Face Recognition with Robust T-SVD

Shanwen Pu, Siya Liu

June 27, 2020

Abstract

A tensor is a multidimensional array. First-order tensors and second-order tensors can be viewed as vectors and matrices, respectively. Traditional tensor algebra generalize matrix SVD to CP decomposition and HOSVD. But in the recently new tensor framework, T-SVD is another generalization of matrix SVD with nearly the same representation. Moreover, the new tensor product(t-product) can be accelerated by FFT, which improves our T-SVD. We solve the face recognition task on several databases by new tensor robust SVD(TRSVD) and compare our new method with several popular methods, e.g PCA, HOSVD, T-SVD to verify performance and stability.

1 Introduction

To develop automatic procedures for face recognition that are robust with respect to varying conditions is a challenging research problem. PCA(SVD) is a popular technique, but it doesn't perform well when several environment factors are varied. By letting the modes of the tensor represent a different viewing condition, it became possible to improve the precision of the recognition algorithm compared to PCA.

A tensor is a multidimensional array. A first-order tensor is a vector, a second-order tensor is a matrix, and a tensor of order 3 or higher is a higher order tensor. In many cases, it is appropriate to store data in higher-order multidimensional arrays, such as storing photos and videos. As a result, a number of tensor decompositions, such as CANDECOMP/PARAFAC(CP), Tucker, and higher-order SVD and tensor-tensor decomposition, have been developed to facilitate the extension of linear algebra tools to this multilinear context.

Instead of considering images as vetorized objects (in traditional matrix-based PCA and in TensorFaces), the approach we use here is based on T-SVD, which is presented in [4]. This method has an advantage over TensorFaces(based on HOSVD) in that it does not require a least square solve for the coeffcients. Moreover, we will use a pivoted tensor QR decomposition as an alternative tensor decomposition scheme with a potential advantage in updating and downdating the original image data.

2 Related Work

Face recognition task can be tackled by several classes of methods, among which matrix or tensor decomposition based method is of our interest.

PCA based methods PCA was invented in 1901 by Karl Pearson [11], and it has since been a classical technique in image recognition and compression. In simple terms, PCA is used to transform a number of possibly correlated variables into a smaller number of uncorrelated variables known as principal components. The first few principal components explain most of the variation in the original data, while the remaining ones make a negligible contribution. Thus we can describe the data more economically using the first several principal components.

Traditional tensor decomposition based methods TensorFaces [12] was the first algorithm to handle the facial recognition problem by manipulating tensors. In this algorithm, the data is represented as a tensor with different modes for different factors. One way in which the algorithm is consistent with traditional PCA is that the images are still vectorized. The multidimensionality therefore is obtained only by using the other modes to represent other features of the data. The approximation of the tensor is done by use of the multidimensional higher-order SVD (HOSVD), which is a Tucker type of representation.

New tensor framework and T-SVD based methods T-SVD is a new approach invented by Kilmer, etc [7] [6] [5] as another generalization of matrix SVD, by brand new tensor framework [6] . T-SVD representation coincides with matrix SVD and the tensor-tensor product in new framework can be accelerated by fast Fourier transform, which will improve the performance of this method. T-SVD method I and II by hao [4] is based on new T-SVD and T-SVD II adopt the new SVD in Fourier domain, which has the potential for further compression of the information in the database.

Robust PCA and t-SVD Tensor Robust Principal Component (TRPCA) [10] problem which extends the known Robust PCA [2] to the tensor case. TRPCA is a new tensor Singular Value Decomposition (t-SVD) [6] and its induced tensor tubal rank and tensor nuclear norm. Interestingly, TRPCA involves RPCA as a special case when 3rd-order $n_3 = 1$ and thus it is a simple and elegant tensor extension of RPCA. Also numerical experiments verify our theory and the application for the image denoising demonstrates the effectiveness of our method.

3 Your algorithm/model/approach

T-SVD: analogous to the matrix SVD, is a generalization of matrix multiplication for tensors of order three.
Let $\mathcal{A}$ be an $l \times m \times n$ real-valued tensor; then $\mathcal{A}$ can be factored as

$$\mathcal{A} = \mathcal{U} * \mathcal{S} * \mathcal{V}^T$$

with an $l \times m \times n$ orthogonal tensor $\mathcal{U}$, an $m \times m \times n$ orthogonal tensor $\mathcal{V}$, and an $l \times m \times n$ f-diagonal tensor $\mathcal{S}$.
The multiplication is based on a convolution-like operation, which can be implemented efficiently using the Fast Fourier Transform.
Robust T-SVD: extends T-SVD to robust T-SVD.

