots, R_N]$, whose n-th entry R_n is the rank of the mode-n unfolded matrix $\mathbf{X}_m^{(n)}$ of $\mathcal{X}_m$. Figure 1 illustrates this decomposition. In this paper, Tucker-rank is equivalent to the dimension of features (each core tensor). Based on Tucker decomposition, Liu *et al.* [25] have defined Tucker-based Tensor Nuclear Norm, that is,

Definition 3. ***Tucker-based Tensor Nuclear Norm*** *[25] of a tensor $\mathcal{X}$ is defined as: $\|\mathcal{X}\|_* = \sum_{n=1}^N \|\mathbf{X}^{(n)}\|_* = \sum_{n=1}^N \sum_{j=1}^{R_n} \sigma_j$, where $\mathbf{X}^{(n)}$ is the mode-n unfolding matrix of $\mathcal{X}$ and σ_j is the singular values of the unfolded matrix.*

2) CP Decomposition: It decomposes a tensor $\mathcal{X}_m \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ as the sum of a set of weighted rank-one tensors:

$$\begin{aligned} \mathcal{X}_m &= \sum_{r=1}^R d_{mr}\, \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \circ \cdots \circ \mathbf{u}_r^{(N)} \\ &= \mathcal{D}_m \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}, \end{aligned} \tag{3}$$

where each common factor $\{\mathbf{u}_r^{(n)}, n = 1, \cdots, N\}$ is a unit vector with a weight absorbed into the weight vector $\mathbf{d} = [d_{m1}, \cdots d_{mr}, \cdots d_{mR}]^\top \in \mathbb{R}^R$, and $\circ$ denotes the outer product [2]. Figure 2 shows that CP decomposition can also be reformulated as Tucker decomposition where the core tensor $\mathcal{D}_m$ is super-diagonal, i.e., $\mathcal{D}_m(r, \cdots, r) = d_{mr}$. R is the *CP-rank* as the minimum number of rank-one components. In this paper, CP-rank is equivalent to the dimension of features (each weight vector). Based on orthogonal CP decomposition, we have defined the CP-based Tensor Nuclear Norm:

Definition 4. ***CP-based Tensor Nuclear Norm*** *[32] of a tensor $\mathcal{X}$ is defined as the L_1 norm of the weight vector $\mathbf{d}$ of its orthogonal CP decomposition: $\|\mathcal{X}\|_{CP} = \|\mathbf{d}\|_1$.*

C. Related Work

Considering the target problem of extracting features from incomplete data, there are four categories of related approaches, which are briefly summarized as follows.

1) Tensor Completion Approach: Tensor completion approach is extended from the matrix case [23] and widely used for recovering missing data. There are many successful tensor completion methods, such as HaLRTC [25], TenALS [27], TNCP [26] and [35], [20], [36], [37]. These completion methods can yield good recovery results under typical conditions, but they focus only on estimating missing data instead of extracting informative features.

2) Feature Extraction Approach: Many tensor methods have been proposed for feature extraction directly from multilinear data, e.g., the classical MPCA [3] and [4], [5], [6],

[38], [11], [39], [40], [41], [42]. These methods can achieve state-of-the-art results for learning features from complete (and noisy) tensors, however, they cannot perform well on data with missing values.

3) Supervised Classification Approach: Some classification algorithms have been well studied for classifying low-rank missing data [12], [16]. Besides, a few studies have integrated a discriminant analysis criterion into low-rank matrix/tensor completion models for classification [29], [30]. However, these methods require labels which are expensive and difficult to obtain in practice, especially for incomplete data.

4) Subspace Clustering Approach: Subspace clustering models such as sparse subspace clustering [43] were applied in the presence of missing data in [21], [22]. In addition, some studies have incorporated matrix completion approaches with subspace clustering for incomplete matrices [44], [45]. However, these algorithms do not yield good results for learning features from incomplete tensors because they are developed for clustering incomplete vectors/matrices.

In Sec. IV, we compare the proposed unsupervised methods with related state-of-the-art algorithms selected the abovementioned category *3)* as they are supervised.

III. The Proposed: TDVM-Tucker and TDVM-CP

A. Problem Definition

Given a total M tensor samples $\{\mathcal{T}_1, \cdots, \mathcal{T}_m, \cdots, \mathcal{T}_M\}$ with *missing entries* in each sample $\mathcal{T}_m \in \mathbb{R}^{I_1 \times \cdots \times I_N}$. I_n is the mode-n dimension. We denote $\mathcal{T} = [\mathcal{T}_1, \cdots, \mathcal{T}_m, \cdots \mathcal{T}_M] \in \mathbb{R}^{I_1 \times \cdots \times I_N \times M}$, where the M are the number of tensor samples concatenated along the mode-$(N+1)$ of $\mathcal{T}$. To achieve feature extraction (dimension reduction) objective, we aim to directly extract low-dimensional features from the given high-dimensional incomplete tensors $\mathcal{T}$.

Remark 1: This problem is different from the case of data with corruptions (e.g., noise and outliers), which has been well studied in [11], [46], [47], [43]. Missing data could be equivalent to the corruption case only if the corruptions are arbitrary and the indices of corruptions are known. However, in reality, the magnitudes of corruptions are not arbitrarily large. In other words, here we study a different feature extraction problem that existing methods cannot solve well.

To solve this problem, we propose an unsupervised feature extraction approach by incorporating *low-rank **T**ensor **D**ecomposition* with *feature **V**ariance **M**aximization* (**TDVM**). Based on two widely used Tucker and CP decomposition models, we develop two algorithms of TDVM as follows.

B. TDVM-Tucker: Learning Low-dimensional Tensor Features

We first propose a TDVM method based on orthogonal Tucker decomposition: we impose the Tucker-based tensor nuclear norm on the core tensors while minimizing the reconstruction error, and meanwhile maximize the variance of core tensors (features), i.e., incorporating low-rank **T**ucker **D**ecomposition with feature **V**ariance **M**aximization, namely **TDVM-Tucker**:

$$\begin{aligned}\min_{\mathcal{X}_m, \mathcal{C}_m, \mathbf{U}^{(n)}} & \sum_{m=1}^{M} \frac{1}{2}\|\mathcal{X}_m - \mathcal{C}_m \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}\|_F^2 \\ & + \sum_{m=1}^{M} \|\mathcal{C}_m\|_* - \sum_{m=1}^{M} \frac{1}{2}\|\mathcal{C}_m - \bar{\mathcal{C}}\|_F^2, \\ \text{s.t. } & \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m), {\mathbf{U}^{(n)}}^\top \mathbf{U}^{(n)} = \mathbf{I}, n = 1 \cdots N,\end{aligned} \tag{4}$$

where $\{\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}\}_{n=1}^N$ are common factor matrices with orthonormal columns. $\mathbf{I} \in \mathbb{R}^{R_n \times R_n}$ is an identity matrix. $\mathcal{C}_m \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ is the core tensor, which consists of the *extracted features* of an incomplete tensor $\mathcal{T}_m$ with observed entries in $\mathbf{\Omega}$. $\|\mathcal{C}_m\|_*$ is the Tucker-based tensor nuclear norm of $\mathcal{C}_m$. $\bar{\mathcal{C}} = \frac{1}{M}\sum_{m=1}^{M} \mathcal{C}_m$ is the mean of core tensors (extracted features).

To optimize the objective function of TDVM-Tucker using ADMM, we apply the variable splitting technique, introduce a set of *auxiliary variables* $\{\mathcal{S}_m \in \mathbb{R}^{R_1 \times \cdots \times R_N}, m = 1 \cdots M\}$, and then reformulate Eq. (4) as follows:

$$\begin{aligned}\min_{\mathcal{X}_m, \mathcal{C}_m, \mathcal{S}_m, \mathbf{U}^{(n)}} & \sum_{m=1}^{M} \frac{1}{2}\|\mathcal{X}_m - \mathcal{C}_m \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}\|_F^2 \\ & + \sum_{m=1}^{M} \|\mathcal{S}_m\|_* - \sum_{m=1}^{M} \frac{1}{2}\|\mathcal{C}_m - \bar{\mathcal{C}}\|_F^2, \\ \text{s.t. } & \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m), \mathcal{S}_m = \mathcal{C}_m, {\mathbf{U}^{(n)}}^\top \mathbf{U}^{(n)} = \mathbf{I}.\end{aligned} \tag{5}$$

Remark 2: The objective function Eq. (5) integrates three terms into a unified framework. The first and second term lead to *low-rank Tucker approximation*, which aims to minimize the reconstruction error and obtains low-dimensional features. Because imposing the Tucker-based tensor nuclear norm on a core tensor $\mathcal{C}_m$ is equivalent to that on its original tensor $\mathcal{X}_m$ [48], we obtain a low-rank solution, i.e., R_n can be small ($R_n < I_n$). Therefore, the feature subspace is naturally low-dimensional. Moreover, imposing nuclear norm on core tensors instead of original ones reduces the computational cost. The third term (minimizing $-\sum_{m=1}^{M} \frac{1}{2}\|\mathcal{C}_m - \bar{\mathcal{C}}\|_F^2$) aims to maximize the variance of learned features inspired by PCA. TDVM-Tucker thus explores the relationship among tensor samples via *feature variance maximization* while estimating the missing data via low-rank Tucker approximation.

1) Derivation of TDVM-Tucker by ADMM: To facilitate the derivation of Eq. (5), we reformulate the equation by unfolding each tensor variable along mode-n and absorbing the constraints [1]. Thus, we obtain the Lagrange function as follows:

$$\begin{aligned}\mathcal{L} = & \sum_{m=1}^{M}\sum_{n=1}^{N} \Big(\frac{1}{2}\|\mathbf{X}_m^{(n)} - \mathbf{U}^{(n)}\mathbf{C}_m^{(n)}{\mathbf{H}^{(n)}}^\top\|_F^2 \\ & + \|\mathbf{S}_m^{(n)}\|_* + \langle \mathbf{Y}_{mn}, \mathbf{C}_m^{(n)} - \mathbf{S}_m^{(n)} \rangle \\ & + \frac{\mu}{2}\|\mathbf{C}_m^{(n)} - \mathbf{S}_m^{(n)}\|_F^2 - \frac{1}{2}\|\mathbf{C}_m^{(n)} - \bar{\mathbf{C}}^{(n)}\|_F^2 \Big)\end{aligned} \tag{6}$$

where $\mathbf{H}^{(n)} = \mathbf{U}^{(N)} \bigotimes \cdots \bigotimes \mathbf{U}^{(n+1)} \bigotimes \mathbf{U}^{(n-1)} \cdots \bigotimes \mathbf{U}^{(1)} \in \mathbb{R}^{\prod_{j \neq n} I_j \times \prod_{j \neq n} R_j}$, and μ and $\{\mathbf{Y}_{mn} \in \mathbb{R}^{R_n \times \prod_{j \neq n} R_j}, n = 1, \cdots, N, m = 1, \cdots, M\}$ are the Lagrange multipliers. $\mathbf{X}_m^{(n)} \in \mathbb{R}^{I_n \times \prod_{j \neq n} I_j}$ and $\{\mathbf{C}_m^{(n)}, \mathbf{S}_m^{(n)}, \bar{\mathbf{C}}^{(n)}\} \in \mathbb{R}^{R_n \times \prod_{j \neq n} R_j}$ are the mode-n unfolded matrices of $\mathcal{X}_m$ and {core tensor $\mathcal{C}_m$, auxiliary variable $\mathcal{S}_m$, mean of features $\bar{\mathcal{C}}$}, respectively.

ADMM solves the problem (6) by successively minimizing $\mathcal{L}$ over $\{\mathbf{S}_m^{(n)}, \mathbf{U}^{(n)}, \mathbf{C}_m^{(n)}, \mathbf{X}_m^{(n)}\}$, and then updating $\mathbf{Y}_{mn}$.

