Robust Face Clustering Via Tensor Decomposition

Xiaochun Cao, *Senior Member, IEEE*, Xingxing Wei, Yahong Han, and Dongdai Lin

***Abstract*—Face clustering is a key component either in image managements or video analysis. Wild human faces vary with the poses, expressions, and illumination changes. All kinds of noises, like block occlusions, random pixel corruptions, and various disguises may also destroy the consistency of faces referring to the same person. This motivates us to develop a robust face clustering algorithm that is less sensitive to these noises. To retain the underlying structured information within facial images, we use tensors to represent faces, and then accomplish the clustering task based on the tensor data. The proposed algorithm is called robust tensor clustering (RTC), which firstly finds a lower-rank approximation of the original tensor data using a L1 norm optimization function. Because L1 norm does not exaggerate the effect of noises compared with L2 norm, the minimization of the L1 norm approximation function makes RTC robust. Then, we compute high-order singular value decomposition of this approximate tensor to obtain the final clustering results. Different from traditional algorithms solving the approximation function with a greedy strategy, we utilize a nongreedy strategy to obtain a better solution. Experiments conducted on the benchmark facial datasets and gait sequences demonstrate that RTC has better performance than the state-of-the-art clustering algorithms and is more robust to noises.**

***Index Terms*—Disguise, face clustering, nongreedy maximization, occlusion, pixel corruption, tensor clustering.**

I. Introduction

FACE clustering, which aims to cluster the given facial images into their underlying groups is an important issue in computer vision. For example, to manage the human-centered photos, faces referring to the same people are usually first clustered together, and then face tagging techniques are applied to associate these faces with people names. Such systems have been successfully exploited in some commercial cases like Google Picasa, Apple iPhoto, and Microsoft Easy Album. Face clustering in videos can be found in video indexing and content analysis [1], preprocessing for face recognition [2], determining the cast of a feature-length film [3], etc. A successful solution of the face clustering problem will improve the performance of these tasks.

Intuitively, face clustering is a special case of the general data clustering problem, and thus, some classic methods like k-means [4], spectral clustering [5] can be directly used to group the facial images. However, human faces in the wild will vary with the poses, expressions, and illumination changes, resulting in the great difference between facial images coming from the same people. Additionally, in the real world, the human face may meet with all kinds of noises, including block occlusions, pixel corruptions, and various disguises [6]. Due to the unpredictable nature of noises, the block occlusions may appear in any part of the image and may be arbitrarily large in magnitude. The percentage of the corrupted pixels within an image is also unknown. Developing a robust face clustering algorithm to handle these uncertainties is necessary.

In order to cluster faces, the standard facial clustering approaches usually extract features from the facial image [7]–[9] or directly use the raw pixels [6] to construct a feature vector, and then accomplish the clustering tasks based on these vectors. One drawback of such vectorization is that some structured priors within an image may be discarded. In contrast, tensors, which are known as multidimensional arrays [10], are the natural representation of visual data. For example, a gray image can be represented by a second-order tensor [11], and a time-series video clip can be represented by a third-order tensor [12]. Compared with the relaxation of tensors by vectorizing an image into a long vector, tensor-based representation can preserve more spatial information of images. Therefore, it is more informative than the vectorization method. In addition, the dimensionality of tensor-based representation is much lower, which can effectively avoid high computational complexity and large memory requirements [13]. If we use a matrix (the second-order tensor) to represent a facial image, then, the face clustering can be converted into a tensor clustering problem [14], [15], which seeks to divide several M-order input tensors into groups by minimizing some certain criterion.

Considering the aforementioned reasons, in this paper, we aim to develop a tensor clustering algorithm that is robust to the noises existing within facial images, like different poses, expressions, illuminations, block occlusions, pixel corruptions, and various disguises. We call it robust tensor clustering (RTC). Our method firstly finds a lower-rank approximation of the original tensor data using a L1 norm optimization function. Because L1 norm doesn't exaggerate the effect of noises compared with L2 norm, the minimization of the L1 norm approximation function makes RTC robust.

Manuscript received February 3, 2014; revised June 11, 2014 and October 8, 2014; accepted November 25, 2014. Date of publication December 23, 2014; date of current version October 13, 2015. This work was supported in part by the National Natural Science Foundation of China under Grant 61422213, Grant 61332012, Grant 61202166, and Grant 61472276, in part by the National Basic Research Program of China under Grant 2013CB329305, in part by the 100 Talents Programme of the Chinese Academy of Sciences, and in part by the Doctoral Fund of Ministry of Education of China under Grant 20120032120042. This paper was recommended by Associate Editor S. Zafeiriou.

X. Cao and D. Lin are with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China.

X. Wei is with the School of Computer Science and Technology, Tianjin University, Tianjin 300072, China (e-mail: xwei@tju.edu.cn).

Y. Han is with the Tianjin Key Laboratory of Cognitive Computing and Application, School of Computer Science and Technology, Tianjin University, Tianjin 300072, China.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2014.2376938

Then, we compute the high-order singular value decomposition (HOSVD) of this approximate tensor to obtain the final clustering results. We formulate the process of approximation into a framework of tensor principal component analysis with L1-norm (TPCAL1) [16]. TPCAL1 is a variant of tensor PCA with Frobenius norm (TPCAF) [17], where the Frobenius norm is replaced by the L1-norm. In this way, TPCAL1 will suppress the negative influence of noises. To maximize the function of TPCAL1, Pang *et al*. [16] use a greedy strategy like [18]. Specifically, the projection directions are sequentially optimized one by one. To obtain a better solution, inspired by [19], we utilize a nongreedy maximization to directly solve the projection directions. Experiments demonstrate that the nongreedy algorithm leads to a better clustering performance.

We summarize the contributions of this paper as below.

1) We propose a RTC algorithm, which can handle the faces with different expressions and illuminations. RTC is also less sensitive to the block occlusions, random pixel corruptions, and various disguises. Meanwhile, RTC utilizes the structured information within the facial images when accomplishing the clustering tasks. These three specialities make RTC different from the existing face clustering algorithms.
2) We present a nongreedy maximization method, and then integrate it into the framework of RTC. Compared with the vector-based nongreedy approach proposed in [19], ours can handle the tensor data, and can obtain better solution than the traditional tensor-based greedy strategy in [16].
3) We provide the formal proof about the equivalence of HOSVD and *k*-means when clustering arbitrary order tensor data. Huang *et al*. [14] have proven the equivalence of HOSVD and *k*-means clustering just for the second-order tensor data. We extend this idea to the high-order case.

We compare RTC with ten state-of-the-art clustering methods on gait sequences and six benchmark facial datasets. Experimental results show the competitiveness of RTC. A shorter version of this paper appeared in [20]. This journal version extends this paper with an added theoretical proof, a computational complexity analysis, and some experiments on the high-order tensor data. Additionally, Cao *et al*. [20] used the synthetic noises. To verify the effectiveness of RTC, we herein focus on facial images and use the real-world facial datasets. On this basis, more experimental comparisons and more detailed discussions are given.

The rest of this paper is organized as follows. The related work about face clustering approaches is introduced in Section II. We present the framework of RTC in Section III. The solution and algorithm are given in Section IV. Section V reports all experimental results. Finally, we summarize the conclusion in Section VI.

II. Related Work

In the past decades, several face clustering algorithms have been reported in the literatures. To make the clustering algorithms robust to noises, some work aim to learn a better feature representation for the facial image. For extracting the "good" facial descriptor, i.e., minimizing intraperson dissimilarities while enlarging the margin between different people, some classic features like Eigenface [21], Fisherface [22], Gabor [23], and LBP faces [24] have been proposed. Among these methods, Gabor and LBP faces are more stable to local changes like illuminations and expressions. In order to obtain more accurate face representation, some novel features are proposed while retaining the advantage of the classic methods. For instance, Vu and Caplier [25] proposed a descriptor named patterns of oriented edge magnitudes (POEM). The POEM feature is built by applying a self-similarity based structure on oriented magnitudes. It is calculated by accumulating a local histogram of gradient orientations over all pixels of image cells. To catch better detailed object appearance, notably when images are degraded, Vu [26] gives an improved version of POEM. In addition, a new feature called patterns of orientation difference (POD) is also introduced in this paper, which encodes the relationships between orientatios of local image patches instead of the relationships between edge magnitude distributions like POEM. The improved POEM and POD are finally concentrated by the principal component analysis (PCA) technique. Experiments show that this hybrid feature is more efficient than contemporary ones in terms of both higher performance and lower complexity. Several other feature learning methods can be found in [7] and [27]. This paper aims at increasing the performance and robustness of the clustering algorithm itself. Therefore, we directly use the raw pixels of facial images to cluster faces, which is different from the above methods that want to develop a robust feature representation for faces.

Another kind of methods try to utilize some contextual cues to help cluster noisy faces. For example, Wu and Tang [28] proposed to use the social context information inherent among the people in a collection to build a social network, and then, combine this knowledge with face similarity measure to generate the final face clusters. Zhang *et al*. [29] first detected human skin, hair, clothing regions. Then, they extract features robustly. By matching these features, high-precision contextual clusters are obtained. Kumar *et al*. [30] proposed to employ human attributes as an additional feature to accomplish the clustering tasks. The above work just explored one or a few contextual feature types when performing the clustering. In contrast, Zhang *et al*. [31] integrated some heterogeneous contexts into a unified framework to jointly cluster faces. Experiments show the proposed method can obtain an improving recall while maintaining high precision of face clustering. Besides, Wu *et al*. [32] focused on face clustering in videos, where faces are organized as face tracks and the frame index of each face is provided. As a result, many pairwise constraints between faces can be obtained from the temporal and spatial knowledge. These constraints are incorporated into a generative clustering model based on hidden Markov random fields. In this paper, instead of utilizing the contextual information, we try to explore the inherent structured prior underlying the facial images to improve the clustering performance.

