Robust Tensor Analysis with Non-Greedy ℓ_1-Norm Maximization

Limei ZHAO, Weimin JIA, Rong WANG, Qiang YU

Xi'an Research Institute of Hi-Tech, Xi'an, China.

wangrong07@tsinghua.org.cn

Manuscript received January 26, 2015

Abstract. *The ℓ_1-norm based tensor analysis (TPCA-L1) is recently proposed for dimensionality reduction and feature extraction. However, a greedy strategy was utilized for solving the ℓ_1-norm maximization problem, which makes it prone to being stuck in local solutions. In this paper, we propose a robust TPCA with non-greedy ℓ_1-norm maximization (TPCA-L1 non-greedy), in which all projection directions are optimized simultaneously. Experiments on several face databases demonstrate the effectiveness of the proposed method.*

Keywords

Principal component analysis (PCA), TPCA, ℓ_1-norm, outliers, non-greedy strategy

1. Introduction

Principal component analysis (PCA) is a widely used dimensionality reduction and feature extraction method due to its simplicity and effectiveness [1–3]. In the domain of image analysis, traditional PCA algorithms reshaped the original two-demensional image into a one-dimensional long vector and aimed to find the optimal projection directions in which the reconstruction error was minimized. However, the intrinsic spatial structural information was damaged. 2DPCA [4] was proposed to alleviate this problem, which can essentially be seen as PCA in matrix forms. In order to capture more spatial information, tensor analysis [5], [6] (also named multilinear subspace analysis) was introduced into PCA, in which an image can be treated as a second-order tensor and a sequence of images as a third-order tensor. Tensorface [7] is one of the earliest tensor analysis methods for face recognition. The main idea of tensorface is high-order singular value decomposition (HOSVD). Generalized low rank approximation (GLRAM) [8] was proposed to solve the problem of low rank approximations of matrices, in which each image is represented as a two-order tensor.

It is known that PCA and 2DPCA are prone to outliers due to the utilization of ℓ_2-norm. By contrast, the ℓ_1-norm is more robust to outliers than the ℓ_2-norm. In order to alleviate the effect of outliers, some ℓ_1-norm based methods were proposed, including L1-PCA [9], R1-PCA [10] and PCA-L1 [11]. Among them, PCA-L1 is invariant to rotations and also robust to outliers. Based on the work of Ye [8], Pang proposes the TPCA-L1 algorithm using ℓ_1-norm instead of Frobenius norm [12]. To solve the ℓ_1-norm maximization problem, these algorithms use a greedy strategy which makes them easy to being stuck in local solutions. In 2011, Nie and Huang [13] propose a PCA with non-greedy ℓ_1-norm maximization (PCA-L1 non-greedy), in which all projection directions are optimized simultaneously without increasing computational complexity. Then, Wang extends 2DPCA-L1 to its non-greedy version (2DPCA-L1 non-greedy) [14]. In this paper, we propose a tensor principal component analysis with non-greedy ℓ_1-norm maximization termed as TPCA-L1 non-greedy. It has three major advantages: 1) It is robust to outliers due to the utilization of ℓ_1-norm; 2) more spatial structure information is preserved compared with PCA-L1; 3) all projection directions can be optimized simultaneously and much better recognition accuracy can be obtained than that of TPCA-L1 without increasing the computational cost.

The rest of this paper is organized as follows: In Sec. 2, we give a brief review of the work of TPCA-L1 greedy algorithm [12]. Then we propose the tensor principal component analysis with non-greedy ℓ_1-norm maximization in Sec. 3. The experimental results are reported and an analysis of the experimental results is given in Sec. 4. Finally Sec. 5 concludes the paper and points out the future work.

2. Brief Review of TPCA-L1

Let $\{\boldsymbol{X}_1, \cdots, \boldsymbol{X}_n\}$ be a sequence of image matrices, where n is the number of images. The size of the matrix $\boldsymbol{X}_i$ is $h \times w$, where h and w represent the image height and width, respectively. In TPCA-L1 [12], an effective optimization algorithm was proposed to find two r-dimensional projection matrices U and V that maximize the ℓ_1-norm based dispersion, i.e.,

$$\max_{\boldsymbol{U},\boldsymbol{V}} \sum_{i=1}^{n} \left\| \boldsymbol{V}^T \boldsymbol{X}_i \boldsymbol{U} \right\|_1, \quad (1)$$

subject to $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_r$, $\boldsymbol{V}^T\boldsymbol{V} = \boldsymbol{I}_r$.

DOI: 10.13164/re.2016.0200

The alternative projection optimization procedure can be repeated until converges, which is proved in [8]. The procedure of how to compute $\boldsymbol{v}$ while $\boldsymbol{u}$ is fixed and then compute $\boldsymbol{u}$ while $\boldsymbol{v}$ is fixed is described in detail.

2.1 Compute $\boldsymbol{v}$ while $\boldsymbol{u}$ is Fixed

Beginning with $r = 1$, equation (1) can be rewritten as

$$\max \boldsymbol{g}(\boldsymbol{u}, \boldsymbol{v}) = \max_{\boldsymbol{u}_1, \boldsymbol{v}_1} \sum_{i=1}^{n} \left\| \boldsymbol{v}_1^T \boldsymbol{X}_i \boldsymbol{u}_1 \right\|_1, \tag{2}$$

subject to $\boldsymbol{u}^T \boldsymbol{u} = 1$, $\boldsymbol{v}^T \boldsymbol{v} = 1$.