[1]For simplicity, the iteration number k is omitted in the updates of all variables in TDVM-Tucker and TDVM-CP optimization.

Algorithm 1 Low-rank **T**ensor (**Tucker**) **D**ecomposition with Feature **V**ariance **M**aximization (**TDVM-Tucker**)

Input: Incomplete tensors $\mathcal{P}_{\mathbf{\Omega}}(\mathcal{T})$, $\mathbf{\Omega}$, μ, and the maximum iterations K, feature dimension $D = [R_1, \cdots, R_N]$ (Tucker-rank), and stopping tolerance *tol*.
Initialization: Set $\mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m), \mathcal{P}_{\mathbf{\Omega}^c}(\mathcal{X}_m) = \mathbf{0}, m = 1, \cdots, M$; initialize $\{\mathcal{C}_m\}_{m=1}^M$ and $\{\mathbf{U}^{(n)}\}_{n=1}^N$ randomly; $\rho = 10, \mu_{max} = 1e10$.
for $k = 1$ **to** K **do**
  for $m = 1$ **to** M **do**
    for $n = 1$ **to** N **do**
      Update $\mathbf{S}_m^{(n)}$, $\mathbf{U}^{(n)}$ and $\mathbf{C}_m^{(n)}$ by (8), (11) and (13) respectively.
      Update $\mathbf{Y}_{mn}$ by $\mathbf{Y}_{mn} = \mathbf{Y}_{mn} + \mu(\mathbf{C}_m^{(n)} - \mathbf{S}_m^{(n)})$.
    end for
    Update $\mathcal{X}_m$ by (15).
  end for
  If $\|\mathcal{C}_m - \mathcal{S}_m\|_F^2 / \|\mathcal{C}_m\|_F^2 < tol$, break; otherwise, continue.
  Update $\mu_{k+1} = \min(\rho\mu_k, \mu_{max})$.
end for
Output: Tensorial features: $\mathcal{C} = [\mathcal{C}_1, \cdots, \mathcal{C}_m, \cdots \mathcal{C}_M] \in \mathbb{R}^{R_1 \times \cdots \times R_N \times M}$.

a) Update $\mathbf{S}_m^{(n)}$*:* Eq. (6) with respect to $\mathbf{S}_m^{(n)}$ is,

$$\mathcal{L}_{\mathbf{S}_m^{(n)}} = \sum_{m=1}^{M} \sum_{n=1}^{N} (\|\mathbf{S}_m^{(n)}\|_* + \frac{\mu}{2}\|(\mathbf{C}_m^{(n)} + \mathbf{Y}_{mn}/\mu) - \mathbf{S}_m^{(n)}\|_F^2), \quad (7)$$

where $\mathbf{S}_m^{(n)}$ is computed via soft-thresholding operator [49]:

$$\mathbf{S}_m^{(n)} = prox_{1/\mu}(\mathbf{C}_m^{(n)} + \mathbf{Y}_{mn}/\mu) = \mathbf{U}\text{diag}(\max \sigma - \frac{1}{\mu}, 0)\mathbf{V}^\top, \quad (8)$$

where *prox* is the soft-thresholding operation and $\mathbf{U}$ diag$(\max \sigma - \frac{1}{\mu}, 0)\mathbf{V}^\top$ is the SVD of $(\mathbf{C}_m^{(n)} + \mathbf{Y}_{mn}/\mu)$.

b) Update $\mathbf{U}^{(n)}$*:* Eq. (6) with respect to $\mathbf{U}^{(n)}$ is:

$$\mathcal{L}_{\mathbf{U}^{(n)}} = \sum_{m=1}^{M} \sum_{n=1}^{N} \frac{1}{2}\|\mathbf{X}_m^{(n)} - \mathbf{U}^{(n)}\mathbf{C}_m^{(n)}{\mathbf{H}^{(n)}}^\top\|_F^2, \text{ s.t. } {\mathbf{U}^{(n)}}^\top\mathbf{U}^{(n)} = \mathbf{I}, \quad (9)$$

The minimization of (9) over the matrices $\{\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}\}$ with orthonormal columns is equivalent to the maximization of the following problem [50]:

$$\mathbf{U}^{(n)} = \arg\max \text{ trace}\big({\mathbf{U}^{(n)}}^\top \mathbf{X}_m^{(n)} ({\mathbf{C}_m^{(n)}\mathbf{H}^{(n)}}^\top)^\top\big) \quad (10)$$

where trace() is the trace of a matrix, and we denote $\mathbf{W}^{(n)} = \mathbf{C}_m^{(n)}{\mathbf{H}^{(n)}}^\top$. The problem (10) is actually the well-known orthogonal Procrustes problem [51], whose global optimal solution is given by the SVD of $\mathbf{X}_m^{(n)}{\mathbf{W}^{(n)}}^\top$, i.e.,

$$\mathbf{U}^{(n)} = \hat{\mathbf{U}}^{(n)}(\hat{\mathbf{V}}^{(n)})^\top, \quad (11)$$

where $\hat{\mathbf{U}}^{(n)}$ and $\hat{\mathbf{V}}^{(n)}$ are the left and right singular vectors of SVD of $\mathbf{X}_m^{(n)}{\mathbf{W}^{(n)}}^\top$, respectively.

c) Update $\mathbf{C}_m^{(n)}$ *:* Eq. (6) with respect to $\mathbf{C}_m^{(n)}$ is:

$$\begin{aligned}\mathcal{L}_{\mathbf{C}_m^{(n)}} = \sum_{m=1}^{M}\sum_{n=1}^{N} \Big(&\|\mathbf{X}_m^{(n)} - \mathbf{U}^{(n)}\mathbf{C}_m^{(n)}{\mathbf{H}^{(n)}}^\top\|_F^2 \\ &+ \frac{\mu}{2}\|\mathbf{C}_m^{(n)} - \mathbf{S}_m^{(n)} + \mathbf{Y}_{mn}/\mu\|_F^2 \\ &- \frac{1}{2}\|(1 - \frac{1}{M})\mathbf{C}_m^{(n)} - \frac{1}{M}\sum_{j\neq m}^{M}\mathbf{C}_j^{(n)}\|_F^2\Big),\end{aligned} \quad (12)$$

setting the partial derivative $\partial\mathcal{L}_{\mathbf{C}_m^{(n)}}/\partial\mathbf{C}_m^{(n)}$ to zero, we get:

$$\begin{aligned}\mathbf{C}_m^{(n)} = \frac{M^2}{M^2\mu + 2M - 1}\Big(&\mu\mathbf{S}_m^{(n)} - \mathbf{Y}_{mn} + {\mathbf{U}^{(n)}}^\top \\ &\mathbf{X}_m^{(n)}\mathbf{H}^{(n)} - \big((\frac{1}{M} - \frac{1}{M^2})\sum_{j\neq m}^{M}\mathbf{C}_j^{(n)}\big)\Big).\end{aligned} \quad (13)$$

d) Update $\mathcal{X}_m$*:* Eq. (5) with respect to $\mathcal{X}_m$ is:

$$\begin{aligned}\sum_{m=1}^{M}\frac{1}{2}\|\mathcal{X}_m - \mathcal{C}_m \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}\|_F^2, \\ \text{s.t. } \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m),\end{aligned} \quad (14)$$

by deriving the Karush-Kuhn-Tucker (KKT) conditions for function (14), we can update $\mathcal{X}_m$ by:

$$\mathcal{X}_m = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) + \mathcal{P}_{\mathbf{\Omega}^c}(\mathcal{C}_m \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)}). \quad (15)$$

We summarize the proposed method, TDVM-Tucker, in **Algorithm** 1.

Remark 3: TDVM-Tucker explores the relationship among tensor samples via feature variance maximization while estimating the missing data via low-rank Tucker approximation, leading to low-dimensional informative features directly from observed entries. The proposed methods differ from both tensor completion methods and two-step strategies as follows.

- Tensor completion methods aim to recover incomplete tensors only without exploring the relationship among samples for effective feature extraction. In contrast, TDVM-Tucker focuses on extracting low-dimensional features instead of estimating missing data. Moreover, TDVM utilizes a feature constraint (feature variance maximization) to capture the relationship among samples for extracting informative features.
- Unlike the "two-step" strategies, which learn the features of incomplete data via two separate stages, TDVM-Tucker simultaneously estimates missing entries and learns low-dimensional features directly from the observed entries in the unified framework. The "two-step" strategies can amplify the approximation error because the missing data and the features are learned in separate stages, and the reconstruction error from the tensor completion step can deteriorate the performance of feature extraction in the subsequent step. This claim has been verified by our experimental results (as shown in the Tables I, II and III in Section IV-C). Therefore, TDVM-Tucker and TDVM-CP (which is introduced in the following) can extract more informative features within the unified framework.

C. *TDVM-CP: Learning Low-dimensional Vector Features*

We further propose another new TDVM method to learn low-dimensional vectorial features based on CP decomposition, i.e., incorporating low-rank CP decomposition with feature variance maximization, namely **TDVM-CP**. Because tensor decomposition with missing data is more challenging than that with complete data in traditional problems, here we consider incorporating orthogonality into the CP model for TDVM-CP (i.e., imposing orthogonality constraints on factors $\{\mathbf{u}_r^{(n)}\}$ in Eq. (3)) with the following two motivations:

- Like HOSVD [31], CP decomposition can be regarded as a generalization of SVD to tensors [52]. It appears natural to inherit the orthogonality of SVD in the CP model.
- The orthogonality constraint is considered unnecessary in general or even impossible in certain cases in exact CP decomposition [53], [54], [55], but some studies have

proved that imposing orthogonality in CP decomposition can transform a non-unique tensor model into a unique one with guaranteed optimality [54], [27], [56].

Like the orthogonality used in TDVM-Tucker, we believe that imposing orthogonality constraints can help TDVM-CP estimate missing values and extract features better. In addition, here we do not use the Tucker-based nuclear norm [25]; instead, we use a new CP-based tensor nuclear norm [2] [32] to achieve low-rank CP approximation.

In other words, TDVM-CP couples orthogonal CP decomposition with the CP-based tensor nuclear norm for the low-rank approximation, while maximizing the variance of learned features as the feature regularization term. Thus, the objective function of TDVM-CP is as follows:

$$\begin{aligned}\min_{\mathcal{X}_m,\mathbf{d}_m,\mathbf{u}_r^{(n)},R} \quad & \sum_{m=1}^{M}\frac{1}{2}\|\mathcal{X}_m-\sum_{r=1}^{R}d_{mr}\mathbf{u}_r^{(1)}\circ\cdots\circ\mathbf{u}_r^{(N)}\|_F^2 \\ & +\sum_{m=1}^{M}\lambda\|\mathbf{d}_m\|_1-\sum_{m=1}^{M}\frac{1}{2}\|\mathbf{d}_m-\bar{\mathbf{d}}\|_2^2, \qquad (16)\\ \text{s.t. } & \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m)=\mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m),{\mathbf{u}_r^{(n)}}^\top\mathbf{u}_r^{(n)}=1,n=1\cdots N,\\ & {\mathbf{u}_r^{(n)}}^\top\mathbf{u}_q^{(n)}=0,q=1\cdots r-1,r=1\cdots R,\end{aligned}$$

where $\|\mathbf{d}_m\|_1$ is the CP-based tensor nuclear norm on each weight vector, and we view the ***weight vector*** $\mathbf{d}_m\in\mathbb{R}^R$ of the orthogonal CP decomposition (analogous to the *vector of singular values* of a matrix) as the ***feature vector*** extracted from a tensor sample $\mathcal{X}_m$. $\bar{\mathbf{d}}=\frac{1}{M}\sum_{m=1}^{M}\mathbf{d}_m$ is the mean of the weight vectors (extracted features). $\lambda>0$ is a penalty parameter. Compared with TDVM-Tucker, TDVM-CP can obtain much lower dimensional features because it learns vectorial features from each tensor sample.