Besides better feature representation and context cues, the third line of face clustering researches focuses on improving

the algorithm itself. As we known, L1 norm does not exaggerate the effect of noises compared with L2 norm. To utilize this advantage, Wright *et al*. [6] proposed a general classification algorithm for object recognition based on a sparse representation with L1-minimization. The experiments demonstrate that the proposed method is robust to the block occlusions, pixel corruptions, and various disguises. Kwak [18] integrates the L1 optimization technique into PCA, presenting a greedy strategy to solve the problem. He tests the robustness of the proposed algorithm by evaluating whether the noisy faces with block occlusions can be well reconstructed. Li *et al.* [33] extended the similar idea to the 2-D case, and propose the L1-norm based 2-D PCA (2-DPCAL1). Different from [18], 2-DPCAL1 directly uses a matrix to represent an image. Experimental results show that 2-DPCAL1 not only makes good use of spatial information but is also robust to outliers. Inoue *et al.* [34] instead of using L1 norm to obtain the robustness, derived iterative algorithms on the basis of Lagrange multipliers to deal with the sample outliers and intrasample outliers. Experiments demonstrate their effectiveness. For more tensor-based analysis methods, please refer to [35] for a survey. Note that, in [18], Kwak uses a greedy strategy to solve the maximization problem. To obtain a better solution, Nie *et al*. [19] proposed a nongreedy strategy optimization method to solve the same problem. Experiments show its effectiveness. Inspired by these work [18], [19], [33], RTC utilizes tensor data to represent facial images, and also uses L1 norm to suppress the influence of noises. However, we extend the nongreedy strategy in [19] to handle the tensor data. Besides, different from the above methods focusing on the application of face reconstruction or recognition, we apply RTC to solve face clustering problem.

III. Proposed Framework

In this section, we introduce the details of proposed framework. Consider a sequence of input M-order (mode) tensor data $\mathcal{X}_j \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M}, j = 1, \ldots, n$ (for a facial image, $M = 2$, and for a video clip, $M = 3$), the goal of clustering is to find a K-way disjoint partitioning $(\mathbb{S}_1, \ldots, \mathbb{S}_K)$ such that the following objective is minimized:

$$F_{\text{clustering}} = \sum_{k=1}^{K} \sum_{\mathcal{X}_j \in \mathbb{S}_k} \left\| \mathcal{X}_j - \mathcal{C}_k \right\|^2 \tag{1}$$

where $\mathcal{C}_k \in \mathbb{R}^{n_1 \times n_2 \times \ldots \times n_M}$ denotes the kth cluster centroid, and K is the number of clusters. $\|\cdot\|$ denotes the Euclidean distance between $\mathcal{X}_j$ and $\mathcal{C}_k$. We introduce an indicator vector $\mathbf{q}_k$ for the cluster $\mathbb{S}_k$

$$\mathbf{q}_k(j) = \begin{cases} 1 & \text{if } \mathcal{X}_j \in \mathbb{S}_k \\ 0 & \text{if } \mathcal{X}_j \notin \mathbb{S}_k \end{cases} \tag{2}$$

where $\mathbf{q}_k^T \mathbf{q}_k$ is the size of cluster $\mathbb{S}_k$. Now, the matrix $\mathbf{Q}$ is defined such that the kth column of $\mathbf{Q}$ equal to $\mathbf{q}_k / (\mathbf{q}_k^T \mathbf{q}_k)^{1/2}$. $\mathbf{Q}$ is an orthonormal matrix, $\mathbf{Q}^T \mathbf{Q} = \mathbf{I}$, and $\mathbf{Q} \in \mathbb{R}^{n \times K}$.

The RTC aims to find a more accurate indicator matrix $\mathbf{Q}$ via making the clustering algorithm robust to noises. To this end, RTC maximizes the following problem to get projection matrices, and then uses the obtained projection matrices to compute the approximation for the original tensor data:

$$\max_{\mathbf{U}_i|_{i=1}^M} \sum_{j=1}^{n} \|\mathcal{X}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i^T\|_1, \quad \text{s.t. } \mathbf{U}_i^T \mathbf{U}_i = \mathbf{I} \tag{3}$$

where $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i} (i = 1, \ldots, M)$ is called the projection matrix. Generally, $r_i < n_i$, so we can obtain a low-rank approximation $\mathcal{Y}_j$ for $\mathcal{X}_j$. $\mathcal{Y}_j \approx \mathcal{X}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i^T$, where $\mathcal{X}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i^T = \mathcal{X}_j \times_1 \mathbf{U}_1^T \cdots \times_M \mathbf{U}_M^T$, and $\mathcal{X}_j \times_i \mathbf{U}_i^T$ denotes the i-mode product of the tensor $\mathcal{X}_j$ with the matrix $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}$. Please refer to [10] for the formal definition of i-mode product. $\|\cdot\|_1$ is the tensor's L1-norm. The L1-norm of a M-order tensor $\mathcal{X} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M}$ is defined as the sum of the absolute value of its elements

$$\|\mathcal{X}\|_1 = \sum_{i_1=1}^{n_1} \cdots \sum_{i_M=1}^{n_M} |\mathcal{X}_{i_1, \ldots, i_m}|. \tag{4}$$

The problem in (3) is called TPCAL1, which is a variant of TPCAF, replacing the Frobenius norm by the L1-norm, and thus can suppress the negative influence of noises [18]. Once $\mathbf{U}_i \in \mathbb{R}^{n_i \times r_i}, i = 1, \ldots, M$ is computed, the low-rank approximation $\mathcal{Y}_j$ for $\mathcal{X}_j$ can be computed.

Thus, we obtain a T-order tensor $\mathcal{Z} \in \mathbb{R}^{r_1 \times \cdots \times r_M \times n}$ composing of n low-rank approximation $\mathcal{Y}_j \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_M}$, where $T = M + 1$. We then decompose the tensor $\mathcal{Z}$ to obtain the final clustering results for $\mathcal{X}_j, (j = 1, \ldots, n)$ by solving the following optimization function:

$$\min_{\mathcal{S}, \mathbf{P}_i|_{i=1}^T} \|\mathcal{Z} - \mathcal{S} \prod_{i=1}^{T} \times_i \mathbf{P}_i\|_F^2, \quad \text{s.t. } \mathbf{P}_i^T \mathbf{P}_i = \mathbf{I}, \quad \forall i \in (1, \ldots, T) \tag{5}$$

where $\mathcal{S} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times K}$ is called the core tensor. $\mathbf{P}_i \in \mathbb{R}^{r_i \times d_i} (i = 1, \ldots, M)$ and $\mathbf{P}_T \in \mathbb{R}^{n \times K}$ are called factor matrices. The solution to problem (5) can be found by HOSVD [17]. Huang *et al*. [14] demonstrated that $\mathbf{P}_T \in \mathbb{R}^{n \times K}$ is the clustering results for n second-order tensors $\mathcal{X}_j$. i.e., $\mathbf{P}_T \in \mathbb{R}^{n \times K}$ is the indicator matrix of clustering n second-order tensors $\mathcal{X}_j$ into K cluster centroids. In Section IV-B, we will prove that this idea still stands when meeting arbitrary order tensor. Owing to this, we obtain $\mathbf{Q} = \mathbf{P}_T \in \mathbb{R}^{n \times K}$.

IV. Solution and Algorithm

In this section, we firstly give the nongreedy maximization algorithm for solving (3). After that, we prove that $\mathbf{P}_T \in \mathbb{R}^{n \times K}$ is still the indicator matrix when clustering arbitrary order tensors. Then, the RTC algorithm is summarized. Finally, we discuss the computational complexity of RTC.

A. Nongreedy Maximization

To obtain the projection matrix $\mathbf{U}_i$, Nie *et al*. [19] have proposed a nongreedy maximization for vector case. Here, we extend this nongreedy strategy to handle the tensor data. Before that, we recall the main procedure in [19].

Given a sequence of vectors (the first-order tensor) $\mathbf{x}_j \in \mathbb{R}^d, j = 1, \ldots, n$, then problem (3) is formulated as follows:

$$\max_{\mathbf{U}} \sum_{j=1}^{n} \left\| \mathbf{U}^T \mathbf{x}_j \right\|_1, \quad \text{s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I} \tag{6}$$

where $\mathbf{U} \in \mathbb{R}^{d\times m}$ is the projection matrix, and $d > m$. Without loss of generality, $\mathbf{x}_j, j = 1, \ldots, n$ are assumed to be centralized, i.e., $\sum_{j=1}^{n} \mathbf{x}_j = 0$. The problem (6) should be replaced by (7)

$$\max_{\mathbf{U}} \sum_{j=1}^{n} \alpha_j^T \mathbf{U}^T \mathbf{x}_j, \quad \text{s.t. } \mathbf{U}^T\mathbf{U} = \mathbf{I} \tag{7}$$

where $\alpha_j = sgn(\beta_j)$, and $\beta_j = \mathbf{U}^T\mathbf{x}_j$. $sgn(\cdot)$ denotes the sign function defined as follows: $sgn(\beta_j(i)) = 1$ if $\beta_j(i) > 0$, $sgn(\beta_j(i)) = -1$ if $\beta_j(i) < 0$, and $sgn(\beta_j(i)) = 0$ if $\beta_j(i) = 0$. $\beta_j(i)$ indicates the ith element in the vector β_j. Denote $\mathbf{M} = \sum_{j=1}^{n} \mathbf{x}_j\alpha_j^T$, [19] proves that $\mathbf{U}$ can be obtained by computing the SVD of $\mathbf{M}$ as $\mathbf{M} = \mathbf{W}\Lambda\mathbf{V}^T$, and then $\mathbf{U} = \mathbf{W}\mathbf{V}^T$. This nongreedy strategy directly solves the projection direction $\mathbf{U}$, and has been demonstrated to get a better solution than the nongreedy method [18].