Define a polarity function $\boldsymbol{p}_i(t)$ to solve the ℓ_1-norm maximization, of which the value is either -1 or 1. The value t in the polarity function indicates iteration number. The polarity function $\boldsymbol{p}_i(t)$ is defined as

$$\boldsymbol{p}_i(t) = \begin{cases} 1, & [\boldsymbol{v}(t)]^T (\boldsymbol{X}_i \boldsymbol{u}_1) > 0, \\ -1, & [\boldsymbol{v}(t)]^T (\boldsymbol{X}_i \boldsymbol{u}_1) \leq 0. \end{cases} \tag{3}$$

So $\boldsymbol{g}(\boldsymbol{u}, \boldsymbol{v})$ can be converted to

$$\begin{aligned} \boldsymbol{g}_{\boldsymbol{u}_1}(\boldsymbol{v}(t)) &= \sum_{i=1}^{n} \left| [\boldsymbol{v}(t)]^T \boldsymbol{X}_i \boldsymbol{u}_1 \right| \\ &= \sum_{i=1}^{n} \boldsymbol{p}_i(t) [\boldsymbol{v}(t)]^T (\boldsymbol{X}_i \boldsymbol{u}_1). \end{aligned} \tag{4}$$

The projection vector $\boldsymbol{v}(t+1)$ at the $(t+1)$th iteration is updated as

$$\boldsymbol{v}(t+1) = \frac{\sum_{i=1}^{n} \boldsymbol{p}_i(t)(\boldsymbol{X}_i \boldsymbol{u}_1)}{\left\| \sum_{i=1}^{n} \boldsymbol{p}_i(t)(\boldsymbol{X}_i \boldsymbol{u}_1) \right\|_2}. \tag{5}$$

Let $t = t + 1$ and repeat (3) and (5), the iteration then runs until converges.

2.2 Compute $\boldsymbol{u}$ while $\boldsymbol{v}$ is Fixed

In the section above, the iteration procedure calculates the optimal $\boldsymbol{v}$ while $\boldsymbol{u}$ is fixed. After the optimal $\boldsymbol{v}$ is obtained, the task becomes computing the optimal $\boldsymbol{u}$ that maximize

$$\begin{aligned} \boldsymbol{g}_{\boldsymbol{v}_1}(\boldsymbol{u}(t)) &= \sum_{i=1}^{n} \left| \boldsymbol{v}_1^T \boldsymbol{X}_i \boldsymbol{u}(t) \right| \\ &= \sum_{i=1}^{n} \boldsymbol{s}_i(t) \left[\boldsymbol{v}_1^T \boldsymbol{X}_i \right] \boldsymbol{u}(t) \end{aligned} \tag{6}$$

where the polarity function $\boldsymbol{s}_i(t)$ is defined as

$$\boldsymbol{s}_i(t) = \begin{cases} 1, & \left(\boldsymbol{v}_1^T \boldsymbol{X}_i \right) \boldsymbol{u}(t) > 0. \\ -1, & \left(\boldsymbol{v}_1^T \boldsymbol{X}_i \right) \boldsymbol{u}(t) \leq 0. \end{cases} \tag{7}$$

The updating rule for $\boldsymbol{u}$ is

$$\boldsymbol{u}(t+1) = \frac{\sum_{i=1}^{n} \boldsymbol{s}_i(t) \left(\boldsymbol{X}_i^T \boldsymbol{v}_1 \right)}{\left\| \sum_{i=1}^{n} \boldsymbol{s}_i(t) \left(\boldsymbol{X}_i^T \boldsymbol{v}_1 \right) \right\|_2}. \tag{8}$$

Then equations (7) and (8) should be repeated until the iteration runs convergence.

2.3 Compute $\boldsymbol{u}_k$ and $\boldsymbol{v}_k$ Based on $\boldsymbol{u}_{k-1}$ and $\boldsymbol{v}_{k-1}$ Respectively

From (5) we can find that $\boldsymbol{v}_1$ is a linear combination of $\boldsymbol{X}_i \boldsymbol{u}$ $(i = 1, \cdots, n)$. To compute $\boldsymbol{v}_k$ $(k > 1)$, $\boldsymbol{X}_i \boldsymbol{u}$ should be updated as a whole

$$(\boldsymbol{X}_i \boldsymbol{u}) = (\boldsymbol{X}_i \boldsymbol{u}) - \boldsymbol{v}_{k-1} \boldsymbol{v}_{k-1}^T (\boldsymbol{X}_i \boldsymbol{u}). \tag{9}$$

Similarly, to calculate $\boldsymbol{u}_k$ where $k > 1$, one has to update $\boldsymbol{v}^T \boldsymbol{X}_i$ by

$$\left(\boldsymbol{v}^T \boldsymbol{X}_i \right) = \left(\boldsymbol{v}^T \boldsymbol{X}_i \right) - \left(\boldsymbol{v}^T \boldsymbol{X}_i \right) \boldsymbol{u}_{k-1} \boldsymbol{u}_{k-1}^T. \tag{10}$$

The TPCA-L1 algorithm does not need to perform eigendecomposition of covariance-like matrices. If the covariance-like matrices are in a large size, the computational cost of traditional tensor analysis algorithms will normally be very large in most cases. However this algorithm optimizes all projection directions sequentially, which makes it easy to being stuck in local solutions.

