Robust Tensor Clustering with Non-Greedy Maximization

Xiaochun Cao[1,3], Xingxing Wei[1], Yahong Han[1], Yi Yang[2], Dongdai Lin[3]
[1]Tianjin Key Laboratory of Cognitive Computing and Application,
School of Computer Science and Technology, Tianjin University.
[2]School of Information Technology & Electrical Engineering, The University of Queensland.
[3]State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences.
{caoxiaochun, ddlin}@iie.ac.cn, {xwei, yahong}@tju.edu.cn, yee.i.yang@gmail.com

Abstract

Tensors are increasingly common in several areas such as data mining, computer graphics, and computer vision. Tensor clustering is a fundamental tool for data analysis and pattern discovery. However, there usually exist outlying data points in real-world datasets, which will reduce the performance of clustering. This motivates us to develop a tensor clustering algorithm that is robust to the outliers. In this paper, we propose an algorithm of Robust Tensor Clustering (RTC). The RTC firstly finds a lower rank approximation of the original tensor data using a L1 norm optimization function. Because the L1 norm doesn't exaggerate the effect of outliers compared with L2 norm, the minimization of the L1 norm approximation function makes RTC robust to outliers. Then we compute the HOSVD decomposition of this approximate tensor to obtain the final clustering results. Different from the traditional algorithm solving the approximation function with a greedy strategy, we utilize a non-greedy strategy to obtain a better solution. Experiments demonstrate that RTC has better performance than the state-of-the-art algorithms and is more robust to outliers.

1 Introduction

Tensors can be taken as a natural representation of visual data. For example, an image can be represented by a second-order tensor [Ma *et al.*, 2013] and a time-series video clip can be represented by a third-order tensor. Compared with the relaxation of tensor by vectorization an image into a long vector, tensor-based representation can preserve more spatial information of images. Therefore, tensor-based representation is more informative than the vectorization method. Additionally, the dimensionality of tensor-based representation is lower, which can effectively avoid high computational complexity, and large memory requirements generated by the vectorization of data [Guo *et al.*, 2012].

Tensor clustering seeks to partition several m-order input tensors into groups by the minimization of certain criterion. It is a basic data analysis task and receives much attention recently in data mining [Zha *et al.*, 2008], computer vision [Guo *et al.*, 2012], and so on. Like clustering on the vectors, tensor clustering is NP-hard to optimize, which needs additional optimization algorithms to approximatively solve it [Huang *et al.*, 2008; Jegelka *et al.*, 2009]. However, it is known that there exist outlying data points in the real-world datasets. When we use the above clustering algorithms to mine the structural information existing within these data, the outliers will reduce the performance of clustering algorithms, resulting in some misclustered sample data.

This urges us to develop a tensor clustering algorithm that is robust to the outliers. We call it the Robust Tensor Clustering (RTC). Our method firstly finds a lower rank approximation of the original tensor data using a L1 norm optimization function. Because the L1 norm doesn't exaggerate the effect of outliers compared with L2 norm, the minimization of the L1 norm approximation function makes RTC robust to outliers. Then we compute the HOSVD decomposition of this approximate tensor to obtain the final clustering results. We formulate the process of approximation into a framework of Tensor PCA with L1-norm (TPCA-L1) [Pang *et al.*, 2010]. TPCA-L1 is a variant of Tensor PCA with Frobenius norm (TPCA-F) [De Lathauwer *et al.*, 2000], where the Frobenius norm is replaced with the L1-norm. In this way, TPCA-L1 will suppress the negative influence of outliers [Kwak, 2008]. However, [Pang *et al.*, 2010] uses a greedy strategy like [Kwak, 2008] to maximize the function of TPCA-L1. Specifically, the projection directions are sequentially optimized one by one. This greedy method is easy to get stuck in a local solution. To overcome this drawback, we utilize a non-greedy maximization to directly solve the projection directions inspired by the work in [Nie *et al.*, 2011], and thus much better solution can be obtained.

To sum up, the main contributions of this paper are as follows: 1) We propose a Robust Tensor Clustering (RTC) algorithm; 2) We utilize a non-greedy maximization method and then integrate it into the framework of RTC. Finally, we evaluate the performance of RTC on five public datasets and compare it with five state-of-the-art methods. Experimental results demonstrate the effectiveness of our method.