Now, we discuss how to extend this nongreedy strategy to handle the tensor data. Given a sequence of n-order tensors $\mathcal{X}_j \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_M}, j = 1, \ldots, n$. Problem (3) is formulated as follows [10]:

$$\max_{\mathbf{U}_i} \sum_{j=1}^{n} \left\| \mathbf{U}_i^T \mathbf{X}_j^{(i)} \mathbf{H}_i \right\|_1, \quad \text{s.t. } \mathbf{U}_i^T\mathbf{U}_i = \mathbf{I} \tag{8}$$

where $\mathbf{X}_j^{(i)}$ denotes the mode-i unfolding of $\mathcal{X}_j$, $\mathbf{X}_j^{(i)} \in \mathbb{R}^{n_i\times J}$, and $J = n_1 \times \cdots \times n_{i-1} \times n_{i+1} \times \cdots \times n_M$. $\mathbf{H}_i = (\mathbf{U}_1 \otimes \cdots \otimes \mathbf{U}_{i-1} \otimes \mathbf{U}_{i+1} \otimes \cdots \otimes \mathbf{U}_M)$, and $\otimes$ is the Kronecker product. The Kronecker product of matrices $\mathbf{U}_i \in \mathbb{R}^{n_i\times r_i}$ and $\mathbf{U}_j \in \mathbb{R}^{n_j\times r_j}$ is defined by

$$\mathbf{U}_i \otimes \mathbf{U}_j = \begin{bmatrix} \mathbf{U}_i^{11}\mathbf{U}_j & \mathbf{U}_i^{12}\mathbf{U}_j & \cdots & \mathbf{U}_i^{1r_i}\mathbf{U}_j \\ \mathbf{U}_i^{21}\mathbf{U}_j & \mathbf{U}_i^{22}\mathbf{U}_j & \cdots & \mathbf{U}_i^{2r_i}\mathbf{U}_j \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{U}_i^{n_i1}\mathbf{U}_j & \mathbf{U}_i^{n_i2}\mathbf{U}_j & \cdots & \mathbf{U}_i^{n_ir_i}\mathbf{U}_j \end{bmatrix} \tag{9}$$

where $\mathbf{U}_i^{n_ir_i}$ is the element located in the n_ith row and r_ith column in $\mathbf{U}_i$, $\mathbf{U}_i \otimes \mathbf{U}_j \in \mathbb{R}^{(n_in_j)\times(r_ir_j)}$. From the definition, we can infer $\mathbf{H}_i \in \mathbb{R}^{J\times F}$, where $F = r_1 \times \cdots \times r_{i-1} \times r_{i+1} \times \cdots \times r_M$.

Based on the above discussions, we solve the projection matrices by a block coordinate ascent optimization technique. Specifically, we solve $\mathbf{U}_i$ while fixing other matrices. After obtaining $\mathbf{U}_i$, we add it into the fixing ones, and solve another. The procedure is iteratively performed until the convergence criteria reaches.

To begin, let $\mathbf{U}_1, \ldots, \mathbf{U}_{i-1}, \mathbf{U}_{i+1}, \ldots, \mathbf{U}_M$ be initialized and then fixed. Now, the problem is to find the optimal $\mathbf{U}_i$ such that (8) is maximized. In this case, problem (8) can be formulated as follows:

$$\max_{\mathbf{U}_i} \sum_{j=1}^{n} \sum_{l=1}^{F} \left\| \mathbf{U}_i^T \mathbf{X}_j^{(i)} (\mathbf{H}_i)_l \right\|_1, \quad \text{s.t. } \mathbf{U}_i^T\mathbf{U}_i = \mathbf{I} \tag{10}$$

where $(\mathbf{H}_i)_l \in \mathbb{R}^{J\times 1}$ denotes the lth column in $\mathbf{H}_i$. Problem (10) can be formulated as

$$\max_{\mathbf{U}_i} \sum_{k=1}^{p} \left\| \mathbf{U}_i^T \mathbf{z}_k \right\|_1, \quad \text{s.t. } \mathbf{U}_i^T\mathbf{U}_i = \mathbf{I} \tag{11}$$

where $p = n \times F$, $\mathbf{z}_k = \mathbf{X}_j^{(i)}(\mathbf{H}_i)_l$ and $\mathbf{z}_k \in \mathbb{R}^{n_i\times 1}$. Problem (11) is identical with (6), and thus can be easily solved. The other projection matrices are solved by the same procedure.

We use the relative reduction of the root mean absolute reconstruction error (RMARE) to check the convergence. The *RMARE* is defined as follows:

$$RMARE = \sqrt{\frac{1}{n} \sum_{j=1}^{n} \left\| \mathcal{X}_j - \mathcal{Y}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i \right\|_1} \tag{12}$$

where $\mathcal{Y}_j = \mathcal{X}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i^T$, and $\mathcal{Y}_j \in \mathbb{R}^{r_1\times r_2\times\cdots\times r_M}$. *RMARE* is a modified version of *RMSRE* [36], replacing the Frobenius norm with the L1-norm. The convergence is determined by checking the following inequality:

$$\frac{|RMARE(t-1) - RMARE(t)|}{RMARE(t-1)} < \eta \tag{13}$$

for some small threshold $\eta > 0$. $RMARE(t)$ denotes the *RMARE* value in the tth iteration step. In our experiments, we choose $\eta = 10^{-3}$. Results in Section V show that the algorithm converges within two to three iterations.

B. Equivalence of HOSVD and Clustering for M-Order Tensor

Huang *et al.* [14] have proven that $\mathbf{P}_T \in \mathbb{R}^{n\times K}$ [see (5)] is the clustering results for n second-order tensors $\mathbf{X}_j$. Actually, this still stands when clustering M-order tensors ($M > 2$). In this section, we give the proof.

Theorem 1: The solution $\mathbf{P}_T \in \mathbb{R}^{n\times K}$ of (5) is still the indicator matrix for clustering M-order tensors when $M > 2$.

Proof: Given n M-order tensors $\mathcal{X}_j \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_M}$, $j = 1, \ldots, n, M > 2$. We unfold each tensor to obtain its mode-1 unfolding $\mathbf{X}_j^{(1)} \in \mathbb{R}^{n_1\times(n_2\ldots n_M)}$, and then construct a third-order tensor $\mathcal{G} \in \mathbb{R}^{n_1\times(n_2\ldots n_M)\times n}$, which is made up of $\mathbf{X}_j^{(1)}, j = 1, \ldots, n$. We accomplish HOSVD factorization for $\mathcal{G} \in \mathbb{R}^{n_1\times(n_2\ldots n_M)\times n}$ as follows:

$$\min_{\mathcal{R},\mathbf{U},\mathbf{V},\mathbf{W}} \|\mathcal{G} - \mathcal{R} \times_1 \mathbf{U} \times_2 \mathbf{V} \times_3 \mathbf{W}\|_F^2 \tag{14}$$

where $\mathbf{U}^T\mathbf{U} = \mathbf{I}, \mathbf{V}^T\mathbf{V} = \mathbf{I}, \mathbf{W}^T\mathbf{W} = \mathbf{I}$, and $\mathbf{W} \in \mathbb{R}^{n\times K}$. Equation (14) is equivalent to (15) according to the definition of Frobenius norm

$$\min_{\mathbf{R},\mathbf{U},\mathbf{A}} \left\| \mathbf{G}^{(1)} - \mathbf{U}\mathbf{R}^{(1)}\mathbf{A}^T \right\|_F^2 \tag{15}$$

where $\mathbf{A} = \mathbf{V} \otimes \mathbf{W}$. $\mathbf{G}^{(1)}$ and $\mathbf{R}^{(1)}$ are the mode-1 unfolding of $\mathcal{G}$ and $\mathcal{R}$, respectively. $\mathbf{G}^{(1)} \in \mathbb{R}^{n_1\times(n_2\cdots n_Mn)}$. According to [14], $\mathbf{W}$ is the indicator matrix of clustering the M-order tensors $\mathcal{X}_j \in \mathbb{R}^{n_1\times n_2\times\cdots\times n_M}, j = 1, \ldots, n$.

Meanwhile, we can directly construct a T-order tensor $\mathcal{Z} \in \mathbb{R}^{n_1\times\cdots\times n_M\times n}$ according to these M-order tensors

Algorithm 1 RTC Algorithm

1: **Input**: $\mathcal{X} = [\mathcal{X}_1, \ldots, \mathcal{X}_n] \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M \times n}$, where $\mathcal{X}$ is centralized, $r_1, r_2, \ldots, r_M$, cluster number K;
2: Initialize $\mathbf{U}_1^t, \ldots, \mathbf{U}_{i-1}^t, \mathbf{U}_{i+1}^t, \ldots, \mathbf{U}_M^t$, where $t = 1$;
3: **while** *not converge* **do**
4: Compute $\mathbf{U}_i^t$ using (11) while fixing the other matrices, $i = 1, \ldots, M$;
5: $t = t + 1$;
6: **end while**
7: Compute $\mathcal{Y}_j = \mathcal{X}_j \prod_{i=1}^{M} \times_i \mathbf{U}_i^T, j = 1, \ldots, n$ to obtain $\mathcal{Z} = [\mathcal{Y}_1, \ldots, \mathcal{Y}_n] \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_M \times n}$;
8: Compute HOSVD of $\mathcal{Z}$ using (5).
9: Perform k-means across the rows in $\mathbf{P}_T$.
10: **Output**: Indicator matrix $\mathbf{Q} \in \mathbb{R}^{n \times K}$

$\mathcal{X}_j \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M}$, where $T = M + 1$. We also accomplish HOSVD factorization for $\mathcal{Z} \in \mathbb{R}^{n_1 \times \cdots \times n_M \times n}$

$$\min_{\mathcal{S}, \mathbf{P}_i|_{i=1}^{M}} \left\| \mathcal{Z} - \mathcal{S} \prod_{i=1}^{T} \times_i \mathbf{P}_i \right\|_F^2, \text{ s.t, } \mathbf{P}_i^T \mathbf{P}_i = \mathbf{I}. \tag{16}$$

Similarly, (16) is equivalent to (17)

$$\min_{\mathbf{S}, \mathbf{P}_1, \mathbf{H}} \left\| \mathbf{Z}^{(1)} - \mathbf{P}_1 \mathbf{S}^{(1)} \mathbf{H}^T \right\|_F^2 \tag{17}$$

where $\mathbf{H} = \mathbf{P}_2 \otimes \cdots \otimes \mathbf{P}_M \otimes \mathbf{P}_T$. $\mathbf{Z}^{(1)}$ and $\mathbf{S}^{(1)}$ are the mode-1 unfolding of $\mathcal{Z}$ and $\mathcal{S}$, respectively. $\mathbf{Z}^{(1)} \in \mathbb{R}^{n_1 \times (n_2 \cdots n_M n)}$.