Consider that we have a 3-way tensor $\mathcal{X}$ such that $\mathcal{X} = \mathcal{L}_0 + \mathcal{S}_0$, where $\mathcal{L}_0$ has low tubal rank and $\mathcal{S}_0$ is sparse. We can recover both the low-rank and the sparse components exactly by simply solving a convex program whose objective is a weighted combination of the tensor nuclear norm and the l_1-norm, i.e

$$\min ||\mathcal{L}||_* + \lambda||\mathcal{E}||_1, \ s.t. \mathcal{X} = \mathcal{L} + \mathcal{E}$$

4 Tensor Algebra

Order of tensor The order of a tensor is the dimensionality of the array needed to represent it, also known as ways or modes. An order-p tensor $\mathcal{A}$ is a p-dimensional array. It can be written as

$$\mathcal{A} = \left(a_{i_1 i_2 \dots i_p}\right) \in \mathbb{R}^{n_1 \times n_2 \times \dots \times n_p}$$

Tensor-Matrix Multiplication An important operation for a tensor is the tensor-matrix multiplication, also known as mode-n product. Given a tensor $\mathcal{A} \in \mathbb{R}^{d_1 \times d_2 \times \dots \times d_N}$ and a matrix $M \in \mathbb{R}^{c_n} \times d_n$, the mode- n product is a tensor

$$\mathcal{B} = \mathcal{A} \times_n M \in \mathbb{R}^{d_1 \times \dots \times d_{n-1} \times c_n \times d_{n+1} \dots \times d_N}$$

where

$$b_{i_1, \dots, i_{n-1}, j_n, i_{n+1}, \dots, i_N} := \sum_{i_n=1}^{d_n} a_{i_1, \dots, i_{n-1}, i_n, i_{n+1}, \dots, i_N} m_{j_n, i_n}$$

for $j_n = 1, 2, \dots, c_n$.

Tensor Decompositions A decomposition of a tensor $\mathcal{A} \in \mathbb{R}^{d_1 \times d_2 \times \dots \times d_N}$ is of the form

$$\mathcal{A} = \mathcal{B} \times_1 S^{(1)} \times_2 S^{(2)} \times \dots \times_N S^{(N)}$$

where $\mathcal{B} \in \mathbb{R}^{c_1 \times c_2 \times \dots \times c_N}$ is called the core tensor, and $S^{(n)} \in \mathbb{R}^{d_n \times c_n}$ for $n = 1, \dots, N$ are called side-matrices. An illustration is given in Figure 2.1.

Fold and Unfold Operators We make use of the fold and unfold operators. Suppose that $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and that each frontal face of the tensor is defined by $n_1 \times n_2$ matrices $\mathcal{A}(:, :, 1), \dots, \mathcal{A}(:, :, n_3)$. Then unfold is defined by

$$\operatorname{unfold}(\mathcal{A}, 1) = \begin{bmatrix} \mathcal{A}(:, :, 1) \\ \mathcal{A}(:, :, 2) \\ \vdots \\ \mathcal{A}(:, :, n_3) \end{bmatrix} \in \mathbb{R}^{n_1 n_3 \times n_2}$$

The second argument of unfold specifies which orientation of the tensor to unfold (see Figure 1.2). For example, $unfold(\mathcal{A}, 1)$ unstacks the tensor according to its front-back faces. Similarly, $unfold(\mathcal{A}, 2)$ refers to unstacking by faces defined by slicing the tensor side to side and $unfold(\mathcal{A}, 3)$ unstacks by slicing top to bottom. See Figure 3.1 for an example.