3. TPCA with Non-Greedy ℓ_1-Norm Maximization

In this section, a general ℓ_1-norm maximization problem is discussed at first, and then TPCA is extended to its non-greedy version. The general ℓ_1-norm maximization problem is formulated as follows:

$$\max_{v \in C} f(v) + \sum_{i=1}^{n} |g_i(v)| \tag{11}$$

which is assumed to have a upper bound. Without loss of generality, $f(\cdot)$ and $g(\cdot)$ are two random functions. Equation (11) can be rewritten as

$$\max_{v \in C} f(v) + \sum_{i=1}^{n} p_i g_i(v) \tag{12}$$

where $p_i = \text{sgn}(g_i(v))$, and $\text{sgn}(\cdot)$ is a sign function defined as $\text{sgn}(x) = 1$ if $x > 1$, $\text{sgn}(x) = -1$ if $x < 1$, and $\text{sgn}(x) = 0$ if $x = 0$. An effective algorithm to solve this general ℓ_1-norm maximization problem has been detailed in [13], which is summarized in Appendix: Algorithm 1.

In each iteration, p_i is calculated by current v^t, and next solution v^{t+1} is updated with the current p_i. The iterative procedure is repeated until the algorithm converges. It has been proved that Algorithm 1 will monotonically increase the objective function of (11) in each iteration, and usually converge to a local solution [13].

Then, we will focus on how to extend the TPCA-L1 algorithm to its non-greedy version. The original problem of TPCA-L1 is to minimize the reconstruction error as follows

$$\min \sum_{i=1}^{n} \left\| X_i - VY_iU^T \right\|_1, \tag{13}$$

subject to $U^TU = I_r$, $V^TV = I_r$, where

$$U = [u_1, u_2, \cdots, u_r] \in \mathbb{R}^{w\times r}$$

and

$$V = [v_1, v_2, \cdots, v_r] \in \mathbb{R}^{h\times r}$$

stand for two orthogonal projection matrices. The problem is corresponding to the following equation:

$$\max \sum_{i=1}^{n} \left\| V^T X_i U \right\|_1, \tag{14}$$

subject to $U^TU = I_r$, $V^TV = I_r$.

In this paper, an alternative projection optimal algorithm is proposed to solve the ℓ_1-norm maximization. When U is fixed, one can compute V, then when V is fixed, compute U. So we extend TPCA-L1 to its non-greedy version from two sides. Here is the description of how to compute U and V, respectively.

3.1 Compute V while U is Fixed

While U is fixed, the problem (1) can be rewritten as

$$\begin{aligned}\arg\max \sum_{i=1}^{n} \left\| V^T X_i U \right\|_1 &= \arg\max \sum_{i=1}^{n} \left\| V^T Y1^{(i)} \right\|_1 \\ &= \arg\max \sum_{i=1}^{n} \sum_{j=1}^{r} \left\| V^T y1_j^{(i)} \right\|_1\end{aligned} \tag{15}$$

where the matrix $Y1^{(i)} = X_iU$, $y1_j^{(i)}$ represents the jth column vector of matrix $Y1^{(i)}$. If define the training matrix as $Y = \left[y1_1^{(1)}, y1_2^{(1)}, \cdots, y1_r^{(1)}, y1_1^{(2)}, \cdots, y1_r^{(n)}\right] \in \mathbb{R}^{h\times nr}$, equation (15) can be converted to

$$\begin{aligned}\arg\max \sum_{i=1}^{n} \left\| V^T X_i U \right\|_1 &= \arg\max \sum_{i=1}^{n} \sum_{j=1}^{r} \left(p^{(i)}(j)\right)^T V^T y1_j^{(i)} \\ &= \arg\max \mathrm{Tr}\left(V^T M\right)\end{aligned} \tag{16}$$

where $p^{(i)}(j) = \mathrm{sgn}\left(V^T y1_j^{(i)}\right)$, $M = YP^T \in \mathbf{R}^{h\times r}$ and $P = \mathrm{sgn}\left(V^T Y\right) \in \mathbf{R}^{r\times nr}$. Suppose the singular value decomposition (SVD) of M as $M = L\Lambda R^T$, where $L \in \mathbb{R}^{h\times h}$, $\Lambda \in \mathbb{R}^{h\times r}$, $R \in \mathbb{R}^{r\times r}$. Then one can get

$$\begin{aligned}\mathrm{Tr}\left(V^T M\right) &= \mathrm{Tr}\left(V^T L\Lambda R^T\right) \\ &= \mathrm{Tr}\left(\Lambda R^T V^T L\right) \\ &= \mathrm{Tr}(\Lambda Z)\end{aligned} \tag{17}$$

where $Z = R^T V^T L \in \mathbb{R}^{r\times h}$, λ_{ii} and z_{ii} are the (i, i)th element of matrices Λ and Z respectively. Note that $ZZ^T = I_r$, where I_r is r dimensional identity matrix, so $z_{ii} \le 1$. While $\lambda_{ii} \ge 0$ for that λ_{ii} is ith singular value of M. That is to say, when $Z = [I_r, 0]$, $\mathrm{Tr}\left(V^T M\right)$ reaches the maximum. So the optimal solution is

$$V = LZ^T R^T = L[I_r, \mathbf{0}]R^T. \tag{18}$$

3.2 Compute U while V is Fixed

Similarly while V is fixed, the problem (1) can be rewritten as

$$\begin{aligned}\arg\max \sum_{i=1}^{n} \left\| V^T X_i U \right\|_1 &= \arg\max \sum_{i=1}^{n} \left\| \left(V^T X_i U\right)^T \right\|_1 \\ &= \arg\max \sum_{i=1}^{n} \left\| U^T X_i^T V \right\|_1 \\ &= \arg\max \sum_{i=1}^{n} \left\| U^T Y2^{(i)} \right\|_1 \\ &= \arg\max \sum_{i=1}^{n} \sum_{j=1}^{r} \left\| U^T y2_j^{(i)} \right\|_1\end{aligned} \tag{19}$$