According to the definition of mode-1 unfolding, $\mathbf{G}^{(1)} = \mathbf{Z}^{(1)}$. Therefore, the solutions of (15) and (17) are identical, i.e., $\mathbf{H} = \mathbf{A}$, $\mathbf{P}_1 = \mathbf{U}$, $\mathbf{S}^{(1)} = \mathbf{R}^{(1)}$. Note $\mathbf{A} = \mathbf{V} \otimes \mathbf{W}$ and $\mathbf{H} = \mathbf{P}_2 \otimes \cdots \otimes \mathbf{P}_M \otimes \mathbf{P}_T$. We can let $\mathbf{V} = \mathbf{P}_2 \otimes \cdots \otimes \mathbf{P}_M$ and $\mathbf{W} = \mathbf{P}_T$. Because $\mathbf{W}$ is the indicator matrix of clustering n second-order tensors, we prove that $\mathbf{P}_T$ is the indicator matrix when $M > 2$. ■

C. RTC Algorithm Summary

After computing $\mathcal{Y}_j \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_M}$, we obtain a T-order tensor $\mathcal{Z} = [\mathcal{Y}_1, \ldots, \mathcal{Y}_n] \in \mathbb{R}^{r_1 \times r_2 \times \cdots \times r_M \times n}$. We use (5) to decompose $\mathcal{Y}$, where $T = M + 1$. Note $\mathbf{P}_T$ is a continuous solution, i.e., the values in $\mathbf{P}_T$ are real numbers, rather than discrete values like 0 or 1. In order to obtain the discrete indicator matrix $\mathbf{Q}$, we perform k-means across the rows in $\mathbf{P}_T \in \mathbb{R}^{n \times K}$. In this way, we get the final discrete indicator matrix $\mathbf{Q} \in \mathbb{R}^{n \times K}$. The whole algorithm for RTC is summarized in Algorithm 1.

D. Computational Complexity

We now discuss the computational complexity of RTC. The major computation of Algorithm 1 is step 4. Since this is an iterative solution, we perform the computational complexity analysis for one iteration.

In step 4, we convert our problem into the vector-based nongreedy maximization issue in [19]. To solve (6), the computational complexity of [19] is $O(ndmt_s)$, where t_s is the number of iterations. The definitions of n, m, d can be found in (6). The experimental results in [19] show $t_s < 10$ in practice. In our problem, we use this vector-based nongreedy algorithm to solve (11). In such case, $d = n_i, n = p, m = r_i$. For simplicity, we assume $r_1 = \cdots = r_M = r$ and $n_1 = \cdots = n_M = l$. Recall $p = n \times F$ and $F = r_1 \times \cdots \times r_{i-1} \times r_{i+1} \times \cdots \times r_M$, the final computational complexity of solving (11) is $O(nlr^M t_s)$. Because, we need to solve M $\mathbf{U}_i$, the computational complexity of step 4 is $O(Mnlr^M t_s)$. The results in Fig. 14 show RTC converges within two or three iterations.

As for step 8, we obtain the core tensor data $\mathcal{Z} \in \mathbb{R}^{r \times r \times \cdots \times r \times n}$. The operation of HOSVD is first to unfold $\mathcal{Z}$ to obtain $\mathbf{Z}^{(M+1)} \in \mathbb{R}^{n \times r^M}$, and then compute the SVD of $\mathbf{Z}^{(M+1)}$. From the results in Fig. 1, we see rank r is usually much lower than l. Therefore, computing HOSVD in step 8 is fast and time-efficient compared with step 4.

V. Experiments

In this section, the evaluation metrics and compared methods are first given in Sections V-A and V-B, respectively. Next, we evaluate the performance of RTC on 2-D facial images, in terms of different poses, illuminations, expressions, and various disguises in Section V-C, as well as random block occlusions and pixel corruptions in Section V-D. Then, we test the algorithm on the high-order tensor dataset (color images and videos) in Section V-E. Finally, the convergence of RTC is provided.

A. Evaluation Metrics

We use two metrics to measure the clustering performance: 1) accuracy (AC) and 2) normalized mutual information (NMI) [37]. Given an image x_i, let r_i and s_i be the obtained cluster label and the ground-truth label, respectively. The AC is defined as follows:

$$AC = \frac{\sum_{i=1}^{n} \delta(s_i, map(r_i))}{n} \tag{18}$$

where n is the total number of images, $\delta(x, y)$ is the delta function that equals to one if $x = y$ and equals to zero otherwise. $map(r_i)$ is the permutation mapping function that maps each cluster label r_i to the equivalent ground-truth label. The best mapping can be found by using the Kuhn–Munkres algorithm.

Let C denote the set of clusters obtained from the ground truth and C' denote the set of clusters obtained from our algorithm. The NMI is defined

$$\overline{MI}(C, C') = \frac{MI(C, C')}{max(H(C), H'(C'))} \tag{19}$$

where $H(C)$ and $H(C')$ are the entropies of C and C', respectively. $MI(C, C')$ denotes the mutual information of C and C'. It is easy to check that $MI(C, C')$ ranges from 0 to 1. $MI = 1$ if the two sets of clusters are identical, and $MI = 0$ if the two sets are independent.

B. Compared Methods

We compare RTC with ten state-of-the-art approaches: k-means [4], two spectral clustering methods: parallel spectral clustering (PSC) [38], and large-scale spectral clustering (LSC-K) [39], as well as three subspace clustering methods: low-rank representation clustering (LRR) [40], sparse

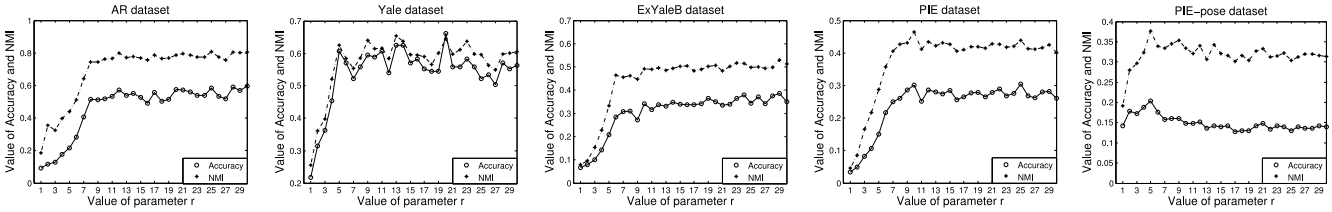


Fig. 1. Performance of RTC in terms of AC and NMI with respect to different values of r_1, r_2. r_1, r_2 are the number of columns of $\mathbf{U_1}$ and $\mathbf{U_2}$ in (3), respectively. For simplicity, we let $r_1 = r_2 = r$, and tune r ranging from 1 to 30.

subspace clustering (SSC) [41], and correlation adaptive subspace segmentation (CASS) [42]. We also compare with the method presented in [14], and call it HOSVD clustering in our experiment. In addition, two PCA clustering methods are considered: TPCA +k-means [10] and MPCA + k-means [43]. To verify the effectiveness of our nongreedy algorithm, we compare RTC with TPCAL1 + HOSVD [16], where the only difference is the method solving the projection matrices.

All the methods need to input the cluster number K. We set K equal to the ground-truth number of classes on their corresponding datasets. In addition, PSC needs to input $\hat{K}$, which is the same to that in K-nearest neighbors (KNN). We tune $\hat{K} \in \{5, 10, 15, 20\}$ in our experiments. LSC-K needs to input the number of landmarks p, which depends on the size of datasets. In our experiments, we tune $p \in \{0.2, 0.4, 0.6, 0.8\}$ of the dataset's size, respectively. Because LSC-K is robust to another parameter r, we let $r = 6$ according to [39]. In LRR, parameter λ is important to the performance. Thereby, we tune $\lambda \in \{0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5\}$, and reports the best performance. For SSC and CASS, we use the default parameters given in their released codes. Similar to RTC, we use TPCA and MPCA to solve projection matrices, and then obtain the low-rank representation for original tensor data. Conversely, this low-rank representation is then vectorized, and input to k-means to obtain the final clustering results. For MPCA, we use the default parameters. Note that, because clustering is an unsupervised process, we do not use the feature extraction step in its released codes.

In the 2-D case, RTC has two parameters: r_1 and r_2. For simplicity, we let $r_1 = r_2 = r$, and then tune r versus different values. The best performance is selected as the results. We tune the parameters of TPCA and TPCAL1 like RTC. For the details of 3-D case, please see Section V-E. For k-means, LRR, SSC, PSC, LSC-K, and CASS, we reshape each image into a vector according to the size of images on different datasets, and then input the vectors to the clustering algorithms. For TPCA + k-means, MPCA + k-means, TPCAL1 + HOSVD, HOSVD, and RTC, we directly use a matrix to represent an image, and then cluster these matrices.