*1) **ADMM-BCD Joint Optimization** for TDVM-CP:* To solve the objective function Eq. (16) which is non-convex and non-smooth, we design a ADMM-BCD joint optimization scheme. We divide all the target variables into $M\times(R+1)$ groups: $\{\{d_{mr},\mathbf{u}_r^{(1)},\mathbf{u}_r^{(2)},\cdots,\mathbf{u}_r^{(N)}\}_{r=1}^{R},\mathcal{X}_m\}_{m=1}^{M}$, where we optimize a group of variables while fixing the other groups, and update one variable while fixing the other variables in each group. After updating the $R+1$ groups for each sample using BCD, we jump to the outside loop to update all samples iteratively using ADMM. To apply ADMM, we introduce a set of auxiliary variables $\{\mathbf{s}_m\in\mathbb{R}^R\}_{m=1}^{M}$ for the weight vectors $\{\mathbf{d}_m\}_{m=1}^{M}$, i.e., $\mathbf{s}_m=\mathbf{d}_m\in\mathbb{R}^R,m=1\cdots M$. Then, we formulate the Lagrangian function of Eq. (16) as follows:

$$\begin{aligned}\mathcal{L} \;=\; & \sum_{m=1}^{M}\Big(\frac{1}{2}\|\mathcal{X}_m-\sum_{r=1}^{R}d_{mr}\mathbf{u}_r^{(1)}\circ\cdots\circ\mathbf{u}_r^{(N)}\|_F^2+\lambda\|\mathbf{d}_m\|_1\\ & -\;\frac{1}{2}\|\mathbf{s}_m-\bar{\mathbf{s}}\|_2^2+\langle\mathbf{y}_m,\mathbf{d}_m-\mathbf{s}_m\rangle+\frac{\gamma}{2}\|\mathbf{d}_m-\mathbf{s}_m\|_2^2\Big)\\ & -\;\eta({\mathbf{u}_r^{(n)}}^\top\mathbf{u}_r^{(n)}-1)\;-\;\sum_{q=1}^{r-1}\mu_q{\mathbf{u}_r^{(n)}}^\top\mathbf{u}_q^{(n)},\end{aligned}\qquad(17)$$

where $\gamma,\eta,\{\mu_q\}_{q=1}^{r-1}$ and $\mathbf{y}_m$ are the Lagrange multipliers.

In the ADMM-BCD joint optimization, we first update the variables $\{d_{mr},\mathbf{u}_r^{(1)},\mathbf{u}_r^{(2)},\cdots,\mathbf{u}_r^{(N)}\}_{r=1}^{R}$ of each data sample *via BCD*. Thus, we formulate the Eq. (17) with respect to the r-th group $\{d_{mr},\mathbf{u}_r^{(1)},\mathbf{u}_r^{(2)}\cdots,\mathbf{u}_r^{(N)}\}$ as follows:

[2]For easy reading, we use $\|\mathbf{d}_m\|_1$ instead of $\|\mathcal{X}_m\|_{\text{CP}}$ in the derivation.

$$\begin{aligned}\mathcal{L}_{d_{mr},\mathbf{u}_r^{(n)}} \;=\; & \frac{1}{2}\|\mathcal{X}_{mr}-d_{mr}\mathbf{u}_r^{(1)}\circ\mathbf{u}_r^{(2)}\circ\cdots\circ\mathbf{u}_r^{(N)}\|_F^2+\lambda|d_{mr}|\\ & +\;\frac{\gamma}{2}\|d_{mr}+y_{mr}/\gamma-s_{mr}\|_2^2\\ & -\;\eta({\mathbf{u}_r^{(n)}}^\top\mathbf{u}_r^{(n)}-1)-\sum_{q=1}^{r-1}\mu_q{\mathbf{u}_r^{(n)}}^\top\mathbf{u}_q^{(n)},\end{aligned}\qquad(18)$$

where $\mathcal{X}_{mr}=\mathcal{X}_m-\sum_{q=1}^{r-1}d_{mq}\mathbf{u}_q^{(1)}\circ\mathbf{u}_q^{(2)}\circ\cdots\circ\mathbf{u}_q^{(N)}$ is the residual of the approximation of each tensor sample.

a) Update $\mathbf{u}_r^{(n)}$: Eq. (18) with respect to $\mathbf{u}_r^{(n)}$ is,

$$\begin{aligned}\mathcal{L}_{\mathbf{u}_r^{(n)}} \;=\; & \frac{1}{2}\|\mathcal{X}_{mr}-d_{mr}\mathbf{u}_r^{(n)}\circ\mathbf{u}_r^{(2)}\cdots\circ\mathbf{u}_r^{(N)}\|_F^2\\ & -\;\eta({\mathbf{u}_r^{(n)}}^\top\mathbf{u}_r^{(n)}-1)-\sum_{q=1}^{r-1}\mu_q\mathbf{u}_r^{(n)}\mathbf{u}_q^{(n)}.\end{aligned}\qquad(19)$$

Then we set the partial derivative of $\mathcal{L}_{\mathbf{u}_r^{(n)}}$ with respect to $\mathbf{u}_r^{(n)}$ to zero and eliminate the Lagrange multipliers, and get:

$$\begin{aligned}\mathbf{u}_r^{(n)} =& (\mathcal{X}_{mr}\times_j\{\mathbf{u}_r^{(j)}\}_{j\neq n})/d_{mr}\\ & -\big(\sum_{q=1}^{r-1}{\mathbf{u}_q^{(n)}}^\top(\mathcal{X}_{mr}\times_j\{\mathbf{u}_r^{(j)}\}_{j\neq n})\;\mathbf{u}_q^{(n)}\big)/d_{mr},\end{aligned}\qquad(20)$$

where $\mathcal{X}_{mr}\times_j\{\mathbf{u}_r^{(j)}\}_{j\neq n}=\mathcal{X}_{mr}\times_1\mathbf{u}_r^{(1)}\cdots\times_{(n-1)}\mathbf{u}_r^{(n-1)}\times_{(n+1)}\mathbf{u}_r^{(n+1)}\cdots\times_N\mathbf{u}_r^{(N)},j=1,2,\cdots,n-1,n+1,\cdots,N$, and we normalize $\mathbf{u}_r^{(n)}=\mathbf{u}_r^{(n)}/\|\mathbf{u}_r^{(n)}\|_2$. Note that we only update the variable groups with non-zero weights (i.e. $d_{mr}\neq0$).

b) Update d_{mr}: Eq. (18) with respect to d_{mr} is:

$$\begin{aligned}\mathcal{L}_{d_{mr}} =& \frac{1}{2}\|\mathcal{X}_r-d_{mr}\mathbf{u}_r^{(1)}\circ\mathbf{u}_r^{(2)}\cdots\circ\mathbf{u}_r^{(n)}\|_F^2\;+\;\lambda|d_{mr}|\\ & +\frac{\gamma}{2}\|d_{mr}+y_{mr}/\gamma-s_{mr}\|_2^2.\end{aligned}\qquad(21)$$

Setting the partial derivative $\partial\mathcal{L}_{d_{mr}}/\partial d_{mr}$ to zero, we obtain,

$$\begin{aligned}d_{mr}=&\frac{1}{(1+\gamma)}\Big(\gamma s_{mr}-y_{mr}+\mathcal{X}_r\times_1\mathbf{u}_r^{(1)}\times_2\mathbf{u}_r^{(2)}\\ & \cdots\times_N\mathbf{u}_r^{(N)}-\lambda|d_{mr}|/\partial d_{mr}\Big).\end{aligned}\qquad(22)$$

According to the soft thresholding algorithm [57] for L_1 regularization, we update d_{mr} by:

$$d_{mr}=\textit{shrink}_t(Q)=\begin{cases}Q-t & (Q>t)\\ 0 & (|Q|\le t)\\ Q+t & (Q<-t)\end{cases}.\qquad(23)$$

where *shrink* is the shrinkage operator [57], and $t=\frac{\lambda}{(1+\gamma)},Q=\frac{1}{(1+\gamma)}(\gamma s_{mr}-y_{mr}+\mathcal{X}_r\times_1\mathbf{u}_r^{(1)}\times_2\cdots\times_N\mathbf{u}_r^{(N)})$.

After updating $\{d_{mr},\mathbf{u}_r^{(1)},\mathbf{u}_r^{(2)},\cdots,\mathbf{u}_r^{(N)}\}_{r=1}^{R}$ by the BCD method, we jump out of the inner loop for each tensor sample and update the variables $\{\mathbf{s}_m,\mathcal{X}_m\}_{m=1}^{M}$ for all tensor samples iteratively *via ADMM*.

c) Update $\mathbf{s}_m$: Eq. (17) with respect to $\mathbf{s}_m$ is,

$$\mathcal{L}_{\mathbf{s}_m}=\sum_{m=1}^{M}\frac{1}{2}\gamma\|\mathbf{d}_m+\mathbf{y}_m/\gamma-\mathbf{s}_m\|_2^2-\sum_{m=1}^{M}\frac{1}{2}\|\mathbf{s}_m-\bar{\mathbf{s}}\|_2^2,\quad(24)$$

where $\mathbf{y}_m$ consists of Lagrange multipliers. Then we set the partial derivative $\partial\mathcal{L}_{\mathbf{s}_m}/\partial\mathbf{s}_m$ to zero and obtain,

$$\begin{aligned}\mathbf{s}_m=&\frac{M^2}{\gamma M^2+1-2M+M^2}\big(\gamma\mathbf{d}_m+\mathbf{y}_m\big)\\ & +\frac{M-1}{\gamma M^2+1-2M+M^2}\sum_{j\neq m}^{M}\mathbf{s}_j\end{aligned}\qquad(25)$$

Algorithm 2 Low-rank **T**ensor (**CP**) **D**ecomposition with Feature **V**ariance **M**aximization (**TDVM-CP**)

Input: Incomplete tensors $\mathcal{P}_{\mathbf{\Omega}}(\mathcal{T})$, $\mathbf{\Omega}$, λ, feature dimension $D = R$ (CP-rank), maximum iterations K, and *tol*.
Initialization: Set $\mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m)$, $\mathcal{P}_{\mathbf{\Omega}^c}(\mathcal{X}_m) = \mathbf{0}$, $\gamma = 10$; Initialize $\{\mathbf{u}_r^{(1)}, \mathbf{u}_r^{(2)}, \cdots \mathbf{u}_r^{(N)}\}_{r=1}^{R}, \{d_m\}_{m=1}^{M}$ randomly.
for $k = 1, ..., K$ **do**
  for $m = 1, ..., M$ **do**
    $\mathcal{X}_{mr} = \mathcal{X}_m$;
    for $r = 1, ..., R$ **do**
      if $d_{mr} \neq 0$ **then**
        Update $\mathbf{u}_r^{(n)}$ and d_{mr} by (20) and (23),respectively.
        $\mathcal{X}_{mr} = \mathcal{X}_{mr} - d_{mr}\mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \cdots \circ \mathbf{u}_r^{(N)}$.
      end if
    end for
    Update $\mathbf{s}_m$ and $\mathcal{X}_m$ by (24) and (27) respectively.
    Update $\mathbf{y}_m = \mathbf{y}_m + \gamma(\mathbf{d}_m - \mathbf{s}_m)$
  end for
  If $\|\mathbf{d}_m - \mathbf{s}_m\|_2^2 / \|\mathbf{d}_m\|_2^2 < tol$, break; otherwise, continue.
end for
output: Vectorial features $\mathbf{D} = [\mathbf{d}_1, \cdots \mathbf{d}_m, \cdots \mathbf{d}_M] \in \mathbb{R}^{R \times M}$.

d) Update $\mathcal{X}_m$: Eq. (16) with respect to $\mathcal{X}_m$ is,

$$\min_{\mathcal{X}_m} \frac{1}{2}\|\mathcal{X}_m - \sum_{r=1}^{R} d_{mr}\mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \cdots \circ \mathbf{u}_r^{(N)}\|_F^2, \quad \text{s.t. } \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}_m), \tag{26}$$

by deriving KKT conditions for Eq. (26), $\mathcal{X}_m$ is updated by:

$$\mathcal{X}_m = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}_m) + \mathcal{P}_{\mathbf{\Omega}^c}(\sum_{r=1}^{R} d_{mr}\mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \cdots \circ \mathbf{u}_r^{(N)}). \tag{27}$$

Using the ADMM-BCD joint optimization, we solve each subproblem of Eq. (16) in a closed-form. Finally, we summarize the proposed TDVM-CP in **Algorithm** 2.

