Feature Extraction for Incomplete Data via Low-rank Tucker Decomposition

Qiquan Shi[1], Yiu-ming Cheung[1**], and Qibin Zhao[2,3]

[1] Department of Computer Science, Hong Kong Baptist University, Hong Kong, China,
{csqqshi,ymc}@comp.hkbu.edu.hk,
[2] Tensor Learning Unit, Center for Advanced Intelligence Project (AIP), RIKEN, Japan.
[3] School of Automation, Guangdong University of Technology, China
qibin.zhao@riken.jp

Abstract. Extracting features from incomplete tensors is a challenging task which is not well explored. Due to the data with missing entries, existing feature extraction methods are not applicable. Although tensor completion techniques can estimate the missing entries well, they focus on data recovery and do not consider the relationships among tensor samples for effective feature extraction. To solve this problem of feature extraction for incomplete data, we propose an unsupervised method, **TDVM**, which incorporates *low-rank **T**ucker **D**ecomposition* with *feature **V**ariance **M**aximization* in a unified framework. Based on Tucker decomposition, we impose nuclear norm regularization on the core tensors while minimizing reconstruction errors, and meanwhile maximize the variance of core tensors (i.e., extracted features). Here, the relationships among tensor samples are explored via variance maximization while estimating the missing entries. We thus can simultaneously obtain lower-dimensional core tensors and informative features directly from observed entries. The alternating direction method of multipliers approach is utilized to solve the optimization objective. We evaluate the features extracted from two real data with different missing entries for face recognition tasks. Experimental results illustrate the superior performance of our method with a significant improvement over the state-of-the-art methods.

Keywords: Missing Data, Feature Extraction, Low-rank Tucker Decomposition, Variance Maximization

1 Introduction

This paper aims to extract features directly from data with missing entries. Many real-world data are multi-dimensional, in the form of *tensors*, which are ubiquitous such as multichannel images and have become increasingly popular [**?**]. Tucker decomposition is widely used to solve tensor learning problems, which decomposes a tensor into a core tensor with factor matrices [**?**]. Based on Tucker decomposition, many tensor methods are proposed for feature extraction (dimension reduction) [**?**,**?**,**?**,**?**,**?**]. For example, multilinear principal component analysis (MPCA) [**?**] extracts features directly from tensors, which is a popular extension of classical Principal Component Analysis

** Yiu-ming Cheung is the corresponding author.

(PCA) [**?**]. Furthermore, some robust methods such as robust tensor PCA (TRPCA) [**?**] are well studied for data with corruptions (e.g., noise and outliers) [**?**,**?**,**?**].

In practice, some entries of tensors are often missing due to the problems in the acquisition process or costly experiments etc. [**?**]. This missing data problem appears in a wide range of fields such as social sciences, computer vision and medical systems [**?**]. For example, partial responses in surveys are common in the social sciences, leading to incomplete datasets with arbitrary patterns [**?**]. Moreover, some images are corrupted during the image acquisition and partial entries are missing [**?**]. In these scenarios, the above existing feature learning methods cannot work well. How to correctly handle missing data is a fundamental yet challenging problem in machine learning [**?**], and the problem of *extracting features* from *incomplete tensors* is not well explored.

One natural solution to solving this problem is to recover the missing data and then view the recovered tensors as the extracted features. Tensor completion techniques are widely used for missing data problems and has drawn much attention in many applications such as image recovery [**?**] and video completion [**?**]. For example, a high accuracy low-rank tensor completion algorithm (HaLRTC) [**?**] is proposed to estimate missing values in tensors of visual data, and a generalized higher-order orthogonal iteration (gHOI) [**?**] achieves simultaneous low-rank Tucker decomposition and completion efficiently. Although these tensor completion methods can recover the missing entries well under certain conditions, they only focus on data recovery without exploring the relationships among samples for effective feature extraction. Besides, taking recovered data as features, the dimension of features cannot be reduced.

Another straightforward solution is a "two-step" strategy, i.e., "tensor completion methods + feature extraction methods": the missing entries are first recovered by the former and then the features are extracted from the completed data by the latter. For example, LRANTD [**?**] performs nonnegative Tucker decomposition (NTD) for incomplete tensors by realizing "low-rank representation (LRA) + nonnegative feature extraction". It needs a tensor completion method to estimate the missing values in the preceding LRA step. However, this "two-step" strategy probably amplifies the reconstruction errors as the missing entries and features are not learned in one stage, and the errors from tensor completion methods can deteriorate the performance of feature extraction in the succeeding step. Moreover, this approach is generally not computationally efficient.

Recently, a few works apply tensor completion methods to feature classification by incorporating completion model with discriminant analysis [**?**,**?**]. These methods are supervised and require labels which are expensive and difficult to obtain. To the best of our knowledge, there is no an unsupervised method to extract features directly from tensors with missing entries.

To solve the problem of extracting features from incomplete tensors, we propose an unsupervised method, i.e., incorporating *Low-rank **T**ucker **D**ecomposition* with *feature **V**ariance **M**aximization* in a unified framework, namely **TDVM**. In this framework, based on Tucker decomposition with orthonormal factor matrices (a.k.a., higher-order singular value decomposition (HOSVD) [**?**]), we impose nuclear norm regularization on the core tensors while minimizing the reconstruction error, and meanwhile maximize the variance of core tensors. In this paper, the learned *core tensors* (analogous to the

singular values of a matrix) are viewed as the *extracted features*. Compared with tensor completion methods and "two-step" strategies:

- Although Tucker decomposition-based tensor completion methods can also obtain core tensors, these core tensors are learned with aiming to recover the tensor samples and without exploring the relationships among samples for effective feature extraction. Unlike these tensor completion methods, here we focus on low-dimensional feature extraction rather than missing data recovery. Besides, we incorporate a specific term (*feature variance maximization*) to enhance the discriminative properties of learned core tensors.
- Different from the "two-step" strategies, we simultaneously learn the missing entries and features directly from observed entries in the unified framework. Besides, TDVM directly learns low-dimensional features in one step, which saves computational cost.

We optimize our model using alternating direction method of multipliers (ADMM) [**?**]. After feature extraction, we evaluate the extracted features for face recognition, which empirically demonstrates that TDVM outperforms the competing methods consistently. In a nutshell, the contributions of this paper are twofold:

- We propose an efficient unsupervised feature extraction method, TDVM, based on low-rank Tucker decomposition. TDVM can simultaneously obtain low-dimensional core tensors and features for incomplete data.
- We incorporate nuclear norm regularization with variance maximization on core tensors (features) to explore the relationships among tensor samples while estimating missing entries, leading to informative features extracted directly from observed entries.