The rest of this paper is organized as follows. We present the framework of Robust Tensor Clustering with Non-Greedy Maximization in Section 2. The solution and algorithm are given in Section 3. Section 4 reports all experimental results. Finally, we summarize the conclusions in Section 5.

2 The Proposed Framework

Consider a sequence of input *M*-order (mode) tensor data $\mathcal{X}_j \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M}, j = 1, \cdots, n$, the goal of clustering is to find a *K*-way disjoint partitioning $(\mathbb{S}_1, ..., \mathbb{S}_K)$ such that the following objective is minimized:

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

where $\mathcal{C}_k \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_M}$ denotes the *k*-th cluster centroid. *K* is the number of clusters. $\|\cdot\|$ denotes the Euclidean distance between $\mathcal{X}_j$ and $\mathcal{C}_k$. We introduce an indicator vector q_k for the cluster $\mathbb{S}_k$.

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

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

The RTC aims to find a more accurate indicator matrix $Q' \in \mathbb{R}^{n \times K}$ via making the clustering algorithm robust to outliers. To this end, RTC maximizes the following problem to get the projection matrix, and then uses the projection matrix to solve approximation for the original tensor data:

$$\max_{U_i|_{i=1}^M} \sum_{j=1}^{n} \|\mathcal{X}_j \prod_{i=1}^{M} \times_i U_i^T\|_1, \tag{3}$$

where $U_i \in \mathbb{R}^{n_i \times r_i} (i = 1, \cdots, M)$ is called projection matrix (which is orthogonal, *i.e.*, $U_i^T U_i = I$). Generally, $r_i < n_i$, so we can obtain a low-rank approximation $\mathcal{Y}_j$ for $\mathcal{X}_j$. $\mathcal{Y}_j \approx \mathcal{X}_j \prod_{i=1}^{M} \times_i U_i^T$, where $\mathcal{X}_j \prod_{i=1}^{M} \times_i U_i^T = \mathcal{X}_j \times_1 U_1^T \cdots \times_M U_M^T$, and $\mathcal{X}_j \times_i U_i^T$ denotes the *i*-mode product of the tensor $\mathcal{X}_j$ with the matrix $U_i \in \mathbb{R}^{n_i \times r_i}$. Please refer to [Kolda and Bader, 2009] 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.

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

The problem in (3) is called TPCA-L1 proposed in [Pang *et al.*, 2010]. TPCA-L1 is a variant of TPCA-F [De Lathauwer *et al.*, 2000], replacing the Frobenius norm with the L1-norm, and thus can suppress the negative influence of outliers [Kwak, 2008].

Once $U_i \in \mathbb{R}^{n_i \times r_i}, i = 1, \cdots, M$ are computed, the low-rank approximation $\mathcal{Y}_j$ for $\mathcal{X}_j$ can be get. Thus we obtain a *T*-order tensor $\mathcal{Z} \in \mathbb{R}^{r_1 \times \cdots \times r_M \times n}$ comprising 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 can decompose the tensor $\mathcal{Z}$ to obtain the final clustering results for $\mathcal{X}_j, (j = 1, \cdots, n)$ by solving the following optimization function:

$$\min_{\mathcal{S}, P_i|_{i=1}^T} \|\mathcal{Z} - \mathcal{S} \prod_{i=1}^{T} \times_i P_i\|_F^2, s.t. P_i^T P_i = I, \forall i \in (1, \cdots, T), \tag{5}$$

where $\mathcal{S} \in \mathbb{R}^{d_1 \times d_2 \times \cdots \times K}$ is called the core tensor. $P_i \in \mathbb{R}^{r_i \times d_i} (i = 1, \cdots, T\text{-}1)$ and $P_T \in \mathbb{R}^{n \times K}$ are called factor matrices. The solution to the optimization problem of (5) can be found by HOSVD [De Lathauwer *et al.*, 2000]. [Huang *et al.*, 2008] demonstrates that $P_T \in \mathbb{R}^{n \times K}$ is the clustering results for *n* two-order tensors $\mathcal{X}_j$. *i.e.*, $P_T \in \mathbb{R}^{n \times K}$ is the indicator matrix of clustering *n* tensor $\mathcal{X}_j$ into *K* cluster centroids. In this way, we obtain $Q' = P_T \in \mathbb{R}^{n \times K}$.