C. Clustering Despite Poses, Illuminations, Expressions, Disguises, and Large-Scale Data

1) Datasets: As we mentioned, faces will vary with the poses, illuminations, expressions, and disguises of individuals. To evaluate the performance of RTC despite these uncertain information, we conduct experiments on the following facial datasets: AR, Yale, ExYaleB, PIE, and PIE-pose. Table I gives a summary of the used datasets.

TABLE I
SUMMARY OF THE USED DATASETS

Name	Subject	Individual	Number	Variation
AR	76	13	988	Disguise
Yale	15	11	165	Expression
ExYaleB	38	64	2242	Illumination
PIE	68	170	11554	Large scale
PIE-pose	30	11	330	Pose

TABLE II
PERFORMANCES OF DIFFERENT IMAGE SIZES ON EXYALEB DATASET

Image Size	16×16	32×32	64×64
AC	0.3497	0.3729	**0.3863**
NMI	0.4980	0.5231	**0.5303**
Speed (sec)	23.59	81.36	117.97

a) AR: This dataset[1] contains 76 male people. Each person has 13 images including wearing sun glasses, wearing scarf, etc. Thus, we have totally 988 images. We use AR dataset to verify the clustering methods despite various disguises.

b) Yale: The Yale face database[2] contains 165 gray scale images of 15 individuals. Each individual has 11 images, covering center-light, w/glasses, happy, left-light, w/no glasses, normal, right-light, sad, sleepy, surprised and wink. We use Yale dataset to verify the clustering methods despite expression changes.

c) Extended YaleB: This dataset[3] has 38 individuals and around 64 near frontal images under different illuminations for each individual. We use Extended YaleB dataset to verify the clustering methods despite illumination changes.

d) PIE: The PIE[4] is a database of 41 368 images of 68 people. Each person is under 13 different poses and 43 different illuminations, four expressions. We choose the five near frontal poses (C05, C07, C09, C27, C29) and all the images under different illuminations and expressions. So, there are 170 images for each individual, and totally 11 554 images. We use this dataset to verify the clustering methods despite large scale data.

e) PIE-Pose: To construct a dataset only with pose variations, we directly download the dataset used in [44]. They

[1] http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html
[2] http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[3] http://vision.ucsd.edu/leekc/ExtYaleDatabase/ExtYaleB.html
[4] http://vasc.ri.cmu.edu/idb/html/face/

TABLE III
COMPARISON OF DIFFERENT METHODS IN TERMS OF AC, NMI, AND SPEED ON THE BENCHMARK DATASETS

Metric	Dataset	K-means [45]	TPCA+ K-means	MPCA+ K-means	TPCAL1+ HOSVD	LRR [40]	SSC [41]	PSC [38]	LSC-K [39]	CASS [42]	HOSVD [14]	RTC (ours)
AC	**AR**	0.4059	0.4059	0.4069	0.5526	0.5638	0.5354	0.5567	0.5061	0.4545	0.5395	**0.6134**
	Yale	0.5030	0.5273	0.4848	0.5879	0.6000	0.6182	0.6061	0.5879	0.5697	0.5333	**0.6606**
	ExYaleB	0.0972	0.1008	0.1070	0.3599	0.6459	**0.6579**	0.3332	0.2748	0.3149	0.3488	0.3863
	PIE	0.0846	0.0910	0.0815	0.2995	**0.3536**	0.1273	0.1458	0.1542	–	0.2658	0.3254
	PIE-pose	0.1576	0.1727	0.1545	0.1424	0.1909	0.1455	0.1606	0.1697	0.1212	0.1333	**0.2030**
NMI	**AR**	0.6840	0.6875	0.6881	0.7869	0.7748	0.7840	0.7682	0.7485	0.7037	0.7774	**0.8104**
	Yale	0.5391	0.6054	0.5673	0.6006	0.5888	0.6367	0.6137	0.6314	0.6166	0.5829	**0.6531**
	ExYaleB	0.1322	0.1458	0.1411	0.5112	**0.7802**	0.7273	0.3824	0.3622	0.4343	0.4918	0.5303
	PIE	0.1894	0.1971	0.1866	0.4456	**0.5012**	0.3336	0.2518	0.3395	–	0.4101	0.4526
	PIE-pose	0.3202	0.3388	0.3190	0.3001	0.3141	0.3273	0.3644	0.3573	0.2958	0.3012	**0.3765**
Speed (sec)	**AR**	8.18	5.48	3.61	2.65	299.86	527.67	1.22	1.51	706.08	6.90	68.92
	Yale	0.67	0.88	0.60	0.36	15.95	117.45	0.14	0.14	1.82	0.49	6.99
	ExYaleB	31.42	14.86	6.06	8.88	878.40	1484.5	6.11	3.94	1957.39	37.84	117.97
	PIE	480.77	149.71	58.83	200.57	5963.3	10297	187.23	105.93	>100000	864.42	432.56
	PIE-pose	1.68	1.69	0.96	0.50	65.83	181.18	0.32	0.43	49.30	1.09	197.08

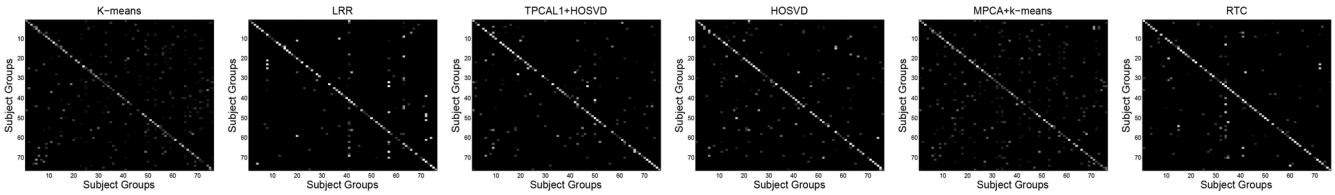


Fig. 2. Visualization of $\mathbf{B} \in \mathbb{R}^{K\times K}$ on AR dataset in terms of different methods. The value $\mathbf{B}_{ij}$ in $\mathbf{B}$ means that how many faces coming from the ith person are grouped into the jth cluster. The larger $\mathbf{B}_{ij}$ is, the brighter the corresponding pixel is. Because most of faces are clustered into the default (correct) cluster and few images are wrongly labeled, the bright pixels mostly locate on the diagonals.

conduct the experiments using a subset of PIE, which contains facial images from the first 30 people with 11 poses and 21 illuminations. Herein, we fix the illumination, and select all the facial images with pose changes. In this way, we obtain a dataset including 330 facial images of 30 people. Each person has 11 images. We call it PIE-pose, and use this dataset to verify the clustering methods despite pose variations.

All the pixel values are scaled into [0,1] in our experiments, and no more preprocessing is applied, i.e., we do not extract features from the facial images.[5]

2) Results: We first test the performance of RTC versus different image sizes. We reshape the facial images on ExYaleB dataset into 16×16, 32×32, and 64×64, respectively. The corresponding AC and NMI output by RTC are reported in Table II. We see 64×64 obtains the best performance while costing a slightly more time. By this, we reshape all the facial images into 64×64 in the following experiments.

Now, we discuss the influence of parameter r to the performance of RTC. Fig. 1 illustrates the parameter tuning results of RTC on five datasets. We see the performance of our method is refining when we increase the value of r, and reaches a relatively stable stage. This is as expected because the approximation cannot retain sufficient information for the original tensor data when r is too small, with the result of poor clustering performance. Instead, the clustering performance of RTC doesn't improve with the increasing of r, which explains that only a little information within the original tensor data is useful, and thus we can use a low-rank approximation to replace the original tensor data to perform face clustering. In addition, we can see that the curve on Yale dataset is less smooth compared with those on PIE, AR, and ExYaleB. The reason for this may be that PIE, AR, and ExYaleB have more images than that on Yale dataset.

Please note that, for PIE-pose dataset, AC, and NMI do not get better with the increasing of r. As mentioned before, RTC attempts to obtain a low-rank representation to improve the clustering performance. This implies that the facial images referring to the same people on this low-rank representation are more similar than the original data. The illuminations, expressions, and disguises are all regarded as some kind of noises, and thus can be removed using the L1 norm maximization in (3). However, this does not stand when the faces meet pose variations. Even so, we still see RTC achieves the best performance when $r = 5$, which shows that the dimension reduction step in RTC is also effective under this case.

Table III lists the comparison results on five datasets. We conduct 30 tests on each dataset and the best performance is reported. To comprehensively evaluate the compared methods, we also give their running time. From the results in Table III, we can draw the following four conclusions. First, PCA clustering methods (TPCA + k-means and MPCA + k-means) achieve better performance than k-means, which demonstrates the effectiveness of PCA dimension reduction. Second, RTC outperforms HOSVD on all six datasets. The improvement varies from 3.75% (ExYaleB) to 12.73% (Yale), and averages at 7.35% for AC (the average is 5.19% for NMI). This demonstrates the effectiveness of our proposed RTC algorithm, which takes HOSVD as its second step. Third, HOSVD achieves better performance than k-means on average. This demonstrates

[5]The Yale, Extended YaleB, and PIE datasets used here are downloaded from the website: http://www.cad.zju.edu.cn/home/dengcai/Data/FaceData.html.

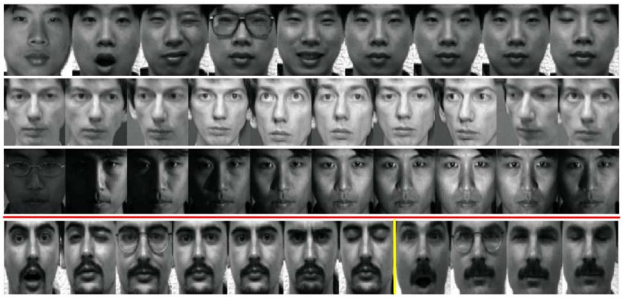


Fig. 3. Some visual clustering results of RTC. Each row denotes a facial cluster output by RTC. The fourth row is a failure case of RTC, where the yellow line separates the faces coming from two different persons. For detailed discussions, please see the text.