2 Preliminaries and Backgrounds

2.1 Notations and Operations

The number of dimensions of a tensor is the *order* and each dimension is a *mode* of it. A vector (i.e. first-order tensor) is denoted by a bold lower-case letter $\mathbf{x} \in \mathbb{R}^I$. A matrix (i.e. second-order tensor) 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 $\|\mathcal{X}\|_F = \langle \mathcal{X}, \mathcal{X} \rangle^{1/2}$[**?**]. $\boldsymbol{\Omega} \in \mathbb{R}^{I_1 \times I_2}$ is a binary index set: $\boldsymbol{\Omega}_{i_1,\cdots,i_N} = 1$ if $\mathcal{X}_{i_1,\cdots,i_N}$ is observed, and $\boldsymbol{\Omega}_{i_1,\cdots,i_N} = 0$ otherwise. $\mathcal{P}_{\boldsymbol{\Omega}}$ is the associated sampling operator which acquires only the entries indexed by $\boldsymbol{\Omega}$, defined as:

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

where $\boldsymbol{\Omega}^c$ is the complement of $\boldsymbol{\Omega}$, and $\mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{X}) + \mathcal{P}_{\boldsymbol{\Omega}^c}(\mathcal{X}) = \mathcal{X}$.

Definition 1 Mode-n Product. *A mode-n product between a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and a matrix/vector $\mathbf{U} \in \mathbb{R}^{I_n \times J_n}$ is denoted by $\mathcal{Y} = \mathcal{X} \times_n \mathbf{U}^T$. The size of $\mathcal{Y}$ is $I_1 \times \cdots \times I_{n-1} \times J_n \times I_{n+1} \times \cdots \times I_N$, 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 we have $\mathbf{Y}_{(n)} = \mathbf{U}^T \mathbf{X}_{(n)}$[?].*

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

2.2 Tucker Decomposition

A tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is represented as a core tensor with factor matrices in Tucker decomposition model [**?**]:

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

where $\{\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times R_n}, n = 1, 2 \cdots N, \text{ and } R_n < I_n\}$ are factor matrices with orthogonal columns and $\mathcal{G} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$ is the core tensor with smaller dimension. Tucker-rank of an Nth-order tensor $\mathcal{X}$ is an N-dimensional vector, denoted as $(R_1, \cdots, R_N)$, where R_N is the rank of the mode-n unfolded matrix $\mathbf{X}_{(n)}$ of $\mathcal{X}$. Figure 1 illustrates this decomposition. In this paper, we regard the *core tensor* consists of the *extracted features* of a tensor.

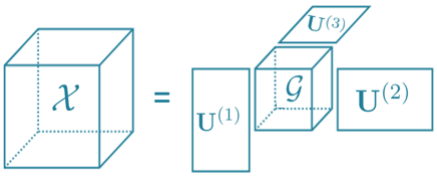


Fig. 1. The Tucker decomposition of tensors (a third-order tensor $\mathcal{X}$ shown for illustration).

3 Feature Extraction for Incomplete Data

3.1 Problem Definition

Given 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 $\mathcal{T}$. For feature extraction (dimension reduction), we aim to directly extract low-dimensional features $\mathcal{G} = [\mathcal{G}_1, \cdots, \mathcal{G}_m, \cdots \mathcal{G}_M] \in \mathbb{R}^{R_1 \times \cdots \times R_N \times M} (R_n < I_n, n = 1, \cdots, N)$ from the given high-dimensional incomplete tensors $\mathcal{T}$.

***Remark**:* This problem is different from the case of data with corruptions (e.g., noise and outliers) widely studied in [**?**,**?**,**?**,**?**]: Only if the corruptions are arbitrary, missing data could be regarded as a special case of corruptions (with the location of corruption being known). However, the magnitudes of corruptions in reality are not arbitrarily large. In other words, here we study a new problem and existing feature extraction methods are not applicable.

3.2 Formulation of the Proposed Method: TDVM

To solve this problem, we propose an unsupervised feature extraction method. Based on Tucker decomposition, we impose the nuclear norm on the core tensors of observed tensors while minimizing reconstruction errors, 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*. Thus, the objective function of TDVM is:

$$\begin{aligned}\min_{\mathcal{X}_m,\mathcal{G}_m,\mathcal{S}_m,\mathbf{U}^{(n)}} & \sum_{m=1}^{M}\frac{1}{2}\|\mathcal{X}_m-\mathcal{G}_m\times_1\mathbf{U}^{(1)}\cdots\times_N\mathbf{U}^{(N)}\|_F^2 \\ & +\sum_{m=1}^{M}\|\mathcal{G}_m\|_*-\frac{1}{2}\sum_{m=1}^{M}\|\mathcal{G}_m-\bar{\mathcal{G}}\|_F^2, \\ \text{s.t. } & \mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{X}_m)=\mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{T}_m), {\mathbf{U}^{(n)}}^\top\mathbf{U}^{(n)}=\mathbf{I}.\end{aligned} \tag{3}$$

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. $\mathcal{G}_m$ is the core tensor which consists of the *extracted features* (analogous to the *singular values* of a matrix) of an incomplete tensor $\mathcal{T}_m$ with observed entries in $\boldsymbol{\Omega}$. $\|\mathcal{G}_m\|_*$ is the nuclear norm of $\mathcal{G}_m$ (i.e., the summation of the singular values of the unfolded matrices along modes of $\mathcal{G}_m$ [**?**]). $\bar{\mathcal{G}}=\frac{1}{M}\sum_{m=1}^M\mathcal{G}_m$ is the mean of core tensors (extracted features).

***Remark**:* Our objective function (3) integrates three terms into a unified framework:

- The first term: minimizing $\sum_{m=1}^M\frac{1}{2}\|\mathcal{X}_m-\mathcal{G}_m\times_1\mathbf{U}^{(1)}\cdots\times_N\mathbf{U}^{(N)}\|_F^2$, aims to minimize the reconstruction error based on given observed entries.
- The second term: minimizing $\sum_{m=1}^M\|\mathcal{G}_m\|_*$, aims to obtain low-dimensional features. It is proved that imposing the nuclear norm on a core tensor $\mathcal{G}_m$ is essentially equivalent to that on its original tensor $\mathcal{X}_m$ [**?**]. We thus obtain a low-rank solution, i.e. R_n can be small ($R_n<I_n$). Thus, the learned feature subspace is naturally low-dimensional. Besides, imposing nuclear norm on core tensors $\mathcal{G}_m$ instead of original $\mathcal{X}_m$ saves computational cost.
- The third term: minimizing $-\sum_{m=1}^M\frac{1}{2}\|\mathcal{G}_m-\bar{\mathcal{G}}\|_F^2$, is equivalent to maximize the variance of extracted features (core tensors) following PCA. We thus explore the relationships of incomplete tensors via *variance maximization* while estimating the missing entries via the first and second term (*low-rank Tucker decomposition*).

By this unified framework, we can efficiently extract low-dimensional informative features directly from observed entries, which is different from *tensor completion methods* (only focusing on data recovery without considering the relationships among samples for effective feature extraction) and *"two-step" strategies* (the reconstruction error from tensor completion step probably deteriorates the performance of feature extraction in the succeeding step, and combining two methods is generally time consuming).