3 Solution and Algorithm

In this section, we firstly discuss the non-greedy maximization algorithm for the vector case, and then extend this non-greedy strategy to handle tensor data.

Given a sequence of vectors (one-order tensor) $x_j \in \mathbb{R}^d, j = 1, \cdots, n$, then problem (3) is formulated as follows:

$$\max_U \sum_{j=1}^{n} \|U^T x_j\|_1, s.t. U^T U = I, \tag{6}$$

where $U \in \mathbb{R}^{d \times m}$ is the projection matrix, and $d > m$. Without loss of generality, $x_j, j = 1, \cdots, n$ are assumed to be centralized, *i.e.*, $\sum_{j=1}^{n} x_j = 0$.

The problem (6) should be replaced with Eq. (7):

$$\max_U \sum_{j=1}^{n} \alpha_j^T U^T x_j, s.t. U^T U = I, \tag{7}$$

where $\alpha_j = sgn(U^T x_j)$. $sgn(\cdot)$ denotes the sign function defined as follows: *sgn*(*x*) = 1 if $x > 0$, *sgn*(*x*) = -1 if $x < 0$, and *sgn*(*x*) = 0 if $x = 0$. Denote $M = \sum_{j=1}^{n} x_j \alpha_j^T$, [Nie *et al.*, 2011] proves that U can be obtained by computing the SVD of *M* as $M = W \Lambda V^T$, and then $U = WV^T$. This non-greedy strategy directly solve the projection direction U, and has been demonstrated to get much better solution than the non-greedy method [Kwak, 2008].

In the next paragraphs, we will discuss how to extend this non-greedy strategy to handle tensor data. Because the optimization algorithm is easy to be generalized to much higher order tensors from two-order tensors [Pang *et al.*, 2010], we constrain to a sequence of second-order tensors from now on.

Given a sequence of second-order tensors $\mathcal{X}_j \in \mathbb{R}^{n_1 \times n_2}, j = 1, \cdots, n$ (*i.e.*, a sequence of matrices, we use X_j to replace $\mathcal{X}_j$ in the following paragraphs). Problem (3) is formulated as follows:

$$\max_{U_1, U_2} \sum_{j=1}^{n} \|U_1^T X_j U_2\|_1, s.t. U_1^T U_1 = I, U_2^T U_2 = I, \tag{8}$$

where $U_1 \in \mathbb{R}^{n_1 \times r_1}, U_2 \in \mathbb{R}^{n_2 \times r_2}$ are projection matrices and $n_1 > r_1, n_2 > r_2$. We use an alternative optimization technique to solve problem (8). Specially, we shall compute U_1 while fixing U_2 and compute U_2 while fixing U_1. The procedure is iteratively performed until the convergence criteria is meeted with.

To begin, let U_2 be initialized and then fixed. Now the problem is to find the optimal U_1 such that (8) is maximized. When U_2 is fixed, problem (8) can be formulated as follows:

$$\max_{U_1} \sum_{j=1}^{n} \sum_{i=1}^{r_2} \|U_1^T X_j (U_2)_i\|_1, s.t. U_1^T U_1 = I, \tag{9}$$

where $(U_2)_i \in \mathbb{R}^{n_2 \times 1}$ denotes the *i*-th column in U_2. Problem (9) can be formulated as follows:

$$\max_{U_1} \sum_{k=1}^{p} \|U_1^T z_k\|_1, s.t. U_1^T U_1 = I, \quad (10)$$

where $p = n \times r_2$, $z_k = X_j(U_2)_i$ and $z_k \in \mathbb{R}^{n_1 \times 1}$. Problem (10) is the same to (6), and thus can be easily solved.

After the optimal U_1 is obtained, we view U_1 as the known term, and the task becomes the problem of computing the optimal U_2 that maximizes (8):

$$\max_{U_2} \sum_{j=1}^{n} \sum_{i=1}^{r_1} \|(U_1)_i^T X_j U_2\|_1, s.t. U_2^T U_2 = I, \quad (11)$$

where $(U_1)_i \in \mathbb{R}^{n_2 \times 1}$ denotes the *i*-th column in U_1. Problem (11) can be formulated as follows:

$$\max_{U_2} \sum_{k=1}^{p} \|z_k U_2\|_1, s.t. U_2^T U_2 = I, \quad (12)$$

where $p = n \times r_1$, $z_k = (U_1)_i^T X_j$, and $z_k \in \mathbb{R}^{1 \times n_2}$. Problem (12) can be formulated as follows:

$$\max_{U_2} \sum_{k=1}^{p} \|U_2^T z_k^T\|_1, s.t. U_2^T U_2 = I, \quad (13)$$

which is the same to problem(6).