where the matrix $Y2^{(i)} = X_i^T V$, $y2_j^{(i)}$ represents the jth column vector of matrix $Y2^{(i)}$. Similarly define the training matrix as $Y = \left[y2_1^{(1)}, y2_2^{(1)}, \cdots, y2_r^{(1)}, y2_1^{(2)}, \cdots, y2_r^{(n)}\right] \in \mathbb{R}^{w\times nr}$, equation (19) can be converted to

$$\begin{aligned}\arg\max \sum_{i=1}^{n} \left\| V^T X_i U \right\|_1 &= \arg\max \sum_{i=1}^{n} \sum_{j=1}^{r} \left(s^{(i)}(j)\right)^T U^T y2_j^{(i)} \\ &= \arg\max \mathrm{Tr}\left(U^T M\right)\end{aligned} \tag{20}$$

where $s^{(i)}(j) = \mathrm{sgn}\left(U^T y2_j^{(i)}\right)$, $M = YS^T \in \mathbf{R}^{w\times r}$ and $S = \mathrm{sgn}\left(U^T Y\right) \in \mathbf{R}^{r\times nr}$. Suppose the SVD of M as $M = L\Lambda R^T$, where $L \in \mathbb{R}^{h\times h}$, $\Lambda \in \mathbb{R}^{h\times r}$, $R \in \mathbb{R}^{r\times r}$. Then we have

$$\begin{aligned}\mathrm{Tr}\left(U^T M\right) &= \mathrm{Tr}\left(U^T L\Lambda R^T\right) \\ &= \mathrm{Tr}\left(\Lambda R^T U^T L\right) \\ &= \mathrm{Tr}(\Lambda Z)\end{aligned} \tag{21}$$

where $Z = R^T U^T L \in \mathbb{R}^{r\times h}$, λ_{ii} and z_{ii} are the (i, i)th element of matrices Λ and Z respectively. Similarly to the method to compute V, the optimal solution of U can be calculated:

$$U = LZ^T R^T = L[I_r, \mathbf{0}]R^T. \tag{22}$$

The whole procedure is summarized in Appendix: Algorithm 2.

4. Experimental Results

In this section, four public image databases are selected for performance evaluation. The brief description of the four databases is listed as following [6], [15]:

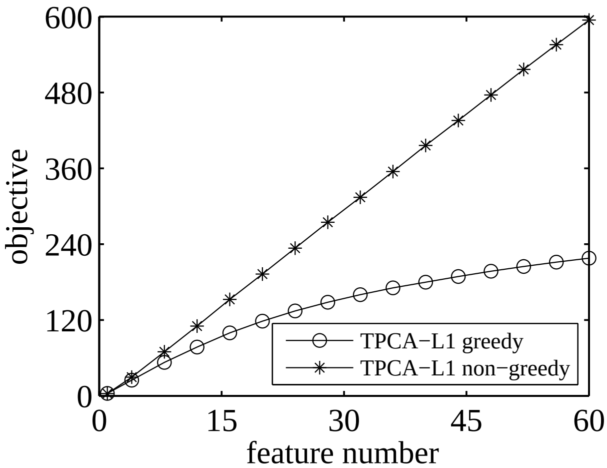
(a) Yale Face Database

(b) ORL Face Database

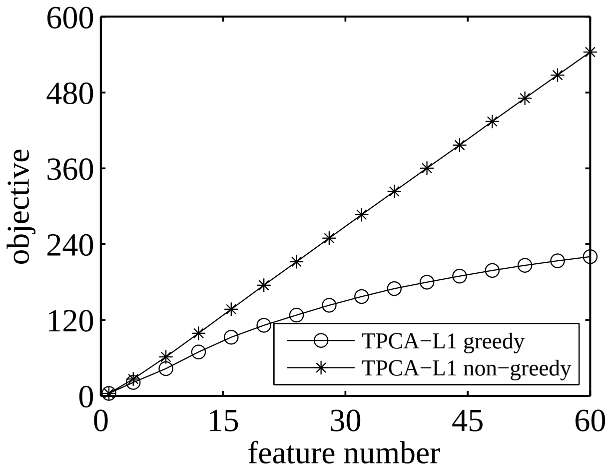
(c) FERET Face Database

(d) USPS Handwritten Digits Database

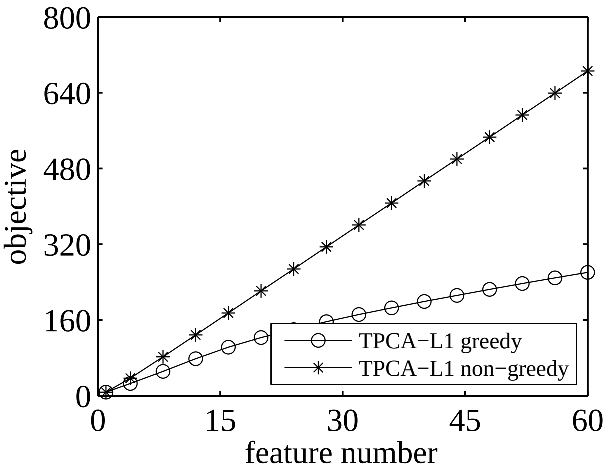
Fig. 1. Sample images from the four databases used in the experiments.