3.3 Optimization by ADMM

To optimize (3) using ADMM, we apply the variable splitting technique and introduce a set of *auxiliary variables* $\{\mathcal{S}_m\in\mathbb{R}^{R_1\times\cdots\times R_N}, m=1\cdots M, n=1,\cdots,N\}$, and then reformulate (3) as:

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

For easy derivation of (4), we reformulate it by unfolding each tensor variable along mode-n and absorbing the constraints. Thus, we get the Lagrange function as follows:

$$\begin{aligned}\mathcal{L}=&\sum_{m=1}^{M}\sum_{n=1}^{N}\Big(\frac{1}{2}\|\mathbf{X}_m^{(n)}-\mathbf{U}^{(n)}\mathbf{G}_m^{(n)}{\mathbf{P}^{(n)}}^\top\|_F^2+\|\mathbf{S}_m^{(n)}\|_* \\ &+\langle\mathbf{Y}_{mn},\mathbf{G}_m^{(n)}-\mathbf{S}_m^{(n)}\rangle+\frac{\mu}{2}\|\mathbf{G}_m^{(n)}-\mathbf{S}_m^{(n)}\|_F^2-\frac{1}{2}\|\mathbf{G}_m^{(n)}-\bar{\mathbf{G}}^{(n)}\|_F^2\Big)\end{aligned} \tag{5}$$

where $\mathbf{P}^{(n)}=\mathbf{U}^{(N)}\bigotimes\cdots\bigotimes\mathbf{U}^{(n+1)}\bigotimes\mathbf{U}^{(n-1)}\cdots\bigotimes\mathbf{U}^{(1)}\in\mathbb{R}^{\prod_{j\neq n}I_j\times\prod_{j\neq n}R_j}$ and $\{\mathbf{Y}_{mn}\in\mathbb{R}^{R_n\times\prod_{j\neq n}R_j},n=1,\cdots,N,m=1,\cdots,M\}$ are the matrices of Lagrange multipliers. $\mu>0$ is a penalty parameter. $\mathbf{X}_m^{(n)}\in\mathbb{R}^{I_n\times\prod_{j\neq n}I_j}$ and $\{\mathbf{G}_m^{(n)},\mathbf{S}_m^{(n)},\bar{\mathbf{G}}^{(n)}\}\in\mathbb{R}^{R_n\times\prod_{j\neq n}R_j}$ are the mode-n unfolded matrices of tensor $\mathcal{X}_m$ and {core tensor $\mathcal{G}_m$, auxiliary variable $\mathcal{S}_m$, mean of features $\bar{\mathcal{G}}$}, respectively.

ADMM solves the problem (5) by successively minimizing $\mathcal{L}$ over $\{\mathbf{X}_m^{(n)},\mathbf{G}_m^{(n)},\mathbf{S}_m^{(n)},\mathbf{U}^{(n)}\}$, and then updating $\mathbf{Y}_{mn}$.

Update $\mathbf{S}_m^{(n)}$ The Lagrange function (5) with respect to $\mathbf{S}_m^{(n)}$ is,

$$\mathcal{L}_{\mathbf{S}_m^{(n)}}=\sum_{m=1}^{M}\sum_{n=1}^{N}\big(\|\mathbf{S}_m^{(n)}\|_*+\frac{\mu}{2}\|(\mathbf{G}_m^{(n)}+\mathbf{Y}_{mn}/\mu)-\mathbf{S}_m^{(n)}\|_F^2\big). \tag{6}$$

To solve (6), we use the spectral soft-thresholding operation [**?**] to update $\mathbf{S}_m^{(n)}$:

$$\mathbf{S}_m^{(n)}=prox_{1/\mu}(\mathbf{G}_m^{(n)}+\mathbf{Y}_{mn}/\mu)=\mathbf{U}\text{diag}(\max\sigma-\frac{1}{\mu},0)\mathbf{V}^\top, \tag{7}$$

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

Update $\mathbf{U}^{(n)}$ The Lagrange function (5) with respect to $\mathbf{U}^{(n)}$ is:

$$\mathcal{L}_{\mathbf{U}^{(n)}}=\sum_{m=1}^{M}\sum_{n=1}^{N}\frac{1}{2}\|\mathbf{X}_m^{(n)}-\mathbf{U}^{(n)}\mathbf{G}_m^{(n)}{\mathbf{P}^{(n)}}^\top\|_F^2,\ \text{ s.t. }\ {\mathbf{U}^{(n)}}^\top\mathbf{U}^{(n)}=\mathbf{I}, \tag{8}$$

According to the Theorem 4 in [**?**], the minimization of the problem (8) over the matrices $\{\mathbf{U}^{(1)},\cdots,\mathbf{U}^{(N)}\}$ having orthonormal columns is equivalent to the maximization of the following problem:

$$\mathbf{U}^{(n)}=\arg\max\ \text{trace}\big({\mathbf{U}^{(n)}}^\top\mathbf{X}_m^{(n)}(\mathbf{G}_m^{(n)}{\mathbf{P}^{(n)}}^\top)^\top\big) \tag{9}$$

where trace() is the trace of a matrix, and we denote $\mathbf{W}^{(n)}=\mathbf{G}_m^{(n)}{\mathbf{P}^{(n)}}^\top$.

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

Input: Incomplete tensors $\mathcal{T} = [\mathcal{T}_1, \cdots, \mathcal{T}_m, \cdots \mathcal{T}_M]$, $\boldsymbol{\Omega}$, μ, and the maximum iterations K, the dimension of core tensors (features) $D = [R_1, \cdots, R_N]$, and stopping tolerance *tol*.
Initialization: Set $\mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{T}_m), \mathcal{P}_{\boldsymbol{\Omega}^c}(\mathcal{X}_m) = \mathbf{0}, m = 1, \cdots, M$; initialize $\{\mathcal{G}_m\}_{m=1}^M$ and $\{\mathbf{U}^{(n)}\}_{n=1}^N$ randomly; $\mu_0 = 5, \rho = 10, \mu_{max} = 1e10$.
for $m = 1$ **to** M **do**
  for $k = 1$ **to** K **do**
    for $n = 1$ **to** N **do**
      Update $\mathbf{S}_m^{(n)}$, $\mathbf{U}^{(n)}$ and $\mathbf{G}_m^{(n)}$ by (7), (10) and (12) respectively.
      Update $\mathbf{Y}_m^{(n)}$ by $\mathbf{Y}_m^{(n)} = \mathbf{Y}_m^{(n)} + \mu(\mathbf{G}_m^{(n)} - \mathbf{S}_m^{(n)})$.
    end for
    Update $\mathcal{X}_m$ by (13).
    Update $\mu_{k+1} = \min(\rho\mu_k, \mu_{max})$.
  end for
  If $\|\mathcal{G}_m - \mathcal{S}_m\|_F^2 / \|\mathcal{G}_m\|_F^2 < tol$, break; otherwise, continue.
end for
Output: Extracted features (core tensors): $\mathcal{G} = [\mathcal{G}_1, \cdots, \mathcal{G}_m, \cdots \mathcal{G}_M]$.