Block-Circulant Matrix It is possible to create a block-circulant matrix from the slices of a tensor. For example, if $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ with $n_1 \times n_2$ frontal faces $A_1 = \mathcal{A}(:,:,1), \ldots, A_{n_3} = \mathcal{A}(;n_3)$ then

$$\operatorname{bcirc}(\operatorname{unfold}(\mathcal{A},1)) = \begin{bmatrix} A_1 & A_{n_3} & A_{n_3-1} & \ldots & A_2 \\ A_2 & A_1 & A_{n_3} & \ldots & A_3 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ A_{n_3} & A_{n_3-1} & \ddots & A_2 & A_1 \end{bmatrix}$$

Tensor Product Let $\mathcal{A}$ be $n_1 \times n_2 \times n_3$ and $\mathcal{B}$ be $n_2 \times \ell \times n_3$. Then the product $\mathcal{A} * \mathcal{B}$ is the $n_1 \times \ell \times n_3$ tensor

$$\mathcal{A} * \mathcal{B} = \text{ fold } (\text{circ}(\text{ unfold } (\mathcal{A},1)) \cdot \text{unfold}(\mathcal{B},1),1)$$

LEMMA

$$\mathcal{A} * (\mathcal{B} * \mathcal{C}) = (\mathcal{A} * \mathcal{B}) * \mathcal{C}$$

Identity Tensor The $n \times n \times \ell$ identity tensor $\mathcal{I}_{nn\ell}$ is the tensor whose front face is the $n \times n$ identity matrix, and whose other faces are all zeros.

Inverse Tensor An $n \times n \times n$ tensor $\mathcal{A}$ has an inverse $\mathcal{B}$ provided that

$$\mathcal{A} * \mathcal{B} = \mathcal{I}, \quad \text{and} \quad \mathcal{B} * \mathcal{A} = \mathcal{I}$$

Tensor Transpose If $\mathcal{A}$ is $n_1 \times n_2 \times n_3$, then $\mathcal{A}^T$ is the $n_2 \times n_1 \times n_3$ tensor obtained by transposing each of the front-back faces and then reversing the order of transposed faces 2 through n_3 . In other words,

$$\mathcal{A}^T = \text{ fold } \left(\text{ unfold } \left(\mathcal{A}, 1, [1, n_3 : -1 : 2]\right), 1\right)$$

LEMMA Suppose $\mathcal{A}, \mathcal{B}$ are two tensors such that $\mathcal{A} * \mathcal{B}$ and $\mathcal{B}^T * \mathcal{A}^T$ is defined. Then $(\mathcal{A} * \mathcal{B})^T = \mathcal{B}^T * \mathcal{A}^T$.

Orthogonal Tensor An $n \times n \times \ell$ real-valued tensor $\mathcal{Q}$ is orthogonal if $\mathcal{Q}^T * \mathcal{Q} = \mathcal{Q} * \mathcal{Q}^T = \mathcal{I}$.

Permutation Tensor A permutation tensor is an $n \times n \times \ell$ tensor $\mathcal{P} = (p_{ijk})$ with exactly n entries of unity, such that if $p_{ijk} = 1$, it is the only non-zero entry in row i, column j, and slice k.

F-Norm of Tensor Suppose $\mathcal{A} = (a_{ijk})$ is size $n_1 \times n_2 \times n_3$. Then

$$\|\mathcal{A}\|_F = \sqrt{\sum_{i=1}^{n_1} \sum_{j=1}^{n_2} \sum_{k=1}^{n_3} a_{ijk}^2}$$

5 Algorithms

We apply TRPCA for image denoising. Note that the choice of λ is critical for the recovery performance. To verify the correctness of our main results, we set $\lambda = 1/\sqrt{n_1 n_3}$ in the experiments.

Algorithm 1 Traditional Matrix PCA Method.

Input: Training: $\mathbf{I}_i$, $i = 1, 2, \ldots, N$; Testing: $\mathbf{J}$; Truncation index k

Output: Comparison of Test Image with Compressed Training Set

for $i = 1 \quad to \quad N$ **do**

$\mathbf{L}(:, i) \leftarrow$ vectorized $\mathbf{I}_i$

end for

$\mathbf{M} \leftarrow$ mean image

$\mathbf{A} \leftarrow$ mean-deviation form of $\mathbf{L}$

$\mathbf{U} \leftarrow$ left singular vectors of $\mathbf{A}\mathbf{G} \leftarrow \mathbf{U}(:, 1:k)^T\mathbf{A}$

$\mathbf{T} \leftarrow \mathbf{J} - \mathbf{M}$

$\mathbf{t} \leftarrow$ vectorized form of $\mathbf{T}$

$\mathbf{c} \leftarrow \mathbf{U}(:, 1:k)^T\mathbf{t}$

for $j = 1 \quad to \quad N$ **do**

do Calculate $\|c - \mathbf{G}(:, j)\|_F$

end for

Claim that the training image whose coefficient is closest to that of the test image is the match.