Fig. 4. Some visual clustering results of RTC on AR dataset. Each row stands for one cluster. From this figure, we see although the faces referring to the same person wear sun glasses or scarf, RTC can still cluster them together, which demonstrates that RTC is robust to the disguises.

that the matrix-based method can exploit more spatial information than vector-based method, and thus obtain the better clustering performance. Fourth, RTC obtains much better performances than TPCAL1 + HOSVD on all six datasets, which shows the proposed nongreedy strategy can achieve better solution than greedy strategy in [16]. In addition, we see RTC almost outperforms all the compared methods expect LRR and SSC. LRR and SSC both achieve a significantly high AC and NMI on ExYaleB and PIE datasets, while obtains the mediocre performance on other datasets. It shows LRR and SSC are more effective to illumination changes than other noises. Even so, the speeds of LRR and SSC are much slower. Actually, we find all the SSC methods (LRR, SSC, CASS) cost more time than other methods. For CASS, it shows a fast speed on Yale and PIE-pose datasets, while shows a slow speed on AR, ExYaleB, and PIE datasets (on PIE dataset, CASS cannot accomplish the clustering even when the running time is above 100 000 s. In such case, we do not obtain its true performance, and use—instead). This is because the time complexity of CASS is mainly related to the sizes of used datasets. The number of images on Yale and PIE-pose is less than that on AR, ExYaleB, PIE. Therefore, the speeds on Yale and PIE-pose are fast. In contrast, PSC and LSC-K show their advantages in terms of speed. This is as expected because PSC uses the parallel technique, and LSC is proposed to handle the large-scale data. RTC achieves the best performance with a reasonable speed, which demonstrates its competitiveness.

In face clustering, we hope to observe how many faces of one person are clustered into the default (correct) cluster (the remaining faces are clustered into wrong clusters). To this end, we compute the confusion matrix $\mathbf{B} \in \mathbb{R}^{K\times K}$ based on the indicator matrix $\mathbf{Q} \in \mathbb{R}^{n\times K}$ in Algorithm 1, and then visualize it. The value $\mathbf{B}_{ij}$ in $\mathbf{B} \in \mathbb{R}^{K\times K}$ means that how many faces coming from the ith person are grouped into the jth cluster.

Fig. 5. Ten sample images with artificial occlusions on PIE dataset. To better simulate the real-world situation, we add the occlusions with both random locations and sizes to the faces. The size of occlusions located on the left is 20×20 pixels, and the size located on the right is 25×25 pixels.

The formula of computing $\mathbf{B} \in \mathbb{R}^{K\times K}$ can be found in [14]. We visualize $\mathbf{B}_{ij}$ by using different gray levels (0–255). The larger $\mathbf{B}_{ij}$ is, the greater the gray value is, and the brighter the corresponding pixel is. The visualizations for different clustering methods on AR dataset are given in Fig. 2. Because most faces are grouped into the default cluster and few images are wrongly labeled, the bright pixels mostly locate on the diagonals. From the figure, we can see RTC shows better clustering result, i.e., the diagonals are brighter, and less bright pixels are located on the off-diagonal area.

We show the final clustering results of RTC in Fig. 3, where each row denotes a facial cluster output by RTC. From the first to the third row, we see that RTC can successfully cluster the faces with different expressions, poses, and illuminations, respectively. In the fourth row, we give a failure case of RTC, where the yellow line separates the faces coming from two different persons. The reason for this is that the faces referring to the different persons in the fourth row have a lot of similarities (such as the beard, mouth, and shape of face, etc.), leading to the confusion even for the human. Because RTC clusters faces mainly based on their appearance, the result in the forth row is reasonable in the view of RTC. We individually give the clustering results on AR dataset in Fig. 4.

D. Clustering Despite Random Block Occlusions and Pixel Corruptions

In order to evaluate the robustness of RTC to the random block occlusions, we add artificial occlusions on some randomly selected images of the above datasets to generate simulative polluted images like that in [6] and [18]. Then, we test the performance of different methods on these polluted datasets. Ten sample images with artificial occlusions are listed in Fig. 5. We generate polluted images in terms of different ratios. The polluted ratio is defined as the number of polluted images divided by the number of all images of the dataset. Fig. 6 illustrates the AC and NMI comparison results for PSC, LSC-K, HOSVD, LRR with RTC in terms of different polluted ratios on Yale dataset. From Fig. 6, we see that RTC outperforms the LSC-K, PSC, HOSVD, and LRR under different polluted ratios. Specifically, we find that both the PSC and LSC-K are sensitive to occlusions. With the polluted ratio increasing, the performances of PSC and LSC-K descend quickly. In contrast, the curves of RTC are relatively flat. The quantitative results on Yale and ExYaleB datasets are listed in Table IV. We see that RTC obtains the best clustering performance, which effectively demonstrates RTC is more robust to the random block occlusions than other clustering algorithms. RTC loses to LRR and SSC on ExYaleB dataset, which shows that: 1) LRR and SSC are more robust to the

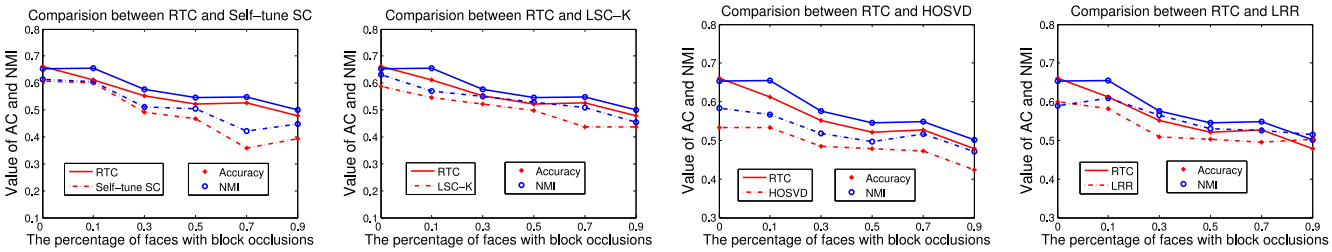


Fig. 6. Comparisons between RTC and PSC (Self-tune SC), LSC-K, HOSVD, LRR versus different percentages of polluted faces on Yale dataset. In each figure, we use the solid line to denote RTC, and use the dotted line to denote the compared methods. Red indicates the performance evaluated by AC, and blue indicates the performance evaluated by NMI.

TABLE IV
COMPARISON OF DIFFERENT METHODS IN TERMS OF AC AND NMI ON THE BENCHMARK DATASETS UNDER RANDOM BLOCK OCCLUSIONS

Metric	Dataset	K-means [45]	TPCA+ K-means	MPCA+ K-means	TPCAL1+ HOSVD	LRR [40]	SSC [41]	PSC [38]	LSC-K [39]	CASS [42]	HOSVD [14]	RTC (ours)
AC	**Yale**	0.4909	0.5758	0.4970	0.5818	0.5818	0.4485	0.5636	0.5636	0.5758	0.5091	**0.6000**
	ExYaleB	0.1097	0.1111	0.0977	0.3550	0.6120	**0.6708**	0.3011	0.2525	0.2507	0.3577	0.3791
NMI	**Yale**	0.5445	0.5877	0.5757	0.6206	0.6093	0.4532	0.5735	0.5945	0.5731	0.5762	**0.6227**
	ExYaleB	0.1458	0.1441	0.1362	0.5051	0.7336	**0.7468**	0.3600	0.3374	0.3875	0.4927	0.5251

Fig. 7. Some clustering results of RTC under random block occlusions. Each row stands for one cluster output by RTC. In the fourth row, we use the red rectangle to mark out the misclassification sample. We see this sample is very similar to its right face, resulting in the error classification. In addition, this figure shows that RTC can successfully cluster the faces referring to the same person even there are some occlusions on these faces.

Fig. 8. Some facial examples with Gaussian noises $\mathcal{N}(0, \sigma)$. From top to bottom row list the noisy faces under $\sigma = \{0.01, 0.05, 0.1, 0.15, 0.2\}$.

illumination changes than RTC and 2) LRR and SSC are less sensitive to the random block occlusions thanks to the used low-rank property and subspace assumption. Even so, from the results in Tables III–VI, we can still say that RTC is better than LRR and SSC. Some visual clustering examples of RTC are given in Fig. 7.

We next evaluate the performance of RTC versus various levels of random pixel corruptions. To this end, we directly add the Gaussian noises with average 0, and variance σ to the faces. The range of σ is $\{0.01, 0.05, 0.1, 0.15, 0.2, 0.25\}$. Some visual examples with different variances are shown in Fig. 8. Fig. 9 illustrates the AC and NMI comparison results for PSC, LSC-K, HOSVD, LRR with RTC in terms of different σ on Yale dataset. From Fig. 9, we also see that the performances of PSC and LSC-K descend quickly with σ increasing, which is similar to the results in Fig. 6. The quantitative results on other datasets are listed in Table V (we show the results under the variance $\sigma = 0.1$). From the table, we can see that RTC obtains the best clustering performance on both the datasets and both the metrics. The average improvement of RTC over CASS on two datasets is 2.2% for AC, and 5.4% for NMI (we choose CASS because it obtains the second best performance among all the compared clustering methods). It effectively demonstrates RTC is more robust to the random pixel corruptions than other clustering algorithms. Some visual examples are given in Fig. 10.

E. Clustering on High-Order Tensor Data

In the above sections, we evaluate RTC on 2-D facial images. Now, we discuss the comparisons on high-order tensor data. We choose color facial images and video clips to conduct this experiments, because they can be naturally represented by the third-order tensor data. Fig. 11 illustrates the tensor-based representations for a color facial image from LFW dataset [46] and a video clip from Gait dataset [47].