The problem (9) is actually the well-known orthogonal procrustes problem [**?**], whose global optimal solution is given by the SVD of $\mathbf{X}_m^{(n)}{\mathbf{W}^{(n)}}^\top$, i.e.,

$$\mathbf{U}^{(n)} = \hat{\mathbf{U}}^{(n)}(\hat{\mathbf{V}}^{(n)})^\top, \tag{10}$$

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.

Update $\mathbf{G}_m^{(n)}$ The Lagrange function (5) with respect to $\mathbf{G}_m^{(n)}$ is:

$$\begin{aligned}\mathcal{L}_{\mathbf{G}_m^{(n)}} = \|\mathbf{X}_m^{(n)} - \mathbf{U}^{(n)}\mathbf{G}_m^{(n)}{\mathbf{P}^{(n)}}^\top\|_F^2 + \frac{\mu}{2}\|\mathbf{G}_m^{(n)} - \mathbf{S}_m^{(n)} + \mathbf{Y}_{mn}/\mu\|_F^2 \\ - \frac{1}{2}\|(1 - \frac{1}{M})\mathbf{G}_m^{(n)} - \frac{1}{M}\sum_{j\neq m}^{M}\mathbf{G}_j^{(n)}\|_F^2\Big).\end{aligned} \tag{11}$$

Then we set the partial derivative $\frac{\partial \mathcal{L}_{\mathbf{G}_m^{(n)}}}{\partial \mathbf{G}_m^{(n)}}$ to zero, and get:

$$\begin{aligned}\mathbf{G}_m^{(n)} = \frac{M^2}{M^2\mu + 2M - 1}\Big(\mu\mathbf{S}_m^{(n)} - \mathbf{Y}_{mn} + {\mathbf{U}^{(n)}}^\top\mathbf{X}_m^{(n)}\mathbf{P}^{(n)}\Big) \\ - \big((\frac{1}{M} - \frac{1}{M^2})\sum_{j\neq m}^{M}\mathbf{G}_j^{(n)}\big).\end{aligned} \tag{12}$$

Update $\mathcal{X}_m$ The Lagrange function (5) with respect to $\mathcal{X}$ is:

$$\begin{aligned}\mathcal{L}_{\mathcal{X}_m} = \frac{1}{2}\sum_{m=1}^{M}\|\mathcal{X}_m - \mathcal{G}_m \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}\|_F^2, \\ \text{s.t. } \mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{X}_m) = \mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{T}_m),\end{aligned} \tag{13}$$

By deriving the Karush-Kuhn-Tucker (KKT) conditions for function (13), we can update $\mathcal{X}_m$ by $\mathcal{X}_m = \mathcal{P}_{\boldsymbol{\Omega}}(\mathcal{X}_m) + \mathcal{P}_{\boldsymbol{\Omega}^c}(\mathcal{Z}_m)$, where $\mathcal{Z}_m = \mathcal{G}_m \times_1 \mathbf{U}^{(1)} \cdots \times_N \mathbf{U}^{(N)}$.

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

3.4 Complexity Analysis

We analyze the complexity of TDVM following [**?**]. For simplicity, we assume the size of tensor is $I_1 = \cdots = I_N = I$, and the feature dimensions are $R_1 = \cdots = R_N = R$. At each iteration, the time complexity of performing the soft-thresholding operator (7) is $O(MNR^{N+1})$. The time complexities of some multiplication operators in (10)/(12) and (13) are $O(MNRI^N)$ and $O(MRI^N)$, respectively. Hence, the total time complexity of TDVM is $O(M(N+1)RI^N)$ per iteration.

4 Experimental Results

We implemented TDVM[4] in MATLAB and all experiments were performed on a PC (Intel Xeon(R) 4.0GHz, 64GB).

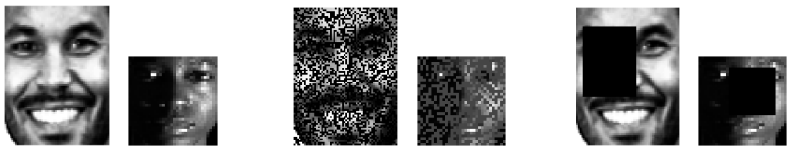
4.1 Experimental Setup

Compared methods: We compare our TDVM with *nine* methods in three categories:

- *Two* tensor completion methods based on Tucker-decomposition: **HaLRTC** [**?**] and **gHOI** [**?**]. The recovered tensor are regarded as the features.
- *Six* {tensor completion methods + feature extraction methods} (i.e., "two-step" strategies): HaLRTC + **PCA** [**?**], gHOI + **PCA**, HaLRTC + **MPCA** [**?**], gHOI + **MPCA**, HaLRTC + **LRANTD** [**?**] and gHOI + **LRANTD**.
- *One* robust tensor feature learning method: **TRPCA** [**?**].

After feature extraction stage, we use the Nearest Neighbors Classifier (**NNC**) to evaluate the extracted features on two real data for face recognition. We had also evaluated TDVM on MNIST handwritten digits [**?**] for object classification, and TDVM obtains the best results in all cases. We do not report here due to limited space.

Data: We evaluate the proposed TDVM on two real data for face recognition tasks. One is a subset of Facial Recognition Technology database (FERET)[5] [**?**], which has 721 face samples from 70 subjects. Each subject has 8-31 faces with at most 15 degrees of pose variation and each face image is normalized to a 80×60 gray image. The other is a subset of extended Yale Face Database B (YaleB)[6] [**?**], which has 2414 face samples from 38 subjects. Each subject has 59-64 near frontal images under different illuminations and each face image is normalized to a 32×32 gray image.



(a) (b) (c)

Fig. 2. One example of (a) original images of FERET and YaleB with (b) 50% pixel-based and with (c) 40×30 and 16×16 block-based missing entries, respectively.

[4] Codes and data: `https://www.dropbox.com/sh/h4k07sstdmthd80/AABMPFEqDDz-NzKWXIhDnLL0a/Qiquan_TDVM(ECML_198)?dl=0`

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

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

Missing data settings: We set the tensors with two types of missing data:

- **Pixel-based missing**: we uniformly select $10\% - 90\%$ pixels (entries) of tensors as missing at random. Pixel-based missing setting is widely used in tensor completion domain. One example (e.g., missing 50% entries) is shown in Fig. 2(b).
- **Block-based missing**: we randomly select $B_1 \times B_2$ block pixels of each tensor sample as *missing*. The missing block is random in each sample. One example (e.g., $\{B_1 = 40, B_2 = 30\}$ for FERET and $\{B_1 = 16, B_2 = 16\}$ for YaleB) is shown in Fig. 2(c). In practice, some parts of a face can be covered by some objects such as a sunglass, which can be regarded as the block-based missing case.