We use the relative reduction of the Root Mean Square Reconstruction Error with L1-norm ($RMSRE_{L1}$) to check the convergence. The $RMSRE_{L1}$ is defined as follows:

$$RMSRE_{L1} = \sqrt{\frac{1}{n} \sum_{j=1}^{n} \|X_j - U_1 Y_j U_2^T\|_1}, \quad (14)$$

where $Y_j = U_1^T X_j U_2$, and $Y_j \in \mathbb{R}^{r_1 \times r_2}$. $RMSRE_{L1}$ is a modified version of *RMSRE* [Ye, 2005], replacing the Frobenius norm with the L1-norm. The convergence is determined by checking whether the following inequality holds:

$$\frac{|RMSRE_{L1}(t-1) - RMSRE_{L1}(t)|}{RMSRE_{L1}(t-1)} < \eta, \quad (15)$$

for some small threshold $\eta > 0$. $RMSRE_{L1}(t)$ denotes the $RMSRE_{L1}$ value in the *t*-th iteration step. In our experiments, we choose $\eta = 10^{-3}$. Results in Section 4 show that the algorithm converges within two to three iterations.

After computing $Y_j \in \mathbb{R}^{r_1 \times r_2}, j = 1, \cdots, n$, we obtain a three-order tensor $\mathcal{Y} = [Y_1, \cdots, Y_n] \in \mathbb{R}^{r_1 \times r_2 \times n}$. We use formula (5) to decompose $\mathcal{Y} \in \mathbb{R}^{r_1 \times r_2 \times n}$, where $T = 3$. Note P_T is a continuous solution, in order to obtain the discrete indicator matrix Q', we do K-means across all the rows in $P_T \in \mathbb{R}^{n \times K}$. In this way, we get the final discrete indicator matrix $Q' \in \mathbb{R}^{n \times K}$. The whole algorithm for Robust Tensor Clustering is summarized in Algorithm 1.

Algorithm 1 Robust Tensor Clustering (RTC) algorithm.

Input: $\mathcal{X} = [X_1, \cdots, X_n] \in \mathbb{R}^{n_1 \times n_2 \times n}$, where $\mathcal{X}$ is centralized, r_1, r_2, cluster number *K*;
Initialize $U_2^1 \in \mathbb{R}^{n_2 \times r_2}$ such that $(U_2^1)^T U_2^1 = I$, *t* = 1;
while *not converge* **do**
  Compute U_1^t using (10) while fixing U_2^t;
  Compute U_2^{t+1} using (13) while fixing U_1^t;
  t = *t* + 1;
end while
Computing $Y_j = U_1^T X_j U_2, j = 1, \cdots, n$ to obtain $\mathcal{Y} = [Y_1, \cdots, Y_n] \in \mathbb{R}^{r_1 \times r_2 \times n}$;
Computing HOSVD of $\mathcal{Y}$ using formula (5), where *T*=3;
Doing K-means across all the rows in P_T.
Output:Indicator matrix $Q' \in \mathbb{R}^{n \times K}$

4 Experiments

4.1 Datasets

We have conducted experiments on five real-world large datasets, including three face datasets (Yale, ORL, PIE), a handwritten digit dataset (USPS) and an object dataset (COIL20). The brief description of the datasets is listed as follows:

Yale The Yale face database[1] contains 165 gray scale images of 15 individuals, each individual has 11 images. Each image is reshaped into 64 × 64 pixels in our experiments.

ORL The ORL face database[2] consists of a total of 400 face images, of a total of 40 people (10 samples per person). Similarly, each image is reshaped into 64 × 64 pixels.

PIE The PIE[3] is a database of 41,368 images of 68 people, each person under 13 different poses. We randomly choose one near frontal poses (C05 in our experiment), and use all the images under different illuminations and expressions, thus we get 49 images for each individual and totally 3332 images. Each image is reshaped into 64 × 64 pixels.