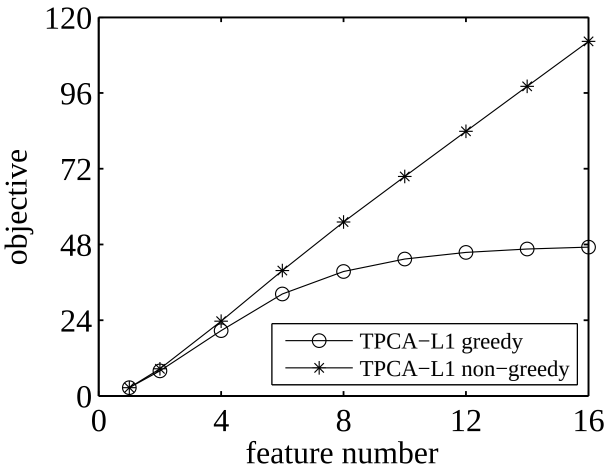
(a) Yale Face Database



(b) ORL Face Database

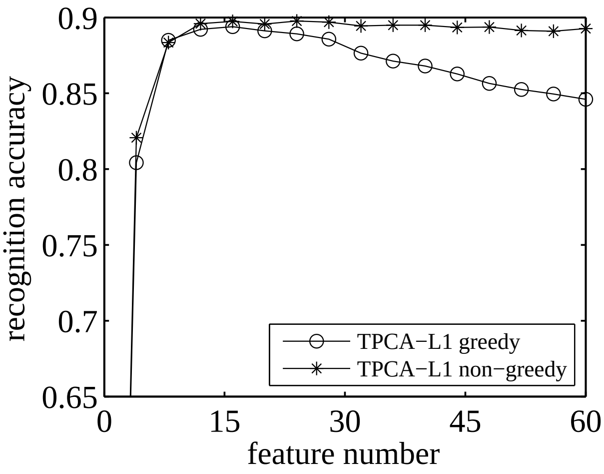


(c) FERET Face Database

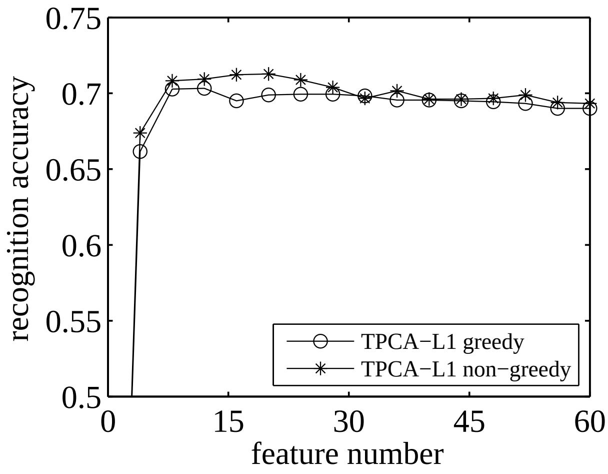


(d) USPS Handwritten Digits Database

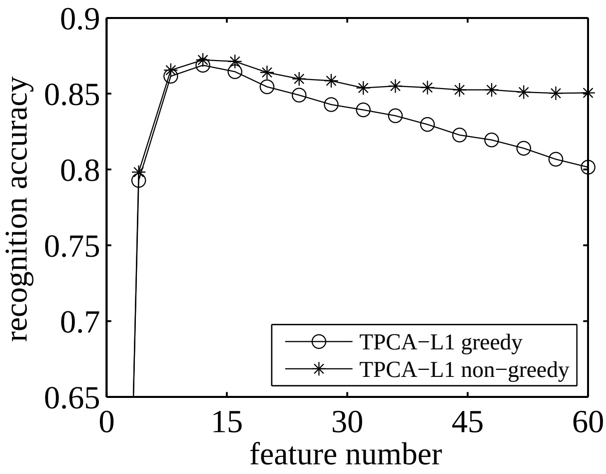
Fig. 2. Objective values versus feature number obtained by TPCA-L1 greedy and TPCA-L1 non-greedy, respectively.