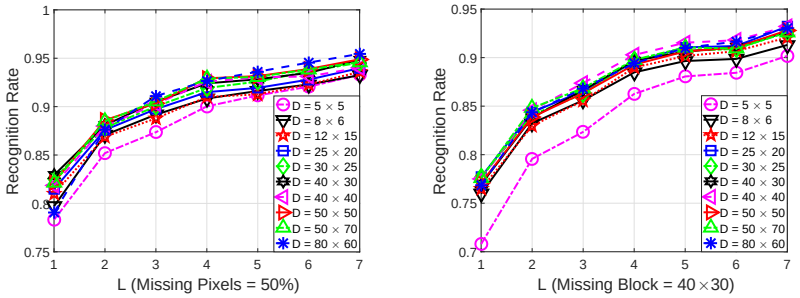
Intuitively, handling data with *block-based missing* is more difficult than that with *pixel-based missing* if same number of entries are missing.

Parameter settings: We set the maximum iterations $K = 200, tol = 1e-5$ for all methods and set the feature dimension $D = R_1 \times R_2 = \{40 \times 30, 16 \times 16\}$ for TDVM, gHOI and LRANTD on $\{$FERET, YaleB$\}$ respectively. In other words, we directly learn $40 \times 30 \times 721$ features from FERET ($80 \times 60 \times 721$) and extract $16 \times 16 \times 2414$ features from YaleB ($32 \times 32 \times 2414$). Other parameters of the compared methods have followed the original papers.

Applying extracted features for face recognition using NNC, we randomly select $L = \{1, 2, \cdots, 7\}$ extracted feature samples from each subject (with 8-31 samples) of FERET for training in NNC. On YaleB, we randomly select $L = \{5, 10, \cdots, 50\}$ extracted feature samples from each subject (with 59-64 samples) for training.

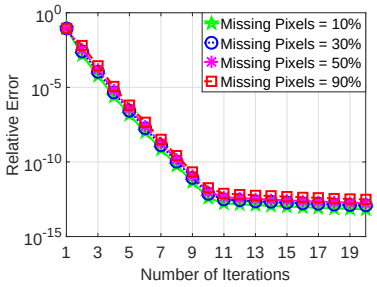
4.2 Parameter Sensitivity and Convergence Study

Effect of feature dimension D: We study the effect of TDVM with different feature dimensions (size of each core tensor) for face recognition on FERET. We set the feature dimension D of each face sample as $R_1 \times R_2$ in TDVM and show the corresponding face recognition results. Figure 3 shows that TDVM with different feature dimensions stably yields similar recognition results on FERET in both pixel-based and block-based missing cases, excepting $D = 5 \times 5$ (i.e., only 25 features are extracted from each 80×60 face image) where the number of features are too limited to achieve good results. Since a larger D costs more time and we aim to learn low-dimensional features, here we set $D = R_1 \times R_2 = 40 \times 30$ and 16×16 for TDVM to extract features from FERET and YaleB, respectively.

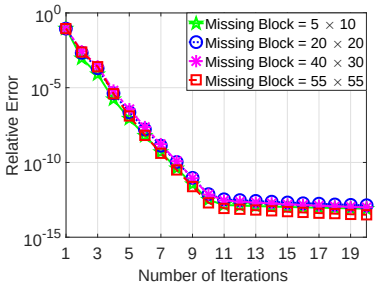


(a) FERET with 50% missing pixels (b) FERET with 40×30 missing block

Fig. 3. Recognition results on FERET via TDVM with different feature dimension Ds.

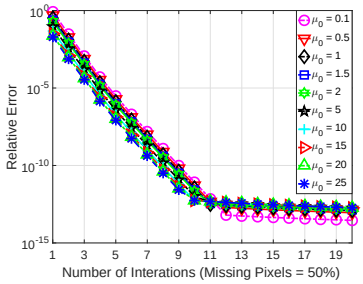


(a) FERET with pixel-based missing entries

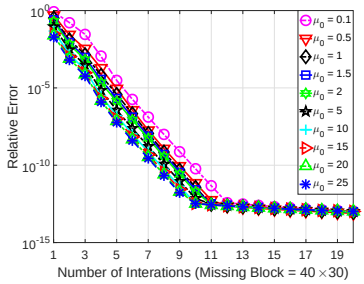


(b) FERET with block-based missing entries

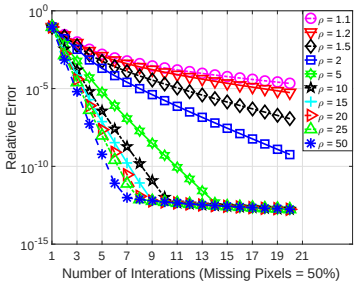
Fig. 4. Convergence curves of TDVM in terms of *Relative Error*: $\|\mathcal{G}_m - \mathcal{S}_m\|_F^2/\|\mathcal{G}_m\|_F^2$ on FERET with (a) pixel-based / (b) block-based missing entries.



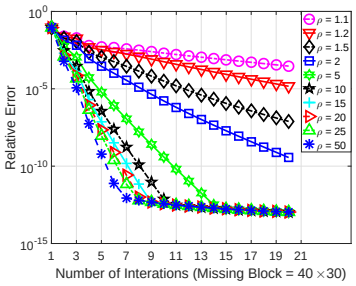
(a) TDVM with different μ_0 (fix $\rho = 10$)



(b) TDVM with different μ_0 (fix $\rho = 10$)



(c) TDVM with different ρ (fix $\mu_0 = 5$)

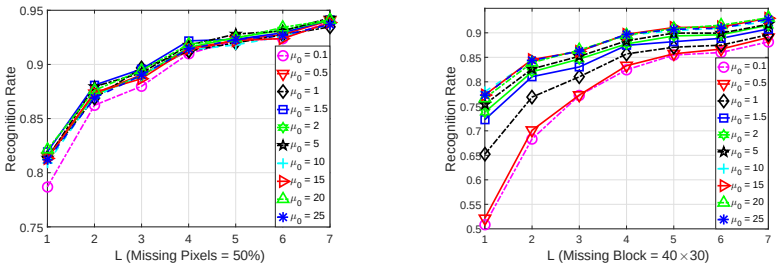


(d) TDVM with different ρ (fix $\mu_0 = 5$)

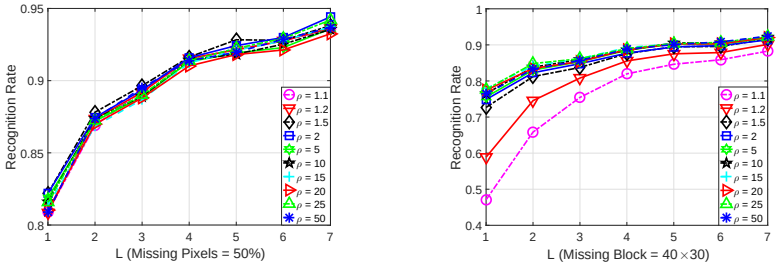
Fig. 5. Convergence curves of feature extraction on FERET with pixel-based (50%) / block-based (40×30) missing entries via TDVM with ten different values of μ_0 and ρ, respectively.

Convergence: We study the convergence of TDVM in terms of *Relative Error*: $\|\mathcal{G}_m - \mathcal{S}_m\|_F^2/\|\mathcal{G}_m\|_F^2$ on FERET with pixel/block-based missing entries. Here, we set $\mu_0 = 5$ and $\rho = 10$ for TDVM. Figure 4 shows that the relative error dramatically decreases to a very small value (around 10^{-13} order) with about 10 iterations. In other words, the proposed TDVM converges fast within 5 iterations if we set $tol = 1e-5$.