For color facial images, the used dataset here is labeled faces in the wild (LFW),[6] which contains 13 233 images covering 5749 people. Because 1680 people in the LFW only have one or two images, which is not suitable for clustering. Therefore, we choose a subset of LFW to conduct the experiments. We just select the persons who have 11 images. In this way, 16 persons and totally 176 images are obtained. We list six samples in Fig. 12, from which we see the facial images on this dataset have noisy background, changing poses, different expressions, and various disguises (such as the athlete with a cap and the

[6]http://vis-www.cs.umass.edu/lfw/

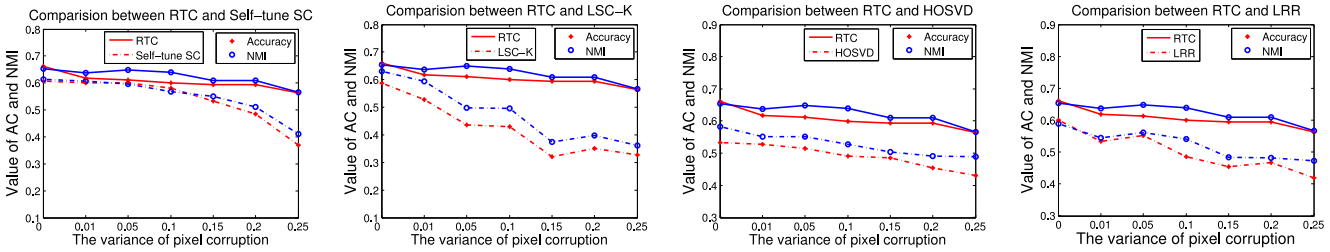


Fig. 9. Comparisons between RTC and PSC (Self-tune SC), LSC-K, HOSVD, LRR versus different variances of pixel corruptions on Yale dataset. In each figure, we use the solid line to denote RTC, and use the dotted line to denote the compared methods. Red indicates the performance evaluated by AC, and blue indicates the performance evaluated by NMI.

TABLE V
COMPARISON OF DIFFERENT METHODS IN TERMS OF AC AND NMI ON THE BENCHMARK DATASETS UNDER RANDOM PIXEL CORRUPTIONS (VARIANCE = 0.1)

Metric	Dataset	K-means [45]	TPCA+ K-means	MPCA+ K-means	TPCAL1+ HOSVD	LRR [40]	SSC [41]	PSC [38]	LSC-K [39]	CASS [42]	HOSVD [14]	RTC (ours)
AC	**Yale**	0.3818	0.5576	0.3455	0.5636	0.4848	0.4727	0.5818	0.4303	0.5818	0.4909	**0.6000**
	ExYaleB	0.1120	0.1111	0.1120	0.2806	0.0562	0.1766	0.1619	0.0977	0.2877	0.2752	**0.3211**
NMI	**Yale**	0.4414	0.5716	0.4019	0.6003	0.5418	0.5075	0.5676	0.4958	0.5975	0.5271	**0.6391**
	ExYaleB	0.1473	0.1534	0.1472	0.4044	0.0552	0.2655	0.2285	0.1438	0.3746	0.3847	**0.4405**

TABLE VI
COMPARISON OF DIFFERENT METHODS IN TERMS OF AC, NMI, AND SPEED ON THE BENCHMARK DATASETS

Metric	Dataset	K-means [45]	TPCA+ K-means	MPCA+ K-means	TPCAL1+ HOSVD	LRR [40]	SSC [41]	PSC [38]	LSC-K [39]	CASS [42]	HOSVD [14]	RTC (ours)
AC	**Gait**	0.2818	0.2996	0.2544	–	0.3653	0.3542	0.3393	0.3447	0.3625	0.3352	**0.3653**
	LFW	0.2557	0.2955	0.2898	–	0.1932	0.3352	0.2955	0.3239	0.3182	0.2841	**0.3523**
NMI	**Gait**	0.5299	0.5599	0.5130	–	0.5581	0.5843	0.6000	0.5988	0.5976	0.5755	**0.6008**
	LFW	0.2880	0.3655	0.3247	–	0.1606	0.3576	0.3687	0.3890	0.3825	0.3319	**0.4042**
Speed (sec)	**Gait**	7.70	7.31	7.96	–	249.69	876.90	1.19	0.95	238.49	4.15	15.68
	LFW	7.03	9.05	1.98	–	128.21	220.35	0.54	0.65	3.80	4.84	18.55

Fig. 10. Some visual clustering examples of RTC. Each row stands for one RTC cluster. RTC can handle the faces with different levels of random pixel corruptions, even under the case with a lot of noise, where the human may not be able to distinguish them.

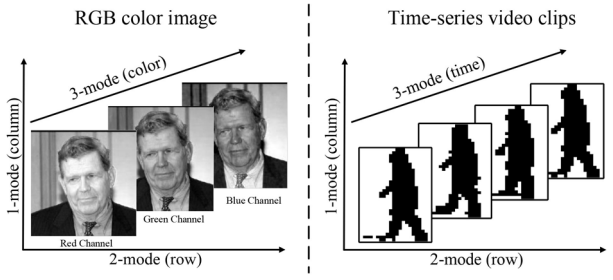


Fig. 11. Illustration of the tensor-based representations for color facial image and video clips.

man with a sunglass). We also reshape each image into 64×64, and thus obtain a tensor data $\mathcal{X} \in \mathbb{R}^{64\times64\times3\times176}$. For video clips, like [43], we use the USF HumanID "Gait challenge"

Fig. 12. Six sample images from LFW dataset, from which we see that the facial images on this dataset have noisy background, changing poses, different expressions, and various disguises.

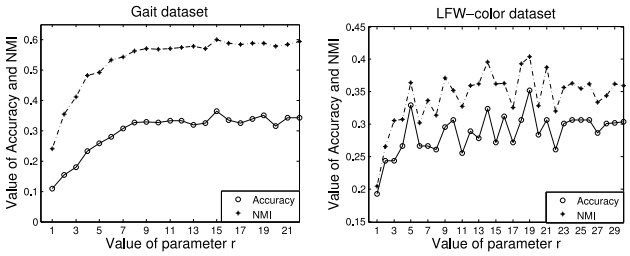


Fig. 13. Performance of RTC in terms of AC and NMI with respect to different r on Gait dataset and LFW dataset.

dataset. This dataset contains 71 sequences covering seven subsets (Probe A–G). For simplicity, we select Probe A to conduct our experiments,[7] and thus obtain 727 videos. Each video clip has ten frames, and each frame is reshaped into 32×22 size. Thus, the tensor data $\mathcal{X} \in \mathbb{R}^{32\times22\times10\times727}$ is obtained.

[7]We directly use the processed Gait challenge dataset downloaded from http://www.comp.hkbu.edu.hk/~haiping/

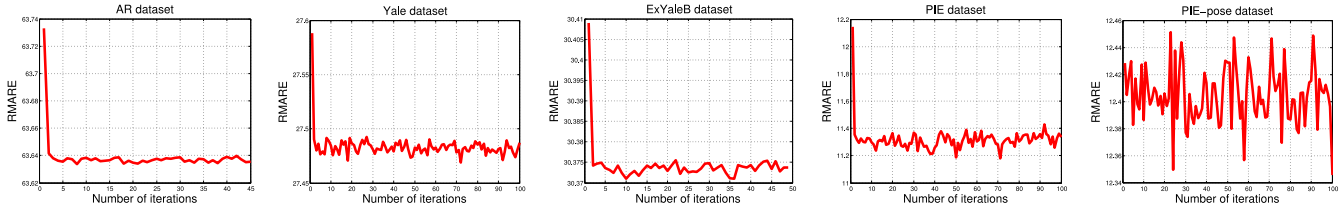


Fig. 14. Convergence curves of RTC on AR, Yale, ExYaleB, PIE, and PIE-pose datasets, respectively. For detailed discussions, please see the text.

$\mathcal{X} \in \mathbb{R}^{64\times64\times3\times176}$ and $\mathcal{X} \in \mathbb{R}^{32\times22\times10\times727}$ are directly input to Algorithm 1 to accomplish the clustering tasks.

Similarly, we first discuss the influence of parameter r to the performance of RTC like 2-D case. For LFW dataset, we let $r_1 = r_2 = r$ and tune r from 1 to 30. As for r_3, we simply let $r_3 = 3$. For Gait dataset, we set $r_3 = 10$, and tune r from 1 to 22. The parameter tuning results are listed in Fig. 13. We see the curves are similar with the 2-D case. The curve on LFW dataset is less smooth because LFW dataset has fewer samples than Gait dataset. The quantitative results are given in Table V. Because TPCAL1 only introduces the solution of 2-D case, we do not give their performances here. From the table, we see RTC achieves the best performances. Although LRR and CASS have the similar performance with RTC on Gait dataset, as expected, they cost more time than other methods, which leads to be unavailable in the real-world applications.

We now evaluate the convergence of Algorithm 1. The experimental results on AR, Yale, ExYaleB, PIE, and PIE-pose datasets are drawn in Fig. 14. We see that RTC converges within two or three iterations on the left four datasets, while doesn't converge on PIE-pose dataset. The reason for this has been analyzed in Section V-C2, i.e., the pose variations are not satisfied with the low-rank assumption, resulting in that RTC cannot find an optimal rank.

VI. Conclusion

In this paper, we proposed the RTC method. Experiments conducted on gait sequences and six benchmark facial datasets showed that RTC were able to handle different expressions, illuminations and disguises within faces. RTC also got better performance when the datasets met block occlusions and pixel corruptions, which demonstrated that RTC was robust to noises. Although RTC cannot deal with pose variations, it still achieved the best performance compared with other clustering methods. Moreover, as a tensor-based method, RTC could exploit more spatial information of facial images, and thus achieved better performance when faces reserved sufficient spatial information. In addition, we utilized a nongreedy maximization algorithm to solve the RTC problem. Experiments showed that our algorithm could quickly converge within two or three iterations. In the future, we plan to integrate some contextual cues (such as the text description of faces) into the framework of RTC, and thus improve the discrimination of tensor-based representation for faces. Making RTC robust to pose changes is also a direction.