Remark 4: TDVM-CP is similar in spirit to TDVM-Tucker, but it can yields features with lower dimension than TDVM-Tucker: the former extracts low-dimensional *vector* features from each data sample, while the latter aims to learn low-dimensional *tensor* features from each sample. Thus, using TDVM-CP to extract features can reduce the computational cost and memory requirements for further applications such as classification and clustering.

D. Computational Complexity Analysis

For TDVM-Tucker, we set the feature dimensions (Tucker-rank) $R_1 = R_2 \cdots = R_N = R$ for simplicity. In each iteration, the time complexity of computing the soft-thresholding operator (8) is $O(MNR^{N+1})$. The time complexities of multiplications in (11)/(13) and (15) are $O(MNR(\prod_{j=1}^{N} I_j))$ and $O(MR(\prod_{j=1}^{N} I_j))$, respectively. Thus, the total time complexity of TDVM-Tucker is $O(M(N+1)R(\prod_{j=1}^{N} I_j))$ in each iteration. For TDVM-CP, the time complexity of performing the shrinkage operator in (23) is $O(R(\prod_{j=1}^{N} I_j)$. This is also the time complexity of computing $\{\mathbf{u}_r^{(n)}\}_{n=1}^{N}$ and Eq. (27). Hence, the total time complexity of TDVM-CP is $O(MR(\prod_{j=1}^{N} I_j)$ in each iteration.

E. Discussion: General Model–TDFR

The proposed TDVM-Tucker and TDVM-CP essentially can be summarized into a general model, i.e., low-rank **T**ensor **D**ecomposition with **F**eature **R**egularization (**TDFR**):

$$\min_{\mathcal{X},Z} F(\mathcal{X}, Z) + G(Z) \text{ s.t. } \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}), \tag{28}$$

where $F(\mathcal{X}, Z)$ refers to a low-rank tensor decomposition model and $G(Z)$ is a regularization of target features Z. $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots I_N \times M}$ is the approximation of incomplete data (tensors) $\mathcal{T}$ based on observed entries indexed by $\mathbf{\Omega}$. Z is a component of $\mathcal{X}$ and could be a lower-dimensional tensor (e.g., a core tensor of Tucker model) or vector (e.g., a weight vector of CP model) that consists of all features extracted from $\mathcal{T}$.

In this paper, we specify TDFR by TDVM-Tucker and TDVM-CP. In addition, we briefly discuss more specific cases of TDFR. For example, considering the whole dataset as a tensor including all samples along the last mode, we can specify TDFR as follows:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{C},\mathbf{U}^{(n)},\mathbf{Z}} & \frac{1}{2}\|\mathcal{X} - \mathcal{C} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \cdots \times_N \mathbf{U}^{(N)} \times_{N+1} \mathbf{Z}\|_F^2 \\ & + \|\mathcal{C}\|_* - \frac{1}{2}\|\mathbf{Z}^\top\|_F^2, \\ \text{s.t. } & \mathcal{P}_{\mathbf{\Omega}}(\mathcal{X}) = \mathcal{P}_{\mathbf{\Omega}}(\mathcal{T}), {\mathbf{U}^{(n)}}^\top \mathbf{U}^{(n)} = \mathbf{I}, n = 1 \cdots N, \end{aligned} \tag{29}$$

where the $(N+1)$th factor matrix $\mathbf{Z} \in \mathbb{R}^{M \times R_{(N+1)}}$ are viewed as the extracted features from $\mathcal{T} = [\mathcal{T}_1, \cdots, \mathcal{T}_m, \cdots \mathcal{T}_M] \in \mathbb{R}^{I_1 \times \cdots \times I_N \times M}$. Such usage of treating the $(N+1)$th factor matrix as features can also be found in [58], [59]. Inspired by PCA, the third term: $\min -\frac{1}{2}\|\mathbf{Z}^\top\|_F^2 = \max \text{trace}(\mathbf{Z}\mathbf{Z}^\top)$, aims to maximize the variance of extracted features. Due to space limitations, more specific cases are discussed in Appendix A of the Supplementary Material [3].

Remark 5: As TDFR simultaneously estimates missing data via low-rank tensor approximation and explores the relationship among samples via feature regularization (e.g., maximizing variance of features in TDVM), we assume that TDFR can solve the problem of extracting features from incomplete tensors. In addition to the two proposed methods, there are many variants of specific cases of the general model TDFR: 1) For the low-rank approximation $F(\mathcal{X}, Z)$ of Eq. (28), we can not only use Tucker and CP decompositions in conjunction with the Tucker- and CP-based tensor nuclear norm, but can also consider other tensor decomposition models such as Tensor SVD [60], [11], Tensor-train decomposition [61], [62], etc., coupled with other constraints such as tensor nuclear norm [60], [11] to achieve low-rank tensor approximation; 2) For the feature regularization term $G(Z)$, we can use not only variance maximization for regularization such as TDVM but also other constraints such as uncorrelation or orthogonality, etc., to learn informative features.

IV. Experiments

We evaluate the performance of the proposed TDVM-Tucker and TDVM-CP on six real-world tensor datasets with $30\% - 90\%$ missing entries under multi-block missing settings. **"MR"** refers to the **M**issing **R**atio. We implement the proposed methods in MATLAB, and all experiments are performed on a PC (Intel Xeon(R) 4.0 GHz, 64 GB memory).

[3]**Supplementary Material**: https://www.dropbox.com/sh/zbqyofzwc5lsd0w/AABiDJVamrMuwwVfGUd-uvOfa?dl=0

A. Experimental Setup

1) Data: We evaluate TDVM-Tucker and TDVM-CP on **six** real-world datasets for three applications, including four third-order tensors and two fourth-order tensors [4]:

- For **face recognition**, we use two face datasets: one is a subset of the Facial Recognition Technology database (**FERET**)[5] [63], which has 721 face samples from 70 subjects. Each subject has 8 to 31 face images with at most 15 degrees of pose variation, and each face image is normalized to an 80×60 gray image. The other dataset is a subset of the extended Yale Face Database B (**YaleB**) [6] [64], which has 2414 face samples from 38 subjects. Each subject has 59 to 64 near frontal images under different illumination and each image is normalized to a 32×32 gray image.
- For **object/action classification** tasks, we evaluate two datasets: one is a subset of the **COIL-100** image database, which contains 100 different objects, each viewed from 72 different angles [7][65]. The size of each sample (totally 1000 samples) is normalized to a 64×64 gray image following [66]. The other dataset is a subset of the **Weizmann** action dataset [8] [67], which consists of 80 videos of 8 actors performing ten different actions: "bending", "jumping", "jumping jacks", "jumping in place", "running", "galloping sideways", "skipping", "walking, "one hand-waving", and "two hands waving". Each video is resized to $32 \times 22 \times 10$.
- For **face/gait clustering** tests, we also test two datasets: one is a subset of the **AR** face database [68], which contains 1200 face images with size 55×40 of 100 subjects including images of non-occluded faces, and face occluded by scarves/glasses following [66][9]; the other dataset is the gallery set (731 samples from 71 subjects) of the **USF** HumanID "**Gait** Challenge" database [10] [69]. Each gait video sample is resized to $64 \times 44 \times 20$.

2) Compared Methods: We compare TDVM-Tucker and TDVM-CP with *17* methods in four categories [11]:

(i) *Three* Tucker-/CP- based tensor completion methods: **HaLRTC** [25], **TenALS** [27] and **TNCP** [26].

(ii) *Nine* {tensor completion methods + feature extraction methods} (i.e., "two-step" strategies): *HaLRTC* + **MPCA** [3], *TenALS* + **MPCA**, *TNCP* + **MPCA**, *HaLRTC* + **SOMPCARS** [6], *TenALS* + **SOMPCARS**, *TNCP* + **SOMPCARS**, *HaLRTC* + **LRANTD** [4], *TenALS* + **LRANTD**, *TNCP* + **LRANTD**.

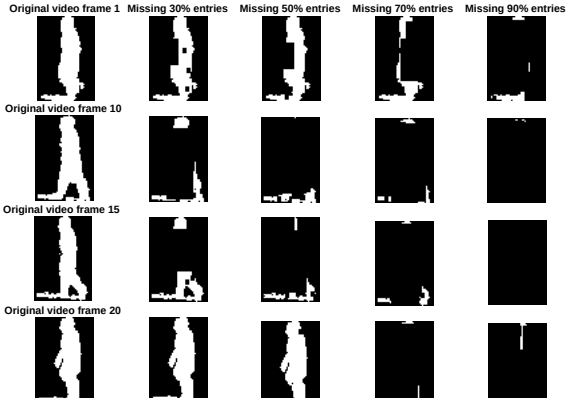
(iii) *One* robust tensor feature learning method: **TRPCA** [11].

(iv) *Four* clustering methods (used for the comparison of clustering with missing data): Sparse subspace clustering (**SSC**) [43], Zero-Fill + SSC (**ZF + SSC**) [21], SSC by Column-wise Expectation-based Completion (**SSC-CEC**) [21], Sparse Representation with Missing Entries and Matrix Completion (**SRME-MC**) [22].

We compare the 13 methods of the first three categories with respect to face recognition and object/action classification tasks, and compare all 17 methods in face/gait clustering tests. After feature extraction, we use the Nearest Neighbors Classifier (**NNC**) to evaluate the extracted features for face recognition and object/action classification. For face/gait clustering tests, we use the **K-means** [71] to cluster the features extracted by the first 13 methods and use a spectral clustering technique as a post-processing step for the four subspace clustering methods.

3) Multi-block Missing (MbM) Setting: In this paper, we design a **"Multi-block Missing (MbM)"** setting to generate random missing patterns of tensors. According to the data sample size, we use a set of tensorial blocks with different sizes as missing blocks to generate missing entries randomly in each tensor sample. We progress from the largest missing blocks to the smallest missing blocks to generate missing patterns until the required ratio of missing entries is achieved. For example, we can use a random set of missing blocks $\{32 \times 20 \times 10, 20 \times 15 \times 5, 4 \times 3 \times 4, 1 \times 1 \times 1\}$ to obtain an incomplete USF gait database (sample size $64 \times 44 \times 20$) with 50% missing entries. We first use the k largest ($32 \times 20 \times 10$) blocks to create missing entries randomly until the $(k+1)$th largest block exceeds the required missing ratio (e.g., $k = 5$); then we use the p second largest ($20 \times 15 \times 5$) blocks until the $(p+1)$th second largest block exceeds the required missing ratio (e.g., $p = 12$). We continue by using the s third largest

(a) FERET with $\{20 \times 15,\ 6 \times 8,\ 4 \times 4,\ 1 \times 1\}$ multi-block missing entries.



(b) USF gait with $\{32 \times 20 \times 10,\ 20 \times 15 \times 5,\ 4 \times 3 \times 4,\ 1 \times 1 \times 1\}$ multi-block missing entries.

Fig. 3. Examples of (a) one sample of FERET database (b) four frames of the first video sample (20 frames) of USF gait database, with $\{30\%, 50\%, 70\%, 90\%\}$ missing entries generated by MbM settings.