Sensitivity analysis of parameter μ_0 and ρ : In line 10 of Algorithm 1, we iteratively update the penalty parameter μ with a step size ρ from an initial μ_0, which has been widely used in many methods such as [**?**] and makes the algorithm converges faster.



(a) TDVM with different μ_0 (fix $\rho = 10$) (b) TDVM with different μ_0 (fix $\rho = 10$)



(c) TDVM with different ρ (fix $\mu_0 = 5$) (d) TDVM with different ρ (fix $\mu_0 = 5$)

Fig. 6. Recognition results on FERET with pixel-based (50%) / block-based (40×30) missing entries via TDVM with ten different values of μ_0 and ρ, respectively.

Figures 5 and 6 show the convergence curves and corresponding recognition results on FERET with 50% missing pixels and 40×30 missing block via TDVM with different μ_0 and ρ respectively. As seen from Fig. 5(a) and 5(b), with different μ_0, TDVM stably converges to a small value (around 10^{-13} order) with around 10 iterations. In terms of ρ, the relative errors converge to a small value faster if TDVM with a larger ρ (e.g., $\rho = 10$), as shown in Fig. 5(c) and 5(d). Figures 6(a) and 6(c) show that the feature extraction performance of our TDVM is stable and not sensitive to the values of μ_0 and ρ on FERET with 50% missing pixels. Besides, as seen from Fig. 6(b) and 6(d), with a larger μ_0 ($\mu_0 > 1$) and ρ ($\rho > 1.5$) for TVDM on FERET with 40×30 block-based missing entries, the corresponding face recognition results are similar and stable.

In general, we do not need to carefully tune the parameter μ_0 and ρ for the proposed TDVM. In this paper, we set $\mu_0 = 5$ and $\rho = 10$ in Algorithm 1 for all tests.

4.3 Evaluate Features Extracted from Data with Pixel/Block-based Missing

To save space, for pixel-based missing case, we report the results of FERET and YaleB with $\{10\%, 30\%, 50\%, 90\%\}$ missing pixels in **Table** 1. For block-based missing case, we report the results of FERET with $\{5 \times 10,\ 20 \times 20,\ 40 \times 30,\ 55 \times 55\}$ missing block and YaleB with $\{5 \times 5,\ 8 \times 10,\ 16 \times 16,\ 30 \times 25\}$ missing block in **Table** 2 respectively. In each pixel/block-based missing case, we report the recognition rates of randomly selecting $L = \{1, 7\}$ and $L = \{5, 50\}$ extracted feature samples from each subject of FERET and YaleB for training in NNC, respectively. We highlight the **best** results in **bold** fonts and second best in underline respectively. We repeat the runs 10 times of *feature extraction* and of *recognition* separately, and report the average results.

Table 1. Face recognition results (average recognition rates %) on the FERET and YaleB with $\{10\%, 30\%, 50\%, 90\%\}$ pixel-based missing entries.

Data	FERET (Image size 80×60)								YaleB (Image size 32×32)							
Missing Pixels	10%		30%		50%		90%		10%		30%		50%		90%	
L	1	7	1	7	1	7	1	7	5	50	5	50	5	50	5	50
HaLRTC	36.44	73.59	36.50	73.07	35.75	72.42	21.75	44.03	35.85	76.71	34.78	75.08	31.55	71.77	12.87	31.83
gHOI	36.41	73.94	36.52	73.94	36.53	74.07	29.65	63.72	35.42	76.38	33.40	73.74	28.68	66.91	10.51	19.82
HaLRTC + PCA	32.23	69.74	26.37	63.38	25.56	59.91	6.84	9.05	22.22	51.26	20.70	48.00	16.47	36.63	13.75	32.59
gHOI + PCA	32.24	68.96	28.74	63.51	26.41	61.26	13.90	24.68	29.71	51.79	15.63	36.69	18.42	41.96	4.95	6.75
HaLRTC + MPCA	40.49	75.37	40.08	73.68	38.94	72.77	4.79	7.62	30.23	73.07	29.62	71.36	28.57	69.34	14.45	38.62
gHOI + MPCA	41.18	75.97	41.01	75.89	40.14	74.68	8.76	14.16	30.02	72.59	27.37	68.07	13.71	43.25	4.79	6.17
HaLRTC + LRANTD	34.85	73.64	34.64	72.60	33.89	71.86	15.73	27.88	23.13	55.93	22.03	52.53	20.52	48.79	9.53	22.55
gHOI + LRANTD	36.44	74.11	36.44	74.07	36.68	74.33	30.06	64.42	22.66	54.30	21.64	52.20	21.28	50.33	10.06	20.53
TRPCA	51.27	84.33	46.24	82.08	41.71	79.09	4.61	10.39	32.66	71.83	29.51	68.72	22.56	53.99	2.65	2.72
TDVM	**85.50**	**96.23**	**84.62**	**95.58**	**82.01**	**94.59**	**58.39**	**79.57**	**93.21**	**98.40**	**92.28**	**98.91**	**91.93**	**98.13**	**87.15**	**97.20**

Face recognition results on FERET/YaleB with pixel-based missing TDVM outperforms the other nine methods by $\{34.69\%, 35.72\%, 35.19\%, 46.65\%\}$ in all cases of **FERET** with $\{10\%, 30\%, 50\%, 90\%\}$ missing pixels on average respectively, as shown in the left half of Table 1. Besides, TRPCA achieves the second best results in six cases given more than 50% observations while its performance drops dramatically when missing 90% pixels, where the gHOI + LRANTD takes the second place. Moreover, with less training features (e.g. $L = 1$) in NNC, our TDVM has more advantage as it aims to extract low-dimensional informative features.

The right half of Table 1 shows that TDVM outperforms the best performing existing algorithm (HaLRTC) in all cases of **YaleB** with $\{10\%, 30\%, 50\%\}$ missing pixels by $\{39.52\%, 40.67\%, 43.37\%\}$ on average, respectively. When the missing rate achieves 90%, the performance of compared methods drop sharply, excepting HaLRTC + MPCA which wins other existing methods in this scenario, where our TDVM keeps the best performance with 77.45% over all the existing methods.

Face recognition results on FERET/YaleB with block-based missing The left half of Table 2 shows that TDVM outperforms all competing methods by $\{35.99\%, 37.70\%, 44.48\%, 50.68\%\}$ in all cases of **FERET** with $\{5 \times 10,\ 20 \times 20,\ 40 \times 30,\ 55 \times 55\}$ missing blocks on average, respectively. Furthermore, gHOI/HaLRTC + MPCA and HaLRTC share the second place in these cases.