References

[1] S. Eickeler, F. Wallhoff, U. Lurgel, and G. Rigoll, "Content based indexing of images and video using face detection and recognition methods," in *Proc. Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Salt Lake City, UT, USA, 2001, pp. 1505–1508.

[2] C. Czirjek, S. Marlow, and N. Murphy, "Face detection and clustering for video indexing applications," in *Proc. Adv. Concepts Intell. Vis. Syst.*, Ghent, Belgium, 2003, pp. 2–5.

[3] A. Fitzgibbon and A. Zisserman, "On affine invariant clustering and automatic cast listing in movies," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Copenhagen, Denmark, 2002, pp. 304–320.

[4] C. M. Bishop and N. M. Nasrabadi, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[5] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. Hoboken, NJ, USA: CRC Press, 2013.

[6] J. Wright, A. Y. Yang, A. Ganesh, S. S. Sastry, and Y. Ma, "Robust face recognition via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 2, pp. 210–227, Feb. 2009.

[7] J. Lu and Y.-P. Tan, "Regularized locality preserving projections and its extensions for face recognition," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 3, pp. 958–963, Jun. 2010.

[8] M. Song, D. Tao, Z. Liu, X. Li, and M. Zhou, "Image ratio features for facial expression recognition application," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 3, pp. 779–788, Jun. 2010.

[9] P.-H. Lee, G.-S. Hsu, Y.-W. Wang, and Y.-P. Hung, "Subject-specific and pose-oriented facial features for face recognition across poses," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 5, pp. 1357–1368, Oct. 2012.

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

[11] Z. Ma, Y. Yang, F. Nie, and N. Sebe, "Thinking of images as what they are: Compound matrix regression for image classification," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, Beijing, China, 2013, pp. 1530–1536.

[12] T.-K. Kim, K.-Y. K. Wong, and R. Cipolla, "Tensor canonical correlation analysis for action classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Minneapolis, MN, USA, 2007, pp. 1–8.

[13] W. Guo, I. Kotsia, and I. Patras, "Tensor learning for regression," *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 816–827, Feb. 2012.

[14] H. Huang, C. Ding, D. Luo, and T. Li, "Simultaneous tensor subspace selection and clustering: The equivalence of high order SVD and k-means clustering," in *Proc. Int. Conf. Knowl. Disc. Data Min. (KDD)*, Las Vegas, NV, USA, 2008, pp. 327–335.

[15] S. Jegelka, S. Sra, and A. Banerjee, "Approximation algorithms for tensor clustering," in *Algorithmic Learning Theory*. Berlin, Germany: Springer, 2009, pp. 368–383.

[16] Y. Pang, X. Li, and Y. Yuan, "Robust tensor analysis with L1-norm," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 20, no. 2, pp. 172–178, Feb. 2010.

[17] 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.

[18] N. Kwak, "Principal component analysis based on L1-norm maximization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 9, pp. 1672–1680, Sep. 2008.

[19] F. Nie, H. Huang, C. Ding, D. Luo, and H. Wang, "Robust principal component analysis with non-greedy L1-norm maximization," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, Barcelona, Spain, 2011, pp. 1433–1438.

[20] X. Cao, X. Wei, Y. Han, Y. Yang, and D. Lin, "Robust tensor clustering with non-greedy maximization," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, Beijing, China, 2013, pp. 1254–1259.

[21] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cogn. Neurosci.*, vol. 3, no. 1, pp. 71–86, 1991.

[22] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.

[23] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced Fisher linear discriminant model for face recognition," *IEEE Trans. Image Process.*, vol. 11, no. 4, pp. 467–476, Apr. 2002.

[24] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns: Application to face recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 12, pp. 2037–2041, Dec. 2006.

[25] N.-S. Vu and A. Caplier, "Face recognition with patterns of oriented edge magnitudes," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Crete, Greece, 2010, pp. 313–326.

[26] N.-S. Vu, "Exploring patterns of gradient orientations and magnitudes for face recognition," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 2, pp. 295–304, Feb. 2013.

[27] Y. Meng, L. Zhang, S. Shiu, and D. Zhang, "Monogenic binary coding: An efficient local feature extraction approach to face recognition," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 6, pp. 1738–1751, Dec. 2012.

[28] P. Wu and F. Tang, "Improving face clustering using social context," in *Proc. ACM Int. Conf. Multimedia*, Florence, Italy, 2010, pp. 907–910.

[29] W. Zhang, T. Zhang, and D. Tretter, "Beyond face: Improving person clustering in consumer photos by exploring contextual information," in *Proc. Int. Conf. Multimedia Expo. (ICME)*, Singapore, 2010, pp. 1540–1545.

[30] N. Kumar, A. Berg, P. N. Belhumeur, and S. Nayar, "Describable visual attributes for face verification and image search," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 10, pp. 1962–1977, Oct. 2011.

[31] L. Zhang, D. V. Kalashnikov, and S. Mehrotra, "A unified framework for context assisted face clustering," in *Proc. Int. Conf. Multimedia Retrieval (ICMR)*, Dallas, TX, USA, 2013, pp. 9–16.

[32] B. Wu, Y. Zhang, B.-G. Hu, and Q. Ji, "Constrained clustering and its application to face clustering in videos," in *Proc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Portland, OR, USA, 2013, pp. 3507–3514.

[33] X. Li, Y. Pang, and Y. Yuan, "L1-norm-based 2DPCA," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 40, no. 4, pp. 1170–1175, Aug. 2010.

[34] K. Inoue, K. Hara, and K. Urahama, "Robust multilinear principal component analysis," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Kyoto, Japan, 2009, pp. 591–597.

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

[36] J. Ye, "Generalized low rank approximations of matrices," *Mach. Learn.*, vol. 61, no. 1, pp. 167–191, 2005.

[37] D. Cai, X. He, and J. Han, "Document clustering using locality preserving indexing," *IEEE Trans. Knowl. Data Eng.*, vol. 17, no. 12, pp. 1624–1637, Dec. 2005.

[38] W. Chen, Y. Song, H. Bai, C. Lin, and E. Chang, "Parallel spectral clustering in distributed systems," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 568–586, Mar. 2011.

[39] X. Chen and D. Cai, "Large scale spectral clustering with landmark-based representation," in *Proc. Conf. Artif. Intell. (AAAI)*, San Francisco, CA, USA, 2011, pp. 313–318.

[40] G. Liu *et al.*, "Robust recovery of subspace structures by low-rank representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 171–184, Jan. 2013.

[41] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 11, pp. 2765–2781, Nov. 2013.

[42] C. Lu, J. Feng, Z. Lin, and S. Yan, "Correlation adaptive subspace segmentation by trace lasso," in *Proc. Int. Conf. Comput. Vis. (ICCV)*, Sydney, VIC, Australia, 2013, pp. 1–8.

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

[44] Y. Chen, C. Hsu, and H. Liao, "Simultaneous tensor decomposition and completion using factor priors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 577–591, Mar. 2014.

[45] C. Deng. (2011). *Litekmeans: The Fastest MATLAB Implementation of k-means*. [Online]. Available: http://www.zjucadcg.cn/dengcai/Data/Clustering.html

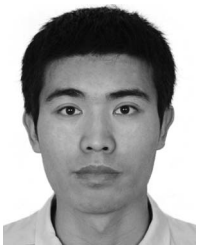
[46] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Dept. Comp. Sci., University of Massachusetts Amherst, Amherst, MA, USA, Tech. Rep. 07-49, 2007.

[47] S. Sarkar *et al.*, "The humanID gait challenge problem: Data sets, performance, and analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 2, pp. 162–177, Feb. 2005.

Xiaochun Cao (SM'14) received the B.E. and M.E. degrees from Beihang University, Beijing, China, and the Ph.D. degree from the University of Central Florida, Orlando, FL, USA, all in computer science.

He was a Research Scientist at ObjectVideo, Inc., Reston, VA, USA, for three years. From 2008 to 2012, he was a Professor at Tianjin University, Tianjin, China. He is a Professor with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing. He has authored and co-authored over 80 journal and conference papers.

Mr. Cao was the recipient of the Piero Zamperoni Best Student Paper Award at the International Conference on Pattern Recognition in 2004 and 2010 and nominated for the university-level Outstanding Dissertation Award for his dissertation.



Xingxing Wei received the B.S. degree from the School of Automation and Electrical Engineering, Beihang University, Beijing, China, in 2010. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Tianjin University, Tianjin, China, under the guidance of Prof. X. Cao.

His current research interests include structured learning and prediction, tensor analysis and its applications, image retrieval, saliency detection, and object recognition.

Yahong Han received the Ph.D. degree from Zhejiang University, Hangzhou, China.

He is an Associate Professor with the School of Computer Science and Technology, Tianjin University, Tianjin, China. His current research interests include multimedia analysis, retrieval, and machine learning.

Dongdai Lin received the M.S. and Ph.D. degrees in fundamental mathematics from the Institute of Systems Science, Chinese Academy of Sciences, Beijing, China, in 1987 and 1990, respectively.

He was an Associate Professor, from 1993 to 1998, and a Professor, from 1998 to 2001, with the Institute of Systems Science, Singapore, and a Professor with the Institute of Software, Chinese Academy of Sciences from 2001 to 2011. He is currently a Professor with the Institute of Information Engineering, Chinese Academy of Sciences, and the Director of the State Key Laboratory of Information Security, Beijing. His current research interests include cryptology, security protocols, symbolic computation and software development, multivariate public key cryptography, sequences and stream cipher, zero knowledge proof, and network-based cryptographic computation. He has published over 150 research papers in journals and conference proceedings.