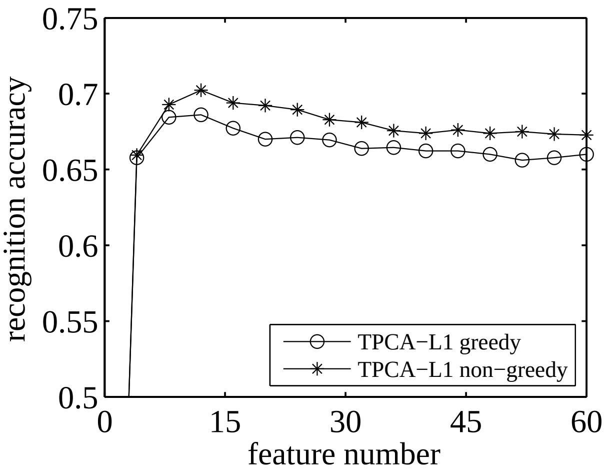
(a) ORL images without outliers



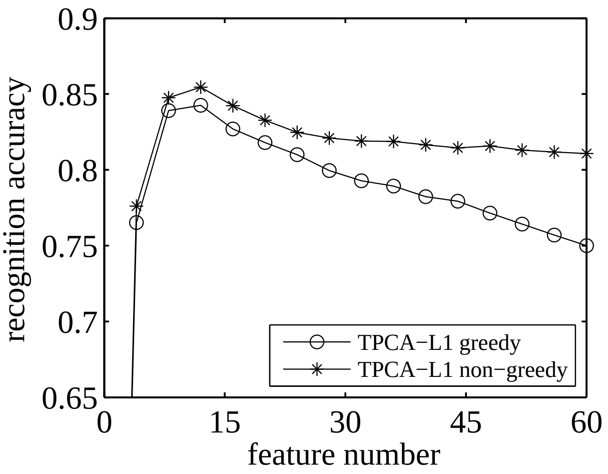
(b) Yale images without outliers



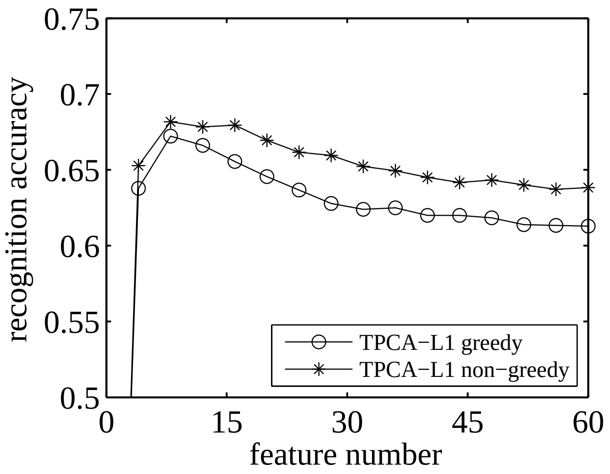
(c) 20% of ORL images with outliers



(d) 20% of Yale images with outliers



(e) 40% of ORL images with outliers



(f) 40% of Yale images with outliers

Fig. 3. Recognition accuracy versus feature number on the ORL and Yale databases obtained by TPCA-L1 greedy and TPCA-L1 non-greedy respectively.

1. Yale Face Database: The Yale Face Database[1] consists of 165 grayscale images of 15 individuals, and each individual has 11 images. Each image is reshaped into 64×64 pixels in this experiment.

2. ORL Face Database: The ORL Face Database[2] consists of 400 facial images of 40 subjects. There are ten different images of each of 40 distinct subjects. Similarly, each image is reshaped into 64×64 pixels in this experiment.

3. USPS Handwritten Digits Database: The USPS Handwritten Digits Database[3] is an image data set consisting of 9298 handwritten digits of 0 through 9. The size of each image is 16×16 pixels in this experiment.

4. FERET Face Database: The FERET Face Database[4] consists of 1400 images belonging to 200 different individuals, each of which has 7 images. The size of each image is 80×80 pixels in this experiment.

Some image samples of the four databases are shown in Fig. 1. In this experiment, we demonstrate the effectiveness of the proposed TPCA-L1 non-greedy algorithm compared to the TCPA-L1 algorithm [12]. Four aforementioned image databases are used to compare the objective values

$$\text{objective} = \frac{1}{n}\sum_{i=1}^{n}\left\|\boldsymbol{V}^T\boldsymbol{X}_i\boldsymbol{U}\right\|_1 . \tag{23}$$

Figure 2 shows the objective values versus feature number obtained by the TPCA-L1 greedy algorithm and the TPCA-L1 non-greedy algorithm, from which we can get the proposed algorithm obtains much higher objective values than that of the TPCA-L1 greedy algorithm on all the four image databases.

The classification is to evaluate the Euclidean distance between the testing image and the training image [4]. Here if define that $\boldsymbol{X}_j(\boldsymbol{j} = 1, 2, \cdots, \boldsymbol{m})$ represent the training images, and $\boldsymbol{Y}_i(\boldsymbol{i} = 1, 2, \cdots, \boldsymbol{n})$ represent the testing images, the Euclidean distance is evaluated by the equation

$$\boldsymbol{d}(\boldsymbol{Y}_i, \boldsymbol{X}_j) = \|\boldsymbol{Y}_i - \boldsymbol{X}_j\|_2. \tag{24}$$

Given one testing sample $Y \in \mathbf{L}$ class, if $\boldsymbol{d}(\boldsymbol{Y}, \boldsymbol{X}_l) = \min \boldsymbol{d}(\boldsymbol{Y}, \boldsymbol{X}_j)$, the testing image $Y \in \mathbf{L}$ class. Then we select two databases (ORL and Yale) to evaluate the proposed algorithm and the TPCA-L1 greedy algorithm for classification. The size of each image is 64×64 pixels in this experiment. We randomly select 5 images for each person as training samples, and the remaining images for testing. Here, a varying percentage of training images was corrupted by outliers. For convenience, we use "a% of training images with outliers" to

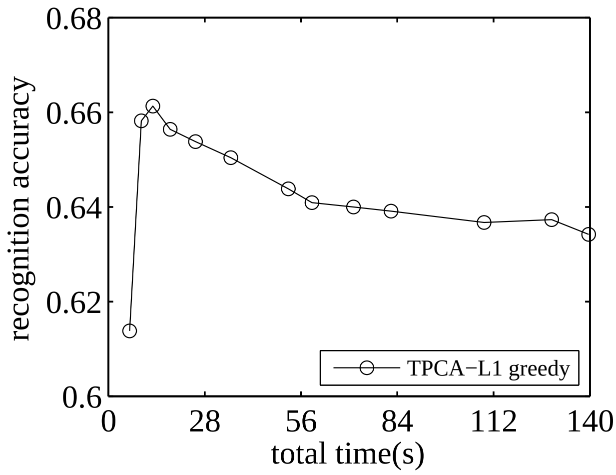


Fig. 4. Recognition accuracy versus the time cost for the Yale database.

feature numbers	2	3	4	5
TPCA-L1	0.1607	0.3849	0.5642	0.5858
TPCA-L1 NG	0.1616	0.3844	0.5791	0.6022

Tab. 1. Recognition accuracy on Yale database.

denote that a% of the images in our training set are corrupted by outliers. The percentage of training images with outliers is 0, 20 and 40 respectively. Figure 3 shows the recognition accuracy versus feature number obtained by the TPCA-L1 non-greedy algorithm and the TPCA-L1 greedy algorithm on the two face databases. As can be seen in Fig. 3, the proposed algorithm gets much better recognition accuracy than that obtained by the TPCA-L1 greedy algorithm on the ORL Face Database, especially when the feature number is large. On the Yale Face Database, when some images are corrupted by outliers, the proposed algorithm gets much higher recognition accuracy than that of the TPCA-L1 greedy algorithm.