As shown in the right half of Table 2: TDVM outperforms the nine state-of-the-art methods by $\{40.53\%, 40.40\%, 46.07\%, 50.42\%\}$ in all cases of **YaleB** with $\{5 \times 5,\ 8 \times 10,\ 16 \times 16,\ 30 \times 25\}$ missing blocks on average, respectively. Specifically, HaLRTC is the best performing existing algorithm in the cases of missing $\{8 \times 10,\ 16 \times 16,\ 30 \times 25\}$ block, but our TDVM outperforms it by $\{29.55\%, 30.31\%, 30.02\%\}$ respectively there.

Table 2. Face recognition results (average recognition rates %) on the FERET and YaleB with block-based missing entries.

Data	FERET (Image size 80×60)								YaleB (Image size 32×32)							
Missing Block	5×10		20×20		40×30		55×55		5×5		8×10		16×16		30×25	
L	1	7	1	7	1	7	1	7	5	50	5	50	5	50	5	50
HaLRTC	36.37	73.64	35.75	72.77	33.96	71.21	27.24	59.31	36.02	76.73	35.27	76.30	33.08	74.38	27.43	67.06
gHOI	36.44	73.55	35.53	72.86	31.55	68.10	22.70	52.12	32.86	74.07	31.24	72.86	30.67	70.27	20.13	49.14
HaLRTC + PCA	31.32	66.67	31.27	65.41	20.14	52.68	17.22	40.22	43.49	83.54	33.61	57.57	19.94	46.61	15.06	35.21
gHOI + PCA	26.82	60.13	20.49	47.92	10.86	25.19	5.38	10.48	21.69	58.72	19.52	52.43	13.27	31.89	6.88	11.95
HaLRTC + MPCA	41.09	76.06	41.38	76.32	21.11	53.64	22.52	53.20	30.31	73.56	29.53	71.54	27.46	68.17	17.52	46.01
gHOI + MPCA	41.43	76.58	22.89	56.80	10.81	21.47	18.08	36.32	24.14	64.81	21.37	60.88	9.32	21.01	7.58	16.23
HaLRTC + LRANTD	36.41	73.98	35.73	73.55	33.72	70.39	26.90	58.53	22.27	53.66	21.47	51.63	21.20	50.00	11.98	30.43
gHOI + LRANTD	36.53	74.16	35.12	74.29	31.31	68.27	21.83	50.69	21.45	52.12	20.69	49.92	19.98	47.68	11.26	24.61
TRPCA	39.63	77.40	36.67	74.98	30.55	63.64	21.89	45.97	33.21	72.20	32.09	70.88	30.21	68.35	25.90	58.85
TDVM	**84.21**	**96.45**	**81.67**	**94.81**	**76.82**	**91.99**	**75.04**	**91.95**	**82.51**	**95.76**	**75.54**	**95.14**	**72.79**	**95.29**	**59.91**	**94.63**

Table 3. Time cost (seconds) of feature extraction on the FERET and YaleB with pixel/block-based missing entries.

Data	Missing Pixels/Block	HaLRTC [?]	gHOI [?]	HaLRTC + PCA [?]	gHOI + PCA	HaLRTC + MPCA [?]	gHOI + MPCA	HaLRTC + LRANTD [?]	gHOI + LRANTD	TRPCA [?]	**TDVM**
FERET	10%	101.6	67.2	112.7	99.4	270.4	289.9	237.1	383.1	231.0	**32.6**
	30%	114.2	149.8	120.8	310.5	313.3	516.8	289.6	413.9	269.2	**33.3**
	50%	123.7	230.4	129.9	533.6	332.6	687.5	325.0	612.9	212.7	**23.7**
	90%	120.0	175.3	104.1	442.1	132.1	654.8	111.2	563.6	118.2	**20.1**
FERET	5×10	103.1	521.3	114.8	166.5	191.5	254.5	445.5	521.3	170.0	**29.3**
	20×20	110.5	599.7	120.4	220.2	178.7	299.1	459.9	599.7	165.9	**24.0**
	40×30	119.1	555.1	129.9	221.7	209.5	220.2	479.0	555.1	139.8	**21.7**
	55×55	166.6	465.9	117.3	208.2	245.0	213.1	377.5	465.9	144.5	**30.3**
YaleB	10%	127.8	318.2	183.6	721.8	619.6	1095.0	494.6	674.3	156.5	**45.9**
	30%	150.8	631.3	203.5	1423.9	650.6	2202.0	575.2	1645.2	160.0	**45.5**
	50%	160.9	624.6	216.1	1304.9	684.5	1783.6	565.0	2218.2	160.2	**45.3**
	90%	241.0	1052.8	223.4	1030.3	425.5	1152.7	590.0	997.6	118.7	**49.6**
YaleB	5×5	177.0	516.3	179.1	653.6	610.7	788.1	499.7	2422.6	160.4	**49.5**
	8×10	169.3	598.9	206.6	601.9	573.2	1483.3	502.1	1576.4	163.6	**47.0**
	16×16	210.9	611.2	226.8	611.0	682.8	1600.7	571.7	880.8	161.8	**52.5**
	30×25	175.4	568.0	212.1	569.0	426.3	1453.0	513.5	548.7	133.6	**45.2**

4.4 Computational Cost

We report the average time cost of feature extraction in **Table** 3. As shown in Table 3, TDVM is much more efficient than all the compared methods in all cases, as we impose nuclear norm on core tensors instead of original tensors to learn low-dimensional features. Specifically, HaLRTC is the second fastest methods on FERET with block-based

missing entries while slower than gHOI and HaLRTC + PCA in two pixel-based missing cases. Besides, HaLRTC is also the second efficient method on YaleB with pixel-based missing entries excepting the case of missing 90% pixels. In the block-based missing cases of YaleB, TRPCA is faster than TDVM, but it yields worse results. Moreover, the "two-step" strategies such as gHOI + MPCA/LRANTD are the most time consuming (more than 10 times slower than TDVM on average).

4.5 Summary of Experimental Results

- TDVM outperforms the nine competing methods in all cases of face recognition on two real data, especially on data with more missing entries. Besides, our method is much more efficient than all compared methods. Moreover, with less training features (e.g. $L = 1$ for FERET and $L = 5$ for YaleB) in NNC, TDVM shows more advantage as it extracts low-dimensional informative features. These results verifies the superiority of incorporating low-rank Tucker decomposition with feature variance maximization.
- The tensor learning method (TRPCA) is the best performing existing algorithm in six cases of FERET with pixel-based missing entries. However, it works much worse than TDVM on data with increasing missing entries. For example, on YaleB with 90% missing pixels, TRPCA loses up to 94.48% than TDVM on average.
- Tensor completion methods (HaLRTC and gHOI) obtain similar results in most cases and HaLRTC achieves the second best results in about half of all cases, while TDVM outperforms these two methods by 34.92% and 41.71% on average on FERET and YaleB respectively. These results echo our claim: tensor completion methods focus on recovering missing data and do not explore the relationships among samples for effective feature extraction.
- The "two-step" strategies (e.g., gHOI + PCA/MPCA) do not have much improvement and even perform worse than using only tensor completion methods (e.g., gHOI), as we claimed that reconstruction errors from completion step can deteriorate the performance in feature extraction step. Although gHOI + LRANTD/MPCA and HaLRTC + PCA/MPCA achieve the second best results in a few cases, TDVM outperforms the "two-step" strategies in all cases as we extracts informative features directly from observed entries.