USPS USPS[4] is an image dataset consisting of 9298 handwritten digits of "0" through "9". The size of each image is 16 ×16 pixels in our experiment.

COIL20 COIL20[5] contains 20 objects. The images of each objects were taken 5 degrees apart as the object is rotated on a turntable and each object has 72 images. The size of each image is 32x 32 pixels, with 256 grey levels per pixel.

All the features (pixel values) are scaled into [0,1] in our experiments and no other preprocessing step is applied.

4.2 Evaluation Metric

We use two metrics to measure the clustering performance: Accuracy (AC) and Normalized Mutual Information (NMI) [Cai *et al.*, 2005]. 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:

$$AC = \frac{\sum_{i=1}^{n} \delta(s_i, map(r_i))}{n}, \quad (16)$$

where *n* is the total number of images, $\delta(x, y)$ is the delta function that equals one if *x* = *y* and equals zero otherwise. $map(r_i)$ is the permutation mapping function that maps each

[1] http://cvc.yale.edu/projects/yalefaces/yalefaces.html
[2] http://www.uk.research.att.com/facedatabase.html
[3] http://vasc.ri.cmu.edu/idb/html/face/
[4] http://www.csie.ntu.edu.tw/∼ cjlin/libsvmtools/datasets/
[5] http://www.cs.columbia.edu/CAVE/software/softlib/

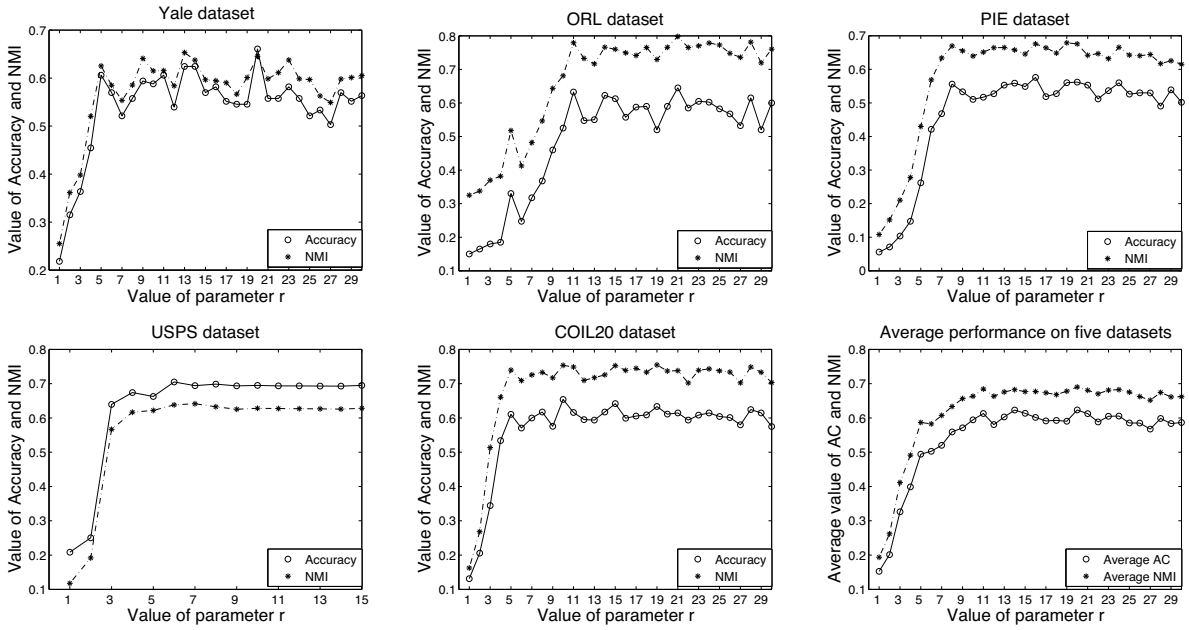

Figure 1: The performance of RTC in terms of AC and NMI, with respect to different values of *r*. From left to right are the results on Yale, ORL, PIE, USPS and COIL20. We also list the average performance of RTC on five datasets, where Average AC is defined as the average value of Accuracy on five datasets in terms of the same value of parameter *r* (when $r > 15$, we let the value of AC equal to that when $r = 15$ on USPS dataset). The definition of Average NMI is similar to Average AC.