As can be seen, the recognition accuracy decreases badly while the number of features is increasing. Note that high dimensional data is usually interrelated and has much redundancy information. In the low dimensional space, redundancy information is removed and the recognition accuracy is high. When the number of features increases, the more information (including the noises and redundancy information) are obtained which can depress the recognition accuracy.

Comparing the performances of the TPCA-L1 non-greedy and the TPCA-L1, it can be seen from Tab. 1 that the TPCA-L1 non-greedy performs better than the TPCA-L1 for small number features. The computational cost is decided by iteration times, feature numbers, image numbers and image size. However, image numbers and image size are fixed for one database. So iteration times and feature numbers are the main factors that decide the computational cost. Here we

[1]http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[2]http://www.uk.research.att.com/facedatabase.html
[3]http://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/
[4]http://www.nist.gov/itl/iad/ig/colorferet.cfm/

use the time cost to represent the computational cost. Figure 4 shows the recognition accuracy versus the time cost for TPCA-L1 non-greedy when the procedure runs 50 times. From the results, it can be seen that the recognition accuracy won't keep increasing while the time cost increasing.

5. Conclusions

In this paper, a non-greedy ℓ_1-norm based TPCA has been proposed, which is robust to outliers. By using tensor representing the training images, the proposed algorithm can exploit more spatial information of images, and thus gets better performance. In addition, all projection directions are optimized simultaneously with a non-greedy method. Experimental results on the databases show that the TPCA-L1 non-greedy algorithm performs better than the greedy method in recognition accuracy and objective values.

Acknowledgement

This work is supported by the National Nature Science Foundation of China under grants 61401471.

References

[1] JOLLIFFE, I. T. *Principal Component Analysis*. 2nd ed. New York: Springer-Verlag, 2002. ISBN: 978-0-387-95442-4. DOI: 10.1007/b98835

[2] KIRBY, M., SIROVICH, L. Application of the Karhunen-Loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1990, vol. 12, no. 1, p. 103–108. DOI: 10.1109/34.41390

[3] WOLD. S., ESBENSEN, K., GELADI, P. Principal component analysis. *Chemometrics and Intelligent Laboratory Systems*, 1987, vol. 2, no. 1, p. 37–52. DOI: 10.1016/0169-7439(87)80084-9

[4] YANG, J., ZHANG, D., FRANGI, A. F., et al. Two-dimensional PCA: a new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2004, vol. 26, no. 1, p. 131–137. DOI: 10.119/TPAMI.2004.1261097

[5] KOLDA, T. G., BADER, B. W. Tensor decompositions and applications. *SIAM review*, 2009, vol. 51, no. 3, p. 455–500. DOI: 10.1137/07070111X

[6] HE, X., CAI, D., NIYOGI, P. Tensor subspace analysis. In *Advances in Neural Information Processing Systems 18 (NIPS)*, Cambridge, 2006, p. 499–506. ISBN: 9780262232531

[7] VASILESCU, M., TERZOPOULOS, D. Multilinear subspace analysis of image ensembles. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Madison, 2003, vol. 2, p. 93-99. ISSN: 1063-6919. DOI: 10.1109/CVPR.2003.1211457

[8] YE, J. Generalized low rank approximations of matrices. *Machine Learning*, 2005, vol. 61, no. 1-3, p. 167–191. ISSN: 0885-6125. DOI: 10.1007/s10994-005-3561-6

[9] KE, Q., KANADE, T. Robust ℓ_1-norm factorization in the presence of outliers and missing data by alternative convex programming. In *Proceedings IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. San Diego, 2005, p. 739–746. ISSN: 1063-6919. DOI: 10.1109/CVPR.2005.309

[10] DING, C., ZHOU, D., HE, X., et al. R1-PCA: Rotational invariant ℓ_1-norm principal component analysis for robust subspace factorization. In *Proceedings of the 23rd International Conference on Machine Learning (ICML)*. Pittsburgh, 2006, p. 281–288. ISBN: 1-59593-383-2. DOI: 10.1145/1143844.1143880

[11] KWAK, N. Principal component analysis based on ℓ_1-norm maximization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2008, vol. 30, no. 9, p. 1672–1680. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2008.114

[12] PANG, Y., LI, X., YUAN, Y. Robust tensor analysis with ℓ_1-norm. *IEEE Transactions on Circuits and Systems for Video Technology*, 2010, vol. 20, no. 2, p. 172–178. ISSN: 1051-8215. DOI: 10.1109/TCSVT.2009.2020337

[13] NIE, F., HUANG, H., DING, C., et al. Robust principal component analysis with non-greedy ℓ_1-norm maximization. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, 2011, p. 1433–1438.