5 Conclusion

In this paper, we have proposed an unsupervised feature extraction method, i.e. TDVM, which solves the problem of feature extraction for tensors with missing data. TDVM incorporates low-rank Tucker decomposition with feature variance maximization in a unified framework, which results in low-dimensional informative features extracted directly from observed entries. We have evaluated the proposed method on two real datasets with different pixel/block-based missing entries and applied the extracted features for face recognition. Experimental results have shown the superiority of TDVM in both pixel-based and block-based missing cases, where the proposed method consistently outperforms the nine competing methods in all cases, especially on data with more missing entries. Moreover, TDVM is not sensitive to parameters and more efficient than the compared methods.

Acknowledgment

This work is supported by the NSFC Grant: 61672444, HKBU Faculty Research Grant: FRG2/16-17/051, the SZSTI Grant: JCYJ20160531194006833, Hong Kong PhD Fellowship Scheme, and JSPS KAKENHI Grant: 17K00326. We thank Prof. Canyi Lu, Dr. Guoxu Zhou, Prof. Guangcan Liu, and Dr. Johann Bengua, for their helpful discussion.

References

1. Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.
2. Ledyard R Tucker. Implications of factor analysis of three-way matrices for measurement of change. *Problems in Measuring Change*, 15:122–137, 1963.
3. Haiping Lu, Konstantinos N Plataniotis, and Anastasios N Venetsanopoulos. MPCA: Multilinear principal component analysis of tensor objects. *IEEE Trans. Neural Networks*, 19(1):18–39, 2008.
4. Jiwen Lu, Yap-Peng Tan, and Gang Wang. Discriminative multimanifold analysis for face recognition from a single training sample per person. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):39–51, 2013.
5. Qiquan Shi and Haiping Lu. Semi-orthogonal multilinear PCA with relaxed start. In *Int. Conf. on Joint Conference on Artificial Intelligence*, 2015.
6. Bokai Cao, Chun-Ta Lu, Xiaokai Wei, S Yu Philip, and Alex D Leow. Semi-supervised tensor factorization for brain network analysis. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 17–32. Springer, 2016.
7. Xutao Li, Michael K Ng, Gao Cong, Yunming Ye, and Qingyao Wu. Mr-ntd: Manifold regularization nonnegative tucker decomposition for tensor data dimension reduction and representation. *IEEE Trans. Neural Netw. Learn. Syst.*, 2016.
8. I. T. Jolliffe. *Principal Component Analysis, second edition*. Springer Serires in Statistics, 2002.
9. Canyi Lu, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, pages 5249–5257, 2016.
10. Yigang Peng, Arvind Ganesh, John Wright, Wenli Xu, and Yi Ma. Rasl: Robust alignment by sparse and low-rank decomposition for linearly correlated images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11):2233–2246, 2012.
11. Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. Robust recovery of subspace structures by low-rank representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):171–184, 2013.
12. Evrim Acar, Daniel M Dunlavy, Tamara G Kolda, and Morten Mørup. Scalable tensor factorizations for incomplete data. *Chemometrics and Intelligent Laboratory Systems*, 106(1):41–56, 2011.
13. David Williams, Xuejun Liao, Ya Xue, Lawrence Carin, and Balaji Krishnapuram. On classification with incomplete data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(3), 2007.
14. David Williams, Xuejun Liao, Ya Xue, and Lawrence Carin. Incomplete-data classification using logistic regression. In *Proc. 22nd Int. Conf. on Machine Learning*, pages 972–979. ACM, 2005.
15. Onur G Guleryuz. Nonlinear approximation based image recovery using adaptive sparse reconstructions and iterated denoising-part i: theory. *IEEE Trans. Image Process.*, 15(3):539–554, 2006.

16. Elad Hazan, Roi Livni, and Yishay Mansour. Classification with low rank and missing data. In *Proc. 32nd Int. Conf. on Machine Learning*, pages 257–266, 2015.
17. Ji Liu, Przemyslaw Musialski, Peter Wonka, and Jieping Ye. Tensor completion for estimating missing values in visual data. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(1):208–220, 2013.
18. Wenrui Hu, Dacheng Tao, Wensheng Zhang, Yuan Xie, and Yehui Yang. The twist tensor nuclear norm for video completion. *IEEE Transactions on Neural Networks and Learning Systems*, 2016.
19. Yuanyuan Liu, Fanhua Shang, Wei Fan, James Cheng, and Hong Cheng. Generalized higher-order orthogonal iteration for tensor decomposition and completion. In *Advances in Neural Information Processing Systems*, pages 1763–1771, 2014.
20. Guoxu Zhou, Andrzej Cichocki, Qibin Zhao, and Shengli Xie. Efficient nonnegative tucker decompositions: Algorithms and uniqueness. *IEEE Trans. Image Process.*, 24(12):4990–5003, 2015.
21. Chengcheng Jia, Guoqiang Zhong, and Yun Fu. Low-rank tensor learning with discriminant analysis for action classification and image recovery. In *Proc. Int. Conf. Artificial Intelligence*, pages 1228–1234. AAAI Press, 2014.
22. Tong Tong Wu and Kenneth Lange. Matrix completion discriminant analysis. *Comput. Stat. Data Anal.*, 92:115–125, 2015.
23. Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.
24. Stephen Boyd. Alternating direction method of multipliers. In *NIPS Workshop on Optimization and Machine Learning*, 2011.
25. Haiping Lu, Konstantinos N Plataniotis, and Anastasios Venetsanopoulos. *Multilinear Subspace Learning: Dimensionality Reduction of Multidimensional Data*. CRC press, 2013.
26. John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(2):210–227, 2009.
27. Guangcan Liu and Shuicheng Yan. Latent low-rank representation for subspace segmentation and feature extraction. In *Proc. IEEE Conf. on Computer Vision*, pages 1615–1622. IEEE, 2011.
28. Ehsan Elhamifar and Rene Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(11):2765–2781, 2013.
29. Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
30. Fanhua Shang, Yuanyuan Liu, and James Cheng. Generalized higher-order tensor decomposition via parallel admm. In *Proc. AAAI Conf. on Artificial Intelligence*, pages 1279–1285. AAAI Press, 2014.
31. Nick Higham and Pythagoras Papadimitriou. Matrix procrustes problems. *Rapport Technique, University of Manchester*, 1995.
32. Yuanyuan Liu, Fanhua Shang, Wei Fan, James Cheng, and Hong Cheng. Generalized higher order orthogonal iteration for tensor learning and decomposition. *IEEE Trans. Neural Netw. Learn. Syst.*, 27(12):2551–2563, 2016.
33. Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits, 1998.
34. P Jonathon Phillips, Hyeonjoon Moon, Syed A Rizvi, and Patrick J Rauss. The FERET evaluation methodology for face-recognition algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(10):1090–1104, 2000.
35. K.C. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 27(5):684–698, 2005.