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 Normalized Mutual Information (NMI) is defined:

$$\overline{MI}(C, C') = \frac{MI(C, C')}{max(H(C), H'(C'))}, \tag{17}$$

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.

4.3 Results

In order to evaluate the performance of proposed Robust Tensor Clustering algorithm, we compare it with five state-of-the-art approaches: two traditional clustering methods: K-means [Bishop, 2006] and PCA+K-means [Bishop, 2006]; two spectral clustering methods: Self-tune SC (SSC) [Chen *et al.*, 2011] and LSC-K [Chen and Cai, 2011]. We also compare the method in [Huang *et al.*, 2008], and call it HOSVD clustering in our experiments.

All the methods need to input the cluster number K, we let K equal to the ground-truth number of classes in different datasets. In addition, SSC needs to input $\hat{K}$, which is the same to that in KNN. We tune $\hat{K}$ in the rang of 5, 10, 15, 20 in our experiments. LSC-K needs to input the number of landmarks p, which is related to the size of datasets. In our experiments, we let p equal to 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 [Chen and Cai, 2011]. In RTC, there are 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. For K-means, PCA+K-means, SSC, and LSC-K, we reshape each image into a vector according to the size of images in different datasets, and then input the vectors to the clustering algorithms. For HOSVD and RTC, we directly use a matrix to represent an image, and then cluster these matrices. The dimensionality reduction parameter for HOSVD is the same as that in RTC.

Table 1 lists the comparison results on five datasets. We conduct 30 tests on each dataset and the best performance is reported. From the results in Table 1, we can draw the following three conclusions. First, PCA+K-means achieves better performance than K-means, which demonstrates the effectiveness of PCA reduction dimension. Second, RTC outperforms HOSVD on all five datasets. The improvement varies from 1.23% (USPS) to 12.73% (Yale), and averages at 5.744% for AC (the average is 4.6% for NMI). This demonstrates the effectiveness of our proposed robust tensor clustering algorithm, which takes HOSVD as its second step. Third, HOSVD achieves better performance than K-means on average. This demonstrates that the matrix-based method can exploit more spatial information than vector-based method, and thus obtain the better clustering performance. In addition, we note that spectral clustering methods (SSC and LSC-K) achieve better performance than RTC on USPS and COIL20 (even the PCA+K-means also shows slightly improvement than RTC on COIL20 dataset). This is because the size of

Table 1: Comparison of different methods for clustering Accuracy and NMI on five real-world datasets.

Metric	Dataset	K-means	PCA+K-means	Self-tune SC	SLC-K	HOSVD	RTC
AC	**Yale**	0.5030	0.5879	0.6061	0.5879	0.5333	**0.6606**
	ORL	0.5775	0.5900	**0.6475**	0.6400	0.5675	0.6450
	PIE	0.1564	0.1600	0.3785	0.3869	0.5315	**0.5759**
	USPS	0.6734	0.6755	0.6762	**0.8132**	0.6926	0.7049
	COIL20	0.6347	0.7111	**0.8090**	0.8028	0.6285	0.6542
NMI	**Yale**	0.5391	0.6056	0.6137	0.6314	0.5829	**0.6531**
	ORL	0.7425	0.7515	0.7965	**0.7990**	0.7558	0.7982
	PIE	0.3592	0.3676	0.5667	0.6068	0.6156	**0.6790**
	USPS	0.6141	0.6145	0.8071	**0.8624**	0.6254	0.6414
	COIL20	0.7359	0.7859	**0.9111**	0.8841	0.7166	0.7546

images in both the two datasets is smaller compared with that on Yale and PIE. Moreover, the content of images in these two datasets is relatively simple. In this case, the spatial information within images isn't sufficient, and can't play a important role in clustering, leading to the worse performance of matrix-based method (RTC and HOSVD) than vector-based methods (PCA+K-means, SSC and LSC-K).

Now we discuss the influence of parameter r to the RTC algorithm. Figure 1 illustrates the parameter tuning results of RTC on five datasets. As expected, 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 can't 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 do clustering. In addition, we can see that the curves of parameter tuning on Yale and ORL are less smooth compared with that on USPS, PIE and COIL20. The reason for this may be that USPS, PIE and COIL20 have more images than that on Yale and ORL.