[4]For fast evaluation, we use resized tensor samples with smaller dimensions, while the proposed methods are applicable to original (larger) tensors without subsampling (resizing). Refer to Appendix D of the Supplementary Material for results on large tensors.

[5]http://www.dsp.utoronto.ca/~haiping/MSL.html

[6]http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html

[7]http://machineilab.org/users/pengxi

[8]http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html

[9]http://machineilab.org/users/pengxi

[10]http://www.dsp.utoronto.ca/~haiping/MSL.html

[11]We have also compared with the state-of-the-art tensor singular value decomposition (t-SVD) methods such as [70] and its combined two-step strategies. Although the t-SVD based tensor completion methods slightly outperform the Tucker and CP based methods (such as HaLRTC and TNCP), they still give much poorer feature extraction results than our TDVM methods. Because the proposed methods are based on the Tucker and CP models, we do not present the comparison against t-SVD methods here for simplicity and please refer to Appendix C of the Supplementary Material for these results.

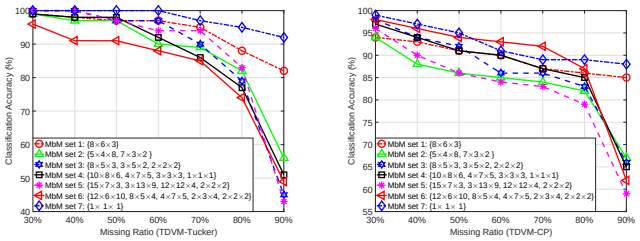
$(4\times3\times4)$ blocks (e.g., $s=25$) to generate the missing data successively. Finally, we use the q smallest missing blocks with smallest size (e.g., $q=70$) to make up the remaining missing region. Thus, we use $(k+p+s+q)$ missing blocks of different sizes to generate an incomplete USF gait tensor sample with 50% missing entries. Here, these missing blocks can be overlapped (i.e., the values of k,p,s,q are different in different samples) and the missing blocks are distributed randomly in each tensor sample. Hence, the irregular missing shapes (positions of missing data, i.e., $\mathbf{\Omega}$) are different in each tensor sample, while the total number of missing entries is the same. Nevertheless, one can set any types of MbM sets with multiple blocks of different sizes under the MbM setting. Figure 3 illustrates the data samples with missing entries generated by the proposed MbM setting.

Remark 6: The Multi-block Missing setting generates different irregular missing shapes (missing patterns) in tensor samples, which is more general and practical in real-world applications. MbM setting with only one type of block (with size $=1$) is equivalent to the pixel-based missing (uniformly selecting MR (e.g., MR $=50\%$) pixels (entries) from each tensor sample as missing at random) which is widely used in matrix/tensor completion fields. MbM setting with only one type of block (with size >1) is equivalent to the block-based missing setting (randomly selecting a single block entries of each tensor sample as missing) which is also commonly used in missing data imputation. In other words, existing missing data settings are special cases of our MbM setting. Intuitively, handling data with general *multi-block missing* is more difficult than that with *pixel-based missing* and *block-based missing* if the number of missing entries is the same. The reason is that the MbM setting is somehow close to the *non-random missing* setting especially when MR is higher (e.g., when MR$=90\%$, some whole rows/columns of images/videos are missing as shown in Figure 3), although the MbM setting is essentially random block missing with overlapping.

4) Parameter Settings: We set the maximum iterations $K=500$, $tol=1e{-}5$ for all methods, although our methods usually converge within 10 iterations. For Tucker decomposition-based methods, namely, TDVM-Tucker and LRANTD, we set the feature dimension $D=[R_1,R_2,\cdots,R_N]$ (Tucker-rank) $=$ round $(1/2{\times}([I_1,I_2,\cdots I_N]))$ for each tensor sample. For CP decomposition-based methods, namely, TDVM-CP, TenALS and TNCP, we set $D=R$ (CP-rank) $=$ round $(\min\{1/2\times$ mean$([I_1,I_2,\cdots,I_N]),\min([I_1,I_2,\cdots,I_N])\})$ for each sample. For other parameters of the compared methods, we have tuned the parameters based on the original papers to obtain the best results under same experimental settings. On the other hand, we further evaluate extracted features for classification via NNC, in which we randomly select $L=\{1,7\}$ extracted feature samples from each subject of FERET for training in NNC. Similarly, we set $L=\{5,50\}$, $\{1,8\}$ and $\{1,7\}$ on the YaleB, COIL-100 and Weizmann datasets, respectively.

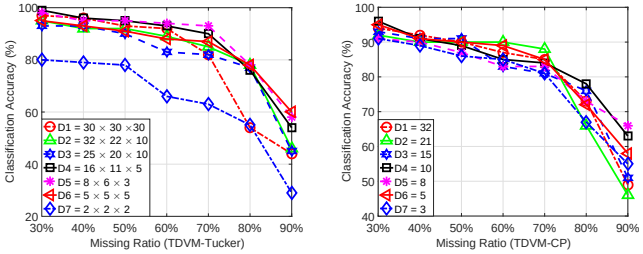
B. Analysis of Different (Parameter) Settings and Convergence

1) Effect of Different Multi-block Missing Settings: Here, we study the effect of applying TDVM-Tucker and TDVM-CP to datasets with different MbM settings. We randomly set



(a) TDVM-Tucker on Weizmann with MbM settings (b) TDVM-CP on Weizmann with MbM settings

Fig. 4. Classification results of Weizmann with $30\%-90\%$ missing entries generated by seven different MbM settings via TDVM-Tucker and TDVM-CP (feature dimension $D=\{16\times11\times5,10\}$ respectively).

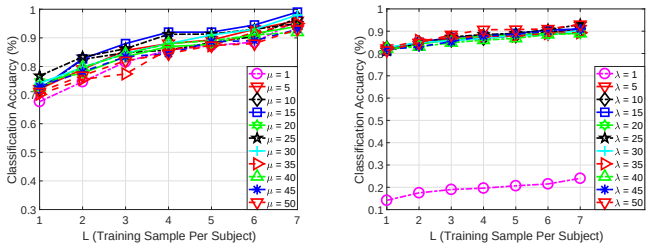


(a) TDVM-Tucker with different feature dimensions (b) TDVM-CP with different feature dimensions

Fig. 5. Classification results on Weizmann with $30\%-90\%$ missing entries (MbM set $=\{10\times8\times6,4\times7\times5,3\times3\times3,1\times1\times1\}$) via TDVM-Tucker and TDVM-CP with seven different feature dimensions.

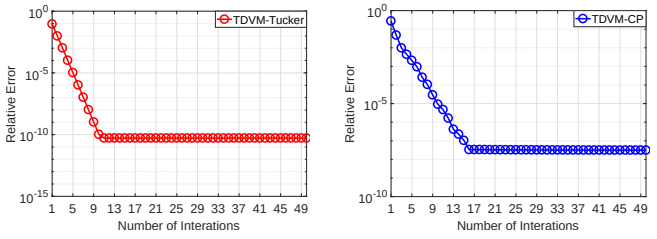
seven MbM sets using different types of missing blocks to generate missing pattern on the Weizmann database to obtain incomplete Weizmann data, i.e., *MbM set 1* using only one type of block with size $\{8\times6\times3\}$, which also refers to the commonly used block-based missing setting; *MbM set 2* using two types of blocks:$\{5\times4\times8,7\times3\times2\}$; *MbM set 3* using three types of blocks: $\{8\times5\times3,3\times5\times2,2\times2\times2\}$; *MbM set 4* using four types of blocks: $\{10\times8\times6,4\times7\times5,3\times3\times3,1\times1\times1\}$; *MbM set 5* using four types of blocks: $\{15\times7\times3,3\times13\times9,12\times12\times4,2\times2\times2\}$; *MbM set 6* using five types of blocks: $\{12\times6\times10,8\times5\times4,4\times7\times5,2\times3\times4,2\times2\times2\}$; and *MbM set 7* using only one type of block with size $=1$ $(1\times1\times1)$, which is equivalent to the pixel-based missing setting widely used in matrix/tensor completion. Using the seven MbM sets, we generate an incomplete Weizmann database $(32\times22\times10\times80)$ with $30\%-90\%$ missing entries. TDVM-Tucker and TDVM-CP directly extract $16\times11\times5\times80$ and 10×80 features from these incomplete tensors, respectively, and these features are further evaluated via NNC using $L=7$ video feature samples per subject (each subject has 8 samples) as training.

Figure 4 shows that on the Weizmann dataset with various missing patterns using different random MbM sets, both TDVM-Tucker and TDVM-CP consistently yield good results. Two cases are particularly worth mentioning. On the Weizmann dataset with *MbM set 1* and *set 7*, TDVM-Tucker and TDVM-CP can achieve better classification results than other cases (*MbM set 2-6*) especially when MR $>70\%$. This verifies our claim mentioned in Remark 6: handling data with the MbM setting which uses multiple missing blocks (MbM set 2-6) is more difficult than that with existing block-based missing (*MbM set 1*) and pixel-based missing (*MbM set 7*) settings. For general MbM settings (MbM set 2-6), TDVM-Tucker and TDVM-CP can obtain similar results with acceptable deviation of classification accuracy. On the other hand, using these



(a) TDVM-Tucker with different μ (b) TDVM-CP with different λ

Fig. 6. Classification results on Weizmann with 50% missing entries (MbM set =$\{10 \times 8 \times 6, 4 \times 7 \times 5, 3 \times 3 \times 3, 1 \times 1 \times 1\}$) via TDVM-Tucker and TDVM-CP with 11 different values of μ and λ, respectively.



(a) TDVM-Tucker (b) TDVM-CP

Fig. 7. Convergence curves of TDVM-Tucker and TDVM-CP in terms of *Relative Error*: $\|\mathcal{G}_m - \mathcal{S}_m\|_F^2/\|\mathcal{G}_m\|_F^2$ and $\|\mathbf{d}_m - \mathbf{s}_m\|_2^2/\|\mathbf{d}_m\|_2^2$ respectively, on Weizmann with 50% missing entries (MbM set = $\{10 \times 8 \times 6, 4 \times 7 \times 5, 3 \times 3 \times 3, 1 \times 1 \times 1\}$).

MbM sets with different types of missing blocks, the achieved missing ratios are likely slightly different, especially for these MbM sets without size $= 1$ block to make up the remaining missing entries. For example, with MbM set 5, we actually obtain Weizmann with $29.94\% - 89.92\%$ instead of *exact* 30%– 90% missing entries because of the sizes of the missing blocks. This slight difference of actual number of missing entries can also slightly affect the classification results, leading to an increase in the deviation of classification accuracy under different MbM settings.

In short, the proposed TDVM-Tucker and TDVM-CP are are not highly sensitive to the missing patterns overall and consistently yield good results under various MbM settings. Thus, in the following tests, we test datasets with four types of missing blocks in MbM settings for simplicity, and each MbM set includes the size $= 1$ block to ensure the total number of missing entries is the same under different MbM settings.

2) Effect of Different Feature Dimensions: We study the effect of different feature dimensions used in TDVM-Tucker and TDVM-CP for feature extraction on an incomplete Weizmann database. Figure 5 shows that, with different dimensions of features, TDVM-Tucker and TDVM-CP yield similar classification results stably on the whole, except in the case of TDVM-Tucker with $D7 = 2 \times 2 \times 2$ (i.e., only 8 features are extracted from each $32 \times 22 \times 10$ video) where the number of features is too limited to achieve good results. TDVM-CP obtains much fewer learned features, but it consistently achieve good results. On the other hand, as TDVM-Tucker and TDVM-CP are based on the orthogonal Tucker and CP models respectively, the dimension of *effective features* for TDVM-Tucker is upper-bound by the data dimension in each mode, and that of TDVM-CP is limited by the minimum sample dimension. Thus, setting $\{D1 = 30 \times 30 \times 30 > 32 \times 22 \times 10\}$ for TDVM-Tucker and $\{D1{=}32, D2{=}21, D3{=}15 > 10{=}\min[32, 22, 10]\}$ for TDVM-CP leads to a slight deterioration of classification performance especially in the cases of Weizmann with higher missing ratio (e.g., MR $= 90\%$), as seen from Figs. 5(a) and 5(b) respectively.