[14] WANG, R., NIE, F., YANG, X., et al. Robust 2DPCA with non-greedy ℓ_1-norm maximization for image analysis. *IEEE Transactions on Cybernetics*, 2015, vol. 45, no. 5, p. 1108–1112. ISSN: 2168-2267. DOI: 101109/TCVB.2014.2341575

[15] CAI, D., HE, X., HAN, J., et al. Orthogonal laplacianfaces for face recognition. *IEEE Transactions on Image Processing*, 2006, vol. 15, no. 11, p. 3608–3614. ISSN: 1057-7149. DOI: 10.1109/TIP.2006.881945

About the Authors . . .

Limei ZHAO was born in 1989. She received the B.E. degree from Xi'an Research Institute of Hi-Tech in 2013. She is currently pursuing the M.E. degree in the Xi'an Research Institute of Hi-Tech. Her current research interests include signal processing and machine learning.

Weimin JIA was born in 1971. She has been a professor of the Xi'an Research Institute of Hi-Tech since 2014. Her current interests include array signal processing and broadband mobile satellite communication.

Rong WANG was born in 1983. He received the B.E. degree in information engineering, the M.E. degree in signal and information processing, the Ph.D. degree in computer science from Xi'an Research Institute of Hi-Tech, Xi'an, China, in 2004, 2007 and 2013, respectively. During 2007 and 2013, he also studied in the Department of Automation, Tsinghua University, Beijing, China for his Ph.D. degree. His current research interests include signal processing and machine learning, together with their applications such as pattern recognition, image processing and computer vision.

Qiang YU was born in 1987. He received the M.E. degree from Xi'an Research Institute of Hi-Tech in 2012. He is currently pursuing the Ph.D. degree in the Xi'an Research Institute of Hi-Tech. His current research interests include signal processing and machine learning.

Appendix:

Algorithm 1: An efficient algorithm to solve a general ℓ_1-norm maximization problem.

Initialize $v^t \in C, t = 1$;
while not converge
1. For each i, compute ${p_i}^t = \text{sgn}(g_i(v^t))$;
2. $v^{t+1} = \arg\max_{v \in C} f(v) + \sum_{i=1}^{n} p_i^t g_i(v)$;
3. $t = t + 1$;

end while
Output: v^{t+1}.

Algorithm 2: TPCA with non-greedy ℓ_1-norm maximization.

Input: $\boldsymbol{X} \in \mathbf{R}^{h \times w \times n}$ and feature number $\boldsymbol{r}$.
Initialize $\boldsymbol{U}$;
while not converge **do**

1. Fixed $\boldsymbol{U}$, compute $\boldsymbol{V}$

while not converge **do**

1). $\boldsymbol{p}^{(i)}(j) = \text{sgn}(\boldsymbol{V}^T \boldsymbol{y1}_j^{(i)})$, $\boldsymbol{y1}_j^{(i)}$ is the $\boldsymbol{j}^{th}$ column vector of matrix $\boldsymbol{Y1}^{(i)} = \boldsymbol{X}_i \boldsymbol{U}$;

2). $\boldsymbol{Y} = [\boldsymbol{y1}_1^{(1)}, \boldsymbol{y1}_2^{(1)}, \cdots, \boldsymbol{y1}_r^{(1)}, \boldsymbol{y1}_1^{(2)}, \cdots, \boldsymbol{y1}_r^{(n)}]$, $\boldsymbol{P} = \text{sgn}(\boldsymbol{V}^T \boldsymbol{Y})$ and $\boldsymbol{M} = \boldsymbol{Y}\boldsymbol{P}^T$;

3). Suppose the SVD of M as $\boldsymbol{M} = \boldsymbol{L}\Lambda\boldsymbol{R}^T$, $\boldsymbol{V}^{t_1+1} = \boldsymbol{R}\boldsymbol{Z}^T\boldsymbol{L}^T = \boldsymbol{R}[\boldsymbol{I}_r, \mathbf{0}]\boldsymbol{L}^T$;

4). $t_1 = t_1 + 1$;

end while

2. Fixed $\boldsymbol{V}$, compute $\boldsymbol{U}$

while not converge **do**

1). $\boldsymbol{s}^{(i)}(j) = \text{sgn}(\boldsymbol{U}^T \boldsymbol{y2}_j^{(i)})$, $\boldsymbol{y2}_j^{(i)}$ is the $\boldsymbol{j}^{th}$ column vector of matrix $\boldsymbol{Y2}^{(i)} = \boldsymbol{X}_i^T \boldsymbol{V}$;

2). $\boldsymbol{Y} = [\boldsymbol{y2}_1^{(1)}, \boldsymbol{y2}_2^{(1)}, \cdots, \boldsymbol{y2}_r^{(1)}, \boldsymbol{y2}_1^{(2)}, \cdots, \boldsymbol{y2}_r^{(n)}]$, $\boldsymbol{S} = \text{sgn}(\boldsymbol{U}^T \boldsymbol{Y})$ and $\boldsymbol{M} = \boldsymbol{Y}\boldsymbol{S}^T$;

3). Suppose the SVD of M as $\boldsymbol{M} = \boldsymbol{L}\Lambda\boldsymbol{R}^T$, $\boldsymbol{U}^{t_2+1} = \boldsymbol{R}\boldsymbol{Z}^T\boldsymbol{L}^T = \boldsymbol{R}[\boldsymbol{I}_r, \mathbf{0}]\boldsymbol{L}^T$;

4). $t_2 = t_2 + 1$;

end while

end while
Output: $\boldsymbol{U}$ and $\boldsymbol{V}$.