In order to evaluate the robustness of RTC to outliers, we add artificial occlusions on some randomly selected images of USPS dataset to generate simulative polluted images (outliers) like that in [Kwak, 2008] and [Pang *et al.*, 2010]. Then we test the performance of different methods on this polluted dataset. 10 sample images with artificial occlusions is listed in Figure 3. 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 in dataset. Figure 2 illustrates the AC and NMI comparison results for SSC, LSC-K and RTC in terms of different polluted ratios. To better show the robustness of RTC, we individually list the comparisons between RTC and HOSVD (see the right figure in Figure 2). From Figure 2, we can see that RTC gradually outperforms the SSC and LSC-K with the increasing of polluted ratio (0.5 for AC and 0.6 for NMI), which demonstrates that RTC can obtain better clustering performance when the dataset is met with a high polluted ratio. Further, we add the artificial occlusions on other four datasets like that on USPS (the polluted ratio is 0.1 for Yale, 0.5 for ORL, 0.1 for PIE, 0.6 for USPS, and 1 for COIL20, respectively). The results

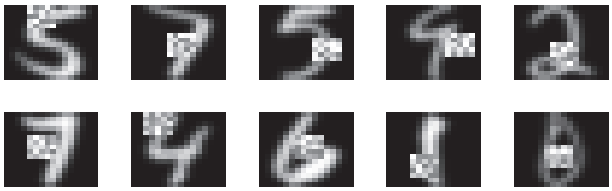

Figure 3: 10 sample images with artificial occlusions on USPS handwritten dataset. We regard these polluted images as outliers like that in [Kwak, 2008] and [Pang *et al.*, 2010].

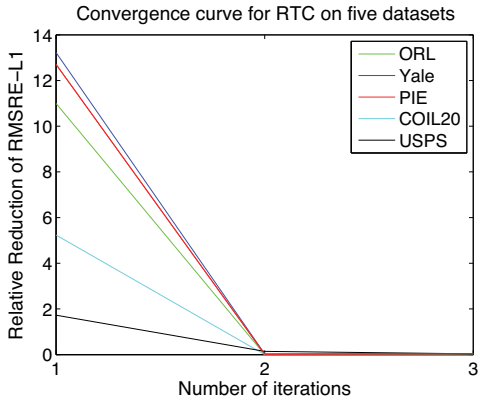

Figure 4: Convergence curve for RTC on five datasets.

on these datasets in listed in Table 2. We can see that RTC obtains the best clustering performance on all datasets, which effectively demonstrates RTC is more robust to the outliers than other clustering algorithms.

We also evaluate the convergence of Algorithm 1, the experimental results are drawn in Figure 4. We can see that RTC converges within two to three iterations on all five datasets.

5 Conclusions

In this paper, we proposed an algorithm of Robust Tensor Clustering (RTC). Experimental results show that RTC usually gets better performance when the dataset meets a polluted ratio, which effectively demonstrates that RTC is more robust to the outliers than other clustering algorithms. Moreover,

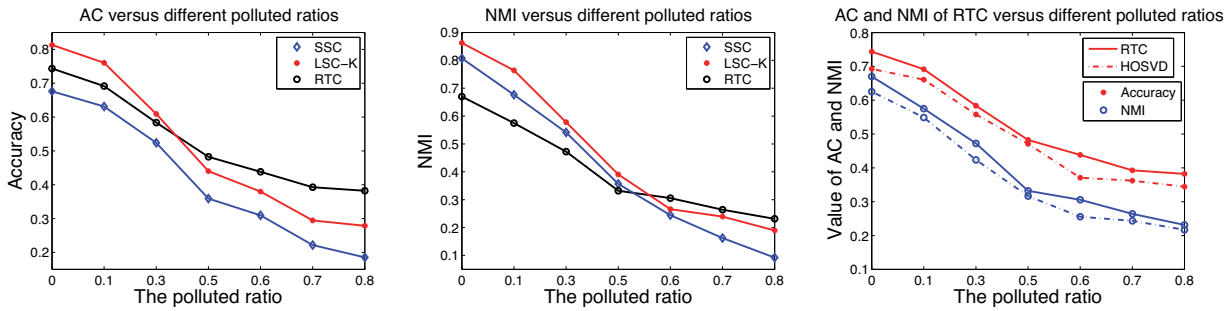

Figure 2: Comparison results of RTC in terms of SSC and LSC-K versus different polluted ratio on USPS dataset. To better show the robustness of RTC, we individually list the comparisons between RTC and HOSVD in the right figure.