In short, the proposed methods are not sensitive to the feature dimensions. Since a larger feature dimension will lead to higher computational costs and memory requirements, and we aim to learn low-dimensional features, we thus set $D = \text{round}\ (1/2 \times ([I_1, I_2, \cdots I_N]))$ and $D = \text{round}\ (\min\{1/2 \times \text{mean}([I_1, I_2, \cdots, I_N]), \min([I_1, I_2, \cdots, I_N])\})$ for TDVM-Tucker and TDVM-CP by default, respectively.

3) Sensitivity Analysis of Parameter μ and λ: Figure 6 shows the classification results given features extracted by TDVM-Tucker and TDVM-CP with 11 different values for the penalty parameters μ and λ, respectively, on Weizmann videos with 50% missing entries via an MbM set. Figure 6(a) show that TDVM-Tucker yields good results stably with different values of μ. Figure 6(b) shows that the feature extraction performance of TDVM-CP is also stable and not sensitive to the values of λ, except for the case in which $\lambda = 1$. In other words, the proposed methods are not sensitive to the parameters overall. In addition, as the parameters ρ and γ can be fixed (fix ρ = 10, γ = 1) within Algorithm 1 and Algorithm 2 respectively based on preliminary studies, we thus do not examine them here.

In short, we do not need to carefully tune the parameters μ and λ for TDVM-Tucker and TDVM-CP, respectively. In this paper, we fix $\mu = \lambda = 10$ for all tests.

4) Convergence: We study the convergence of TDVM in terms of the relative error on a Weizmann dataset with 50% missing entries via an MbM set. Figure 7 shows that: TDVM-Tucker converges within 10 iterations while TDVM-CP requires more iterations (about 20) to reach convergence. If set $tol = 1e{-}5$, our methods converge fast with around 5-10 iterations.

C. Evaluation of Extracted Features from Incomplete Tensors

To save space, we report the results of six real tensor datasets with $\{30\%, 50\%, 70\%, 90\%\}$ missing pixels under random MbM settings in Tables I, II and III [12]. We highlight the **best** results in **bold** font and underline the second best results, and we use "–" to indicate that the method diverges (e.g., TenALS) in some cases. The average results of 10 runs are reported.

*1) **Face Recognition**:* Table I shows that TDVM-Tucker and TDVM-CP consistently outperform all the methods compared in all cases. Specifically, TDVM-Tucker and TDVM-CP directly learn $40 \times 30 \times 721$ features and 35×721 features from FERET ($80 \times 60 \times 721$) with $\{30\%, 50\%, 70\%, 90\%\}$ missing pixels via a random MbM set ($\{32 \times 32, 10 \times 4, 8 \times 16, 1 \times 1\}$). As shown in the left half of Table I, TDVM-Tucker and TDVM-CP share the two best recognition results,

[12]The proposed methods are based on low-rank decompositions and thus can yield good results on tensors with good low-rank structure even when the missing ratio reaches 90%. However, if too many (e.g., $95\%, 99\%$) entries are missing, the performance of our methods will drop dramatically.

TABLE I
FACE RECOGNITION RESULTS (AVERAGE RECOGNITION RATES %) ON THE FERET AND YALEB DATABASE WITH $\{30\%, 50\%, 70\%, 90\%\}$ MISSING ENTRIES UNDER MULTI-BLOCK MISSING SETTINGS.

Data	FERET (Image sample size 80×60)								YaleB (Image sample size 32×32)							
Multi-Block Missing Setting	$\{20 \times 15,\ 6 \times 8,\ 4 \times 4,\ 1 \times 1\}$								$\{16 \times 16,\ 10 \times 5,\ 4 \times 8,\ 1 \times 1\}$							
Missing Ratio (MR)	30%		50%		70%		90%		30%		50%		70%		90%	
L	1	7	1	7	1	7	1	7	5	50	5	50	5	50	5	50
HaLRTC [25]	34.1	71.2	31.1	67.1	26.3	58.4	19.2	46.3	32.9	73.9	30.8	70.8	27.2	63.0	19.5	44.8
TenALS [27]	31.3	65.6	29.4	63.3	25.0	55.5	12.4	24.8	25.0	56.5	18.0	38.5	7.7	14.8	2.7	2.7
TNCP [26]	33.6	71.0	32.0	66.6	26.5	59.4	20.8	45.2	28.9	66.9	24.5	58.7	20.1	47.8	17.8	36.5
HaLRTC + **MPCA** [3]	40.4	73.8	30.8	65.2	21.2	51.7	16.9	41.0	24.7	65.6	12.9	38.4	10.2	18.5	8.7	20.7
TenALS + **MPCA** [3]	31.1	62.0	23.9	49.4	12.9	24.9	4.8	9.1	18.6	40.8	16.8	36.0	13.4	24.2	8.5	13.8
TNCP + **MPCA** [3]	33.4	63.7	19.7	33.7	13.7	22.3	12.6	18.4	19.9	55.4	11.4	30.7	12.9	26.3	10.3	16.1
HaLRTC + **SOMPCARS** [6]	32.0	63.7	25.7	51.1	17.7	30.8	12.2	19.6	11.6	23.3	11.4	20.1	11.2	19.3	8.5	11.7
TenALS + **SOMPCARS** [6]	28.5	52.2	11.8	17.1	6.4	7.8	3.6	4.9	11.1	24.0	12.9	22.4	11.1	20.8	8.4	14.4
TNCP + **SOMPCARS** [6]	29.5	55.2	14.4	26.4	7.5	10.7	8.5	10.0	17.8	38.8	15.3	33.7	9.8	17.7	6.2	10.3
HaLRTC + **LRANTD** [4]	34.0	72.3	30.5	66.4	25.3	58.3	19.2	45.7	24.5	58.4	21.4	52.3	17.6	42.5	13.8	29.7
TenALS + **LRANTD** [4]	30.9	65.5	29.0	63.2	26.4	58.4	14.6	29.4	12.7	27.5	12.9	26.5	13.2	25.6	9.5	16.5
TNCP + **LRANTD** [4]	34.1	71.0	31.1	67.9	26.8	60.2	20.1	45.0	21.1	49.6	17.8	41.9	16.3	36.6	15.5	31.2
TRPCA [11]	30.0	64.4	21.8	51.1	17.8	43.1	16.5	36.0	32.5	72.5	30.0	66.3	26.2	54.4	19.6	36.0
TDVM-Tucker	75.7	**92.1**	73.9	**91.1**	71.8	**91.0**	70.6	**90.3**	58.2	94.8	47.2	93.0	46.2	92.1	45.1	91.0
TDVM-CP	**76.3**	90.9	**75.9**	88.0	**75.2**	87.2	**72.6**	86.6	**94.9**	**97.6**	**93.0**	**95.6**	**90.6**	**95.4**	**81.9**	**91.4**

TABLE II
CLASSIFICATION RESULTS (AVERAGE CLASSIFICATION ACCURACIES %) OF THE COIL-100 OBJECT IMAGES AND WEIZMANN ACTION VIDEOS WITH $\{30\%, 50\%, 70\%, 90\%\}$ MISSING ENTRIES UNDER MULTI-BLOCK MISSING SETTINGS.

Data	COIL-100 (Image sample size 64×64)								Weizmann (Video sample size $32 \times 22 \times 10$)							
Multi-Block Missing Setting	$\{32 \times 32,\ 10 \times 4,\ 8 \times 16,\ 1 \times 1\}$								$\{10 \times 8 \times 6,\ 4 \times 7 \times 5,\ 3 \times 3 \times 3,\ 1 \times 1 \times 1\}$							
Missing Ratio (MR)	30%		50%		70%		90%		30%		50%		70%		90%	
L	1	8	1	8	1	8	1	8	1	7	1	7	1	7	1	7
HaLRTC [25]	49.4	67.3	47.1	65.6	43.7	58.7	30.9	41.5	47.1	77.0	35.7	64.0	21.9	44.0	11.3	19.0
TenALS [27]	50.8	71.7	44.9	60.8	35.3	42.9	21.3	25.7	25.6	56.0	10.6	15.0	–	–	–	–
TNCP [26]	50.7	68.7	49.5	65.4	44.6	57.9	30.5	38.9	54.9	90.0	54.1	87.0	40.7	66.0	19.9	41.0
HaLRTC + **MPCA** [3]	57.8	77.7	50.8	68.2	39.0	51.0	9.6	5.3	56.3	91.0	35.9	65.0	24.6	39.0	12.4	21.0
TenALS + **MPCA** [3]	39.3	55.5	26.0	35.5	16.6	18.0	10.2	5.2	–	–	–	–	–	–	–	–
TNCP + **MPCA** [3]	52.5	70.3	47.8	62.8	19.1	21.8	14.0	11.1	59.0	88.0	44.0	68.0	26.4	44.0	10.6	19.0
HaLRTC + **SOMPCARS** [6]	42.9	57.3	43.8	57.3	20.5	24.3	10.3	5.0	39.1	63.0	25.7	44.0	20.7	40.0	16.4	28.0
TenALS + **SOMPCARS** [6]	37.6	45.4	22.0	22.3	13.3	10.7	9.5	3.9	–	–	–	–	–	–	–	–
TNCP + **SOMPCARS** [6]	30.4	37.1	21.6	24.3	16.9	14.9	17.3	14.7	45.1	65.0	38.1	53.0	21.6	38.0	11.7	18.0
HaLRTC + **LRANTD** [4]	51.8	71.4	51.2	68.6	47.1	62.0	15.8	15.2	55.4	90.0	52.1	87.0	44.4	82.0	22.0	43.0
TenALS + **LRANTD** [4]	51.6	70.2	47.3	63.1	37.0	46.8	20.6	21.0	–	–	–	–	–	–	–	–
TNCP + **LRANTD** [4]	52.0	72.0	50.0	66.2	45.9	60.6	30.4	37.1	56.7	90.0	54.7	82.0	45.6	69.0	19.4	38.0
TRPCA [11]	42.2	58.4	33.9	49.5	24.8	28.8	14.5	11.8	46.9	77.0	38.4	67.0	25.4	49.0	13.3	21.0
TDVM-Tucker	82.9	**96.8**	76.7	**96.1**	**76.3**	**94.6**	70.3	**93.7**	85.7	**99.0**	74.0	**96.0**	51.6	85.0	44.7	48.0
TDVM-CP	**83.4**	92.4	82.0	92.0	73.9	86.0	**72.6**	85.2	82.0	94.0	**82.3**	89.0	**82.3**	**87.0**	**56.7**	**61.0**

while TDVM-CP shows greater advantages, particularly when the number of training features is smaller (e.g., $L = 1$). As the missing ratio increases, the performance of the compared methods drops more quickly than that of our methods, which retain high accuracy. When there are 90% missing entries, TDVM-Tucker and TDVM-CP outperform all other methods by $\{60.9\%, 59.4\%\}$ on average, respectively.

As reported in the right half of Table I: TDVM-Tucker and TDVM-CP outperform the 13 competing methods by $29.1\% - 69.2\%$ and $47.3\% - 75.5\%$ on average, respectively. TDVM-CP consistently yields the best results in all cases, especially in the case of $L = 5$ (use only 5 feature samples per subject for training), where the method outperforms the second best performing method (TDVM-Tucker) by $\{36.7\%, 45.8\%, 44.4\%, 36.8\%\}$ on YaleB with $\{30\%, 50\%, 70\%, 90\%\}$ missing entries, respectively. Here, TDVM-CP only uses 16×2414 features extracted directly from the YaleB database ($32 \times 32 \times 2414$). Moreover, HaLRTC, TRPCA and TNCP perform better than other existing methods on the whole, although their results are much worse than ours.