Table 2: Comparison of different methods for clustering Accuracy and NMI on five real-world datasets with outliers.

Metric	Dataset	K-means	PCA+K-means	Self-tune SC	SLC-K	HOSVD	RTC
AC	**Yale**	0.4909	0.5091	0.5636	0.5636	0.5091	**0.6000**
	ORL	0.2400	0.3725	0.3400	0.4175	0.3475	**0.4525**
	PIE	0.1462	0.1348	0.3568	0.3640	0.4937	**0.5267**
	USPS	0.3202	0.3228	0.3097	0.3799	0.3708	**0.4384**
	COIL20	0.3278	0.3389	0.3493	0.3493	0.3542	**0.3576**
NMI	**Yale**	0.5445	0.5449	0.5735	0.5945	0.5762	**0.6227**
	ORL	0.3911	0.5729	0.5488	0.6001	0.5234	**0.6221**
	PIE	0.3423	0.3448	0.5412	0.5971	0.6127	**0.6467**
	USPS	0.2423	0.2440	0.2444	0.2661	0.2556	**0.3057**
	COIL20	0.4268	0.4334	0.4344	0.4125	0.4260	**0.4412**

as a matrix-based method, RTC can exploit more spatial information of images, and thus has better performance when the images reserve sufficient spatial information. In addition, we utilize a non-greedy maximization algorithm to solve the RTC problem. Experiments show that our algorithm can effectively converges within two to three iterations.

Acknowledgments

We would like to thank prof. Zhi-Hua Zhou for his insightful comments on this work. This work is supported by National High-tech R&D Program of China (2013AA01A601), 100 Talents Programme of The Chinese Academy of Sciences, the NSFC (under Grant 61202166), and Doctoral Fund of Ministry of Education of China (under Grant 20120032120042).

References

[Bishop, 2006] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[Cai *et al.*, 2005] D. Cai, X. He, and J. Han. Document clustering using locality preserving indexing. *IEEE Transactions on KDE*, 17(12):1624–1637, 2005.

[Chen and Cai, 2011] X. Chen and D. Cai. Large scale spectral clustering with landmark-based representation. In *AAAI*, 2011.

[Chen *et al.*, 2011] W.Y. Chen, Y. Song, H. Bai, C.J. Lin, and E.Y. Chang. Parallel spectral clustering in distributed systems. *IEEE Transactions on PAMI*, 33(3):568–586, 2011.

[De Lathauwer *et al.*, 2000] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on MAA*, 21(4):1253–1278, 2000.

[Guo *et al.*, 2012] W. Guo, I. Kotsia, and I. Patras. Tensor learning for regression. *IEEE Transactions on Image Processing*, 21(2):816–827, 2012.

[Huang *et al.*, 2008] 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 *KDD*, pages 327–335. ACM, 2008.

[Jegelka *et al.*, 2009] S. Jegelka, S. Sra, and A. Banerjee. Approximation algorithms for tensor clustering. In *Algorithmic Learning Theory*, pages 368–383. Springer, 2009.

[Kolda and Bader, 2009] T.G. Kolda and B.W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[Kwak, 2008] N. Kwak. Principal component analysis based on l1-norm maximization. *IEEE Transactions on PAMI*, 30(9):1672–1680, 2008.

[Ma *et al.*, 2013] Z. Ma, Y. Yang, F. Nie, and N. Sebe. Thinking of images as what they are: Compound matrix regression for image classification. In *IJCAI*. AAAI Press, 2013.

[Nie *et al.*, 2011] F. Nie, H. Huang, C. Ding, D. Luo, and H. Wang. Robust principal component analysis with non-greedy l 1-norm maximization. In *IJCAI*, pages 1433–1438. AAAI Press, 2011.

[Pang *et al.*, 2010] Y. Pang, X. Li, and Y. Yuan. Robust tensor analysis with l1-norm. *IEEE Transactions on CSVT*, 20(2):172–178, 2010.

[Ye, 2005] J. Ye. Generalized low rank approximations of matrices. *Machine Learning*, 61(1):167–191, 2005.

[Zha *et al.*, 2008] H. Zha, C. Ding, T. Li, and S. Zhu. Workshop on data mining using matrices and tensors. In *KDD*, 2008.