*2) **Object/Action Classification**:* We further evaluate the proposed methods using COIL-100 object images ($64 \times 64 \times 1000$) and Weizmann action videos ($32 \times 22 \times 10 \times 80$) for object and action classification, respectively. Table II shows that TDVM-Tucker and TDVM-CP outperform the compared methods in all cases. Specifically, TDVM-Tucker outperforms the other methods by $\{34.7\%, 38.5\%, 50.8\%, 63.9\%\}$ in cases of COIL-100 with $\{30\%, 50\%, 70\%, 90\%\}$ missing values respectively, where TDVM-CP outperforms these compared methods by $\{32.8\%, 39.1\%, 45.3\%, 60.8\%\}$ respectively using 32 features learned from each object sample. These results clearly demonstrate that: With more missing entries, the proposed methods show more superiority, particularly when the missing rate reaches 90% where the performance of the other methods drops sharply. This observation is further confirmed in the cases of Weizmann action videos: Although some

TABLE III
CLUSTERING RESULTS (AVERAGE NORMALIZED MUTUAL INFORMATION (NMI) AND CLUSTERING ACCURACY (ACC) %) ON THE AR FACIAL IMAGES AND USF GAIT VIDEOS WITH $\{30\%, 50\%, 70\%, 90\%\}$ MISSING ENTRIES UNDER MULTI-BLOCK MISSING SETTINGS.

Data	AR (Image sample size 55×40)								USF gait (Video sample size $64 \times 44 \times 20$)							
Multi-Block Missing Setting	$\{20 \times 20,\ 16 \times 7,\ 3 \times 10,\ 1 \times 1\}$								$\{32 \times 20 \times 10,\ 20 \times 15 \times 5,\ 4 \times 3 \times 4,\ 1 \times 1 \times 1\}$							
Missing Ratio (MR)	30%		50%		70%		90%		30%		50%		70%		90%	
Metric	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC	NMI	ACC
HaLRTC [25]	51.5	17.5	48.5	15.8	49.0	15.4	46.6	15.0	49.7	19.0	48.7	17.5	48.9	16.1	48.5	15.5
TenALS [27]	50.1	15.9	48.2	15.3	45.2	13.3	42.5	12.6	–	–	–	–	–	–	–	–
TNCP [26]	51.1	17.8	48.9	15.8	46.3	14.8	45.6	14.3	49.2	18.2	48.2	16.6	48.5	16.3	48.1	15.2
HaLRTC + **MPCA** [3]	49.6	17.6	48.6	15.8	46.6	15.4	44.8	15.0	49.7	19.8	48.8	16.3	48.2	14.9	48.1	14.9
TenALS + **MPCA** [3]	50.2	16.3	47.4	15.1	44.4	13.7	41.8	12.7	–	–	–	–	–	–	–	–
TNCP + **MPCA** [3]	50.0	16.0	49.0	15.1	47.2	14.8	46.1	13.8	50.3	19.0	48.9	16.8	48.4	15.5	48.0	14.9
HaLRTC + **SOMPCARS** [6]	52.0	18.1	50.6	17.4	47.7	15.6	45.6	14.8	49.6	20.0	47.8	18.1	45.7	17.2	45.4	16.7
TenALS + **SOMPCARS** [6]	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
TNCP + **SOMPCARS** [6]	50.8	16.7	48.5	15.2	47.3	15.0	46.0	14.1	50.3	20.0	48.5	17.4	48.2	17.2	47.6	15.9
HaLRTC + **LRANTD** [4]	50.8	16.5	49.2	15.8	47.9	15.7	46.3	15.6	49.7	18.7	48.3	15.9	48.5	15.3	48.3	15.3
TenALS + **LRANTD** [4]	50.6	16.7	47.9	15.5	45.2	13.9	43.3	13.7	–	–	–	–	–	–	–	–
TNCP + **LRANTD** [4]	51.2	17.3	49.8	16.4	48.3	15.1	45.9	14.8	49.7	18.7	49.2	16.6	49.2	16.1	48.1	15.2
TRPCA [11]	45.8	12.9	42.7	12.4	39.1	11.7	36.1	10.8	48.2	18.2	47.0	15.2	46.0	15.2	45.8	13.8
***SSC* [43]**	46.3	13.9	46.3	13.3	45.8	13.1	45.5	13.0	49.4	19.4	47.0	16.3	45.9	15.2	45.2	14.3
SSC-CEC [21]	48.1	14.3	47.6	13.4	47.5	13.0	46.5	12.1	44.4	14.6	44.4	13.5	–	–	–	–
ZF + SSC[21]	48.8	15.0	48.5	14.2	47.7	13.5	47.6	13.2	50.0	19.4	48.6	16.8	47.9	15.7	46.7	13.8
SRME-MC [22]	49.1	15.6	47.3	12.3	47.3	12.1	47.1	11.8	50.0	19.3	47.4	15.6	47.1	14.8	47.0	14.6
TDVM-Tucker	**85.8**	**65.1**	**84.2**	**61.8**	**83.2**	**61.6**	**82.1**	**58.8**	**90.0**	**75.4**	**88.4**	**71.8**	**88.2**	**71.8**	**87.6**	**70.3**
TDVM-CP	84.2	59.6	82.3	54.8	79.2	48.7	77.0	45.4	87.4	68.4	86.5	67.7	85.5	65.8	82.8	64.3

compared methods such as HaLRTC + LRANTD, TNCP and TNCP + LRANTD also achieve good results especially in the cases of MR$\leq$ 70%, these state-of-the-art methods cannot maintain good performance with increasing missing data (MR $>$ 70%). In this scenario, TDVM-CP achieves the best performance although it extracts only 10 features from each video sample. Moreover, TenALS and its combined "two-step" strategies fail to work on the higher-order dataset (Weizmann).

*3) **Face/Gait Clustering**:* For clustering tasks, we test on AR facial images ($55 \times 40 \times 1200$) and USF gait videos ($64 \times 44 \times 20 \times 731$). To measure the clustering performance, we adopt two metrics: Normalized Mutual Information (NMI) and Clustering Accuracy (ACC). Table III shows that TDVM-Tucker and TDVM-CP still outperform all other methods, including the four state-of-the-art clustering methods. Although these subspace clustering methods have shown good performance in the original papers, they do not achieve good results (and even fail to work on a few cases in which MR $\geq$ 70%) on these incomplete tensors probably because they are not applicable in this scenario (i.e., tensors with multi-block missing). Here, TDVM-Tucker achieves the best clustering results in all cases followed closely by TDVM-CP. In terms of NMI, TDVM-Tucker and TDVM-CP outperform the other methods by at least 36.7% and 33.5% on average on the AR dataset, and this improvement increases to over 40.5% and 37.5% on average on the USF gait database, respectively. In terms of ACC, the results of the 17 compared methods are worse than those of measuring in NMI, while TDVM-Tucker and TDVM-CP achieve much better clustering accuracy, particularly on the gait videos with 55.8% and 50.1% improvements on average, respectively.

4) Time Cost: We report the average time cost of feature extraction in Appendix B of the Supplementary Material. TDVM-Tucker is faster than the compared methods in most cases although our implementations are not optimized for efficiency as our focus here is accuracy. TDVM-CP is slightly slower than TDVM-Tucker because it requires more iterations to achieve convergence. Besides, HaLRTC, TNCP and TRPCA are more efficient than other existing methods, while certain two-step strategies such as TenALS + MPCA/LRANTD are very time consuming (more than 50 times slower than TDVM). Nevertheless, the efficiency can be improved, for example, by using sparse implementations in future work.

D. Summary of Experimental Results

- The proposed methods, TDVM-Tucker and TDVM-CP, outperform the 17 competing methods with 15.3% to 75.5% improvements in all cases of face recognition, object/action classification and face/gait clustering on six real-world image and video datasets. With more missing entries, our methods show more advantages with much better results than other methods. These results verify the effectiveness and superiority of incorporating low-rank tensor decomposition with feature variance maximization.
- TDVM-Tucker and TDVM-CP consistently achieve good results regardless of various multi-block missing settings and parameters. Besides, our methods also demonstrate its stability and superiority with respect to feature dimension reduction, benefitting from low-rank (low-dimensional) tensor decomposition. Although TDVM-CP yields the best results in fewer cases than TDVM-Tucker, it extracts low-dimensional vector features resulting in more dimensionality reduction. TDVM-CP, therefore, not only provides informative features but also reduces more time cost and memory space for further application (e.g., classification).
- HaLRTC, HaLRTC + LRANTD and HaLRTC + SOMPCARS are the best performing existing algorithms on the whole in Tables I, II and III respectively. TNCP

and TNCP + LRANTD also achieve the best results in some cases while TenALS shows much worse performance (and even fails to work) than HaLRTC and TNCP. Nevertheless, the proposed methods outperform these tensor completion methods significantly. That supports our prediction: tensor completion methods only focus on recovering missing data and do not explore the relationship among samples for effective feature extraction.

– Although a few "two-step" strategies show slight improvement in some cases, more "two-step" strategies (e.g., TNCP + SOMPCARS, TenALS + MPCA/SOMPCARS) perform worse than when using only the tensor completion methods (e.g., TNCP and TenALS) in most cases with high computational costs. That confirms our assumption: the reconstruction error from the completion step can deteriorate performance in feature extraction step, and "two-step" strategies usually work slowly. Moreover, although TRPCA is the state-of-the-art robust feature learning method for corrupted tensors, it does not perform well on these incomplete tensors as we predicted.
– Although SSC, SSC-CEC, ZF + SSC and SRME-MC have achieved good clustering results shown in the original papers, they are not applicable to these incomplete tensors with irregular missing patterns via multi-block missing settings, and even fail to work in a few cases in which MR $\geq 70\%$ as shown in Table III. This probably because these subspace clustering methods cannot handle this scenario as we discussed with the authors. In addition, these results also supports our claim: the MbM setting is more general and difficult than existing widely used pixel-based and block-based missing settings (e.g., used in the tensor completion and SSC methods), which is also verified in Sec. IV-B1.

V. Conclusion

In this paper, we have proposed two unsupervised methods, TDVM-Tucker and TDVM-CP, to solve the problem of feature extraction from incomplete tensors, based on orthogonal Tucker and CP decompositions, respectively. We first propose the TDVM approach which incorporates low-rank tensor decomposition with feature variance maximization into the unified framework. Focusing on orthogonal Tucker and CP decompositions, we have further proposed TDVM-Tucker which learns low-dimensional tensor features viewing the core tensors as features and TDVM-CP which extracts low-dimensional vector features viewing the weight vectors as features. TDVM-Tucker and TDVM-CP explore the relationship among data samples via feature variance maximization while estimating the missing entries via low-rank Tucker/CP approximation. We further discuss the generalization of the proposed methods by formulating the general model TDFR. Besides, we have developed the ADMM-BCD joint optimization scheme to solve the TDVM-CP model. Finally, we have evaluated our methods on six real-world image and video datasets with missing entries under the newly designed multi-block missing settings. Experimental results demonstrate that: the proposed methods not only stably yield similar good results under various MbM settings and different parameters on the whole, but also outperform the state-of-the-art methods with significant improvements in the applications of face recognition, object/action classification and face/gait clustering.

Acknowledgment

We thank Prof. René Vidal, Prof. Daniel Robinson and Dr. Rick Fan for their code sharing and helpful discussion. Qibin Zhao's work is supported by JSPS KAKENHI (Grant No. 17K00326) and NSFC China (Grant No. 61773129).

References

[1] I. T. Jolliffe, *Principal Component Analysis, second edition*. Springer Serires in Statistics, 2002.
[2] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.
[3] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "MPCA: Multilinear principal component analysis of tensor objects," *IEEE Trans. Neural Networks*, vol. 19, no. 1, pp. 18–39, 2008.
[4] G. Zhou, A. Cichocki, Q. Zhao, and S. Xie, "Efficient nonnegative Tucker decompositions: Algorithms and uniqueness," *TIP*, vol. 24, no. 12, pp. 4990–5003, 2015.
[5] X. Li, M. K. Ng, G. Cong, Y. Ye, and Q. Wu, "MR-NTD: Manifold regularization nonnegative Tucker decomposition for tensor data dimension reduction and representation," *TNNLS*, 2016.
[6] Q. Shi and H. Lu, "Semi-orthogonal multilinear PCA with relaxed start," in *IJCAI*. AAAI Press, 2015, pp. 3805–3811.
[7] L. R. Tucker, "Implications of factor analysis of three-way matrices for measurement of change," *Problems in Measuring Change*, vol. 15, pp. 122–137, 1963.
[8] R. A. Harshman, "Foundations of the PARAFAC procedure: models and conditions for an "explanatory" multi-modal factor analysis," *UCLA Working Papers in Phonetics*, pp. 1–84, 1970.
[9] J. D. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.
[10] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Uncorrelated multilinear principal component analysis for unsupervised multilinear subspace learning," *IEEE Trans. Neural Networks*, vol. 20, no. 11, pp. 1820–1836, 2009.
[11] C. Lu, J. Feng, Y. Chen, W. Liu, Z. Lin, and S. Yan, "Tensor robust principal component analysis: exact recovery of corrupted low-rank tensors via convex optimization," in *CVPR*, 2016, pp. 5249–5257.
[12] D. Williams, X. Liao, Y. Xue, L. Carin, and B. Krishnapuram, "On classification with incomplete data," *TPAMI*, vol. 29, no. 3, 2007.
[13] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.
[14] D. Williams, X. Liao, Y. Xue, and L. Carin, "Incomplete-data classification using logistic regression," in *ICML*. ACM, 2005, pp. 972–979.
[15] G. Doquire and M. Verleysen, "Feature selection with missing data using mutual information estimators," *Neurocomputing*, vol. 90, pp. 3–11, 2012.
[16] E. Hazan, R. Livni, and Y. Mansour, "Classification with low rank and missing data," in *ICML*, 2015, pp. 257–266.
[17] K. Lakshminarayan, S. A. Harp, R. Goldman, and T. Samad, "Imputation of missing data using machine learning techniques," in *Proceedings of the Second Int. Conf. on Knowledge Discovery and Data Mining*. AAAI Press, 1996, pp. 140–145.
[18] P. J. García-Laencina, J.-L. Sancho-Gómez, and A. R. Figueiras-Vidal, "Pattern classification with missing data: a review," *Neural Computing and Applications*, vol. 19, no. 2, pp. 263–282, 2010.
[19] H. Ozkan, O. S. Pelvan, and S. S. Kozat, "Data imputation through the identification of local anomalies," *TNNLS*, vol. 26, no. 10, pp. 2381–2395, 2015.
[20] Y. Liu, F. Shang, W. Fan, J. Cheng, and H. Cheng, "Generalized higher order orthogonal iteration for tensor learning and decomposition," *TNNLS*, vol. 27, no. 12, pp. 2551–2563, 2016.
[21] C. Yang, D. Robinson, and R. Vidal, "Sparse subspace clustering with missing entries," in *ICML*, 2015, pp. 2463–2472.

[22] J. Fan and T. W. Chow, "Sparse subspace clustering for data with missing entries and high-rank matrix completion," *Neural Networks*, vol. 93, pp. 36–44, 2017.

[23] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *FoCM*, vol. 9, no. 6, pp. 717–772, 2009.

[24] Q. Shi, H. Lu, and Y. M. Cheung, "Rank-one matrix completion with automatic rank estimation via l1-norm regularization," *TNNLS*, vol. 29, no. 10, pp. 4744–4757, 2018.

[25] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *TPAMI*, vol. 35, no. 1, pp. 208–220, 2013.

[26] Y. Liu, F. Shang, L. Jiao, J. Cheng, and H. Cheng, "Trace norm regularized CANDECOMP/PARAFAC decomposition with missing data," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2437–2448, 2015.

[27] P. Jain and S. Oh, "Provable tensor factorization with missing data," in *NIPS*, 2014, pp. 1431–1439.

[28] V. De Silva and L.-H. 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.

[29] T. T. Wu and K. Lange, "Matrix completion discriminant analysis," *Comput. Stat. Data Anal.*, vol. 92, pp. 115–125, 2015.

[30] C. Jia, G. Zhong, and Y. Fu, "Low-rank tensor learning with discriminant analysis for action classification and image recovery," in *AAAI*. AAAI Press, 2014, pp. 1228–1234.

[31] L. De Lathauwer, B. De Moor, and J. Vandewalle, "A multilinear singular value decomposition," *SIAM J. Matrix Anal. Appl.*, vol. 21, no. 4, pp. 1253–1278, 2000.

[32] Q. Shi, h. Lu, and Y. M. Cheung, "Tensor rank estimation and completion via CP-based nuclear norm," in *CIKM*. ACM, 2017, pp. 949–958.

[33] S. Boyd, "Alternating direction method of multipliers," in *NIPS Workshop on Optimization and Machine Learning*, 2011.

[34] Q. Shi, Y. M. Cheung, and Q. Zhao, "Feature extraction for incomplete data via low-rank Tucker decomposition," in *ECML-PKDD*. Springer, 2017, pp. 564–581.

[35] Q. Zhao, L. Zhang, and A. Cichocki, "Bayesian CP factorization of incomplete tensors with automatic rank determination," *TPAMI*, vol. 37, no. 9, pp. 1751–1763, 2015.

[36] W. Hu, D. Tao, W. Zhang, Y. Xie, and Y. Yang, "The twist tensor nuclear norm for video completion," *TNNLS*, vol. 28, no. 12, pp. 2961–2973, 2017.

[37] X. Chen, Z. Han, Y. Wang, Q. Zhao, D. Meng, L. Lin, and Y. Tang, "A generalized model for robust tensor factorization with noise modeling by mixture of gaussians," *TNNLS*, vol. PP, no. 99, pp. 1–14, 2018.

[38] Z. Fan, Y. Xu, W. Zuo, J. Yang, J. Tang, Z. Lai, and D. Zhang, "Modified principal component analysis: An integration of multiple similarity subspace models," *TNNLS*, vol. 25, no. 8, pp. 1538–1552, 2014.

[39] Y. Zhou, H. Lu, and Y. M. Cheung, "Bilinear probabilistic canonical correlation analysis via hybrid concatenations," in *AAAI*. AAAI Press, 2017, pp. 2949–2955.

[40] X. Peng, J. Lu, Z. Yi, and R. Yan, "Automatic subspace learning via principal coefficients embedding," *IEEE Trans. Cybern.*, vol. 47, no. 11, pp. 3583–3596, 2017.

[41] F. Ju, Y. Sun, J. Gao, Y. Hu, and B. Yin, "Vectorial dimension reduction for tensors based on bayesian inference," *TNNLS*, vol. PP, no. 99, pp. 1–14, 2017.

[42] Y. Zhou and Y. M. Cheung, "Probabilistic rank-one discriminant analysis via collective and individual variation modeling," *IEEE Trans. Cybern.*, DOI:10.1109/TCYB.2018.2870440.

[43] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *TPAMI*, vol. 35, no. 11, pp. 2765–2781, 2013.

[44] B. Eriksson, L. Balzano, and R. Nowak, "High-rank matrix completion," in *AISTATS*, 2012, pp. 373–381.

[45] E. Elhamifar, "High-rank matrix completion and clustering under self-expressive models," in *NIPS*, 2016, pp. 73–81.

[46] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *TPAMI*, vol. 31, no. 2, pp. 210–227, 2009.

[47] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *ICCV*, 2011, pp. 1615–1622.

[48] Y. Liu, F. Shang, W. Fan, J. Cheng, and H. Cheng, "Generalized higher-order orthogonal iteration for tensor decomposition and completion," in *NIPS*, 2014, pp. 1763–1771.

[49] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM J. on Optim.*, vol. 20, no. 4, pp. 1956–1982, 2010.

[50] F. Shang, Y. Liu, and J. Cheng, "Generalized higher-order tensor decomposition via parallel admm," in *AAAI*, 2014, pp. 1279–1285.

[51] N. Higham and P. Papadimitriou, "Matrix procrustes problems," 1995.

[52] W. Chu and Z. Ghahramani, "Probabilistic models for incomplete multi-dimensional arrays," in *AISTATS*. Citeseer, 2009.

[53] J. B. Denis and T. Dhorne, "Orthogonal tensor decomposition of 3-way tables," in *Multiway Data Analysis*, 1989, pp. 31–37.

[54] T. Zhang and G. H. Golub, "Rank-one approximation to high order tensors," *SIAM J. Matrix Anal. Appl.*, vol. 23, no. 2, pp. 534–550, 2001.

[55] G. Bergqvist and E. G. Larsson, "The higher-order singular value decomposition: theory and an application," *IEEE Signal Process. Mag.*, vol. 27, no. 3, pp. 151–154, 2010.

[56] A. Anandkumar, R. Ge, D. Hsu, S. M. Kakade, and M. Telgarsky, "Tensor decompositions for learning latent variable models." *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 2773–2832, 2014.

[57] S. Osher, Y. Mao, B. Dong, and W. Yin, "Fast linearized bregman iteration for compressive sensing and sparse denoising," *arXiv preprint arXiv:1104.0262*, 2011.

[58] B. Cao, L. He, X. Wei, M. Xing, P. S. Yu, H. Klumpp, and A. D. Leow, "t-BNE: tensor-based brain network embedding," in *Proceedings of the SIAM Int. Conf. on Data Mining*, 2017, pp. 189–197.

[59] J. Zhang, X. Li, P. Jing, J. Liu, and Y. Su, "Low-rank regularized heterogeneous tensor decomposition for subspace clustering," *IEEE Signal Processing Letters*, 2017.

[60] Z. Zhang, G. Ely, S. Aeron, N. Hao, and M. Kilmer, "Novel methods for multilinear data completion and de-noising based on tensor-SVD," in *CVPR*, 2014, pp. 3842–3849.

[61] I. V. Oseledets, "Tensor-train decomposition," *SIAM J. Sci. Comput.*, vol. 33, no. 5, pp. 2295–2317, 2011.

[62] 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," *TIP*, vol. 26, no. 5, pp. 2466–2479, 2017.

[63] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face-recognition algorithms," *TPAMI*, vol. 22, no. 10, pp. 1090–1104, 2000.

[64] K. Lee, J. Ho, and D. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *TPAMI*, vol. 27, no. 5, pp. 684–698, 2005.

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

[66] X. Peng, Z. Yu, Z. Yi, and H. Tang, "Constructing the L2-graph for robust subspace learning and subspace clustering," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 1053–1066, 2017.

[67] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri, "Actions as space-time shapes," in *ICCV*, vol. 2. IEEE, 2005, pp. 1395–1402.

[68] A. M. Martinez, "The AR face database," *CVC technical report*, 1998.

[69] S. Sarkar, P. Phillips, Z. Liu, I. R. Vega, P. Grother, and K. W.Bowyer, "The human ID gait challenge problem: data sets, performance, and analysis," *TPAMI*, vol. 27, no. 2, pp. 162–177, 2005.

[70] Z. Zhang and S. Aeron, "Exact tensor completion using t-svd," *TSP*, vol. 65, no. 6, pp. 1511–1526.

[71] D. Cai, "Litekmeans: the fastest matlab implementation of kmeans," *Software available at: http://www. zjucadcg. cn/dengcai/Data/Clustering. html*, 2011.