Algorithm 2 TensorFaces Method.

Input: Training images $\mathbf{I}_{p,v,i,e}, p = 1, \ldots, P, v = 1, \ldots, V, i = 1, \ldots, I, e = 1, \ldots, E$; Testing Image $\mathbf{J}$

Output: Coefficients of testing images with some tensor basis and comparison

for $p = 1, \ldots, P, v = 1, \ldots, V, i = 1, \ldots, I, e = 1, \ldots, E$ **do**

$\mathcal{L}(p, v, i, e, :) \leftarrow vectorized \quad \mathbf{I}_{p,v,i,e}$

end for

Do the HOSVD to get the core tensor $\mathcal{Z}$ and orthogonal factor matrices $\mathbf{U}_i$

$\mathcal{L} \leftarrow \mathcal{Z} \times_1 \mathbf{U}_p \times_2 \mathbf{U}_v \times_3 \mathbf{U}_i \times_4 \mathbf{U}_e \times_5 \mathbf{U}_{pix}$

$\mathcal{B} \leftarrow \mathcal{Z} \times_2 \mathbf{U}_v \times_3 \mathbf{U}_i \times_4 \mathbf{U}_e \times_5 \mathbf{U}_{pix}$

$\mathcal{T}(1, 1, 1, 1, :) \leftarrow$ vectorized testing image

for $v = 1, \ldots, V, i = 1, \ldots, I, e = 1, \ldots, E$; **do**

$c_{v,i,e} \leftarrow \left(\mathcal{B}^{\dagger}_{v,i,e}\right)^T \mathcal{T}(1, 1, 1, 1, :)$

for $p = 1, \ldots, P$ **do**

$Calculate \left\|c_{v,i,e} - \mathbf{U}_p(p, :)^T\right\|_F$

end for

end for

Claim that the training image whose coefficient is closest to that of the test image is the match.

Algorithm 3 T-SVD Method I.

Input: Training: $\mathbf{I}_i$, $i = 1, 2, \ldots, N$, Testing: $\mathbf{J}$, Truncation index k

Output: Match of Test image against Compressed Representation of Training Set

for $i = 1 \quad to \quad N$ **do**

  $\mathcal{L}(:, i, :) \leftarrow \mathbf{I}_i$

end for

$\mathcal{M} \leftarrow$ mean image

$\mathcal{A} \leftarrow$ mean-deviation form of $\mathcal{L}$

$\mathcal{U} \leftarrow$ left singular vectors of tensor $\mathcal{A}$

$\mathcal{C} \leftarrow \mathcal{U}(:, 1 : k, :)^T * \mathcal{A}$

$\mathcal{T}(:, 1, :) \leftarrow \text{twist}(\mathbf{J} - \mathcal{M})$

$\mathcal{B} \leftarrow \mathcal{U}(:, 1 : k, :)^T * \mathcal{T}$

for $j = 1 \quad to \quad N$ **do**

  Calculate $\|\mathcal{B} - \mathcal{C}(:, j, :)\|_F$

end for

Claim that the training image whose coefficient is closest to that of the test image is the match.

Algorithm 4 Tensor QR Method.

Input: Training: $\mathbf{I}_i$, $i = 1, 2, \ldots, N$; Testing: $\mathbf{J}$, Truncation index k

Output: Coefficients of testing images with a basis of reduced dimension

for $i = 1 \quad to \quad N$ **do**

  $\mathcal{L}(:, i, :) \leftarrow \mathbf{I}_i$

end for

$\mathcal{M} \leftarrow$ mean image

$\mathcal{A} \leftarrow$ mean-deviation form of $\mathcal{L}$

$[Q, \mathcal{R}] \leftarrow$ tensor (pivoted) QR decomposition of $\mathcal{A}$

$\mathcal{C} \leftarrow \mathcal{Q}(:, 1 : k, :)^T * \mathcal{A}$

$\mathcal{T}(:, 1, :) \leftarrow \text{twist}(\mathbf{J} - \mathcal{M})$

$\mathcal{B} \leftarrow \mathcal{Q}(:, 1 : k, :)^T * \mathcal{T}$

for $j = 1 \quad to \quad N$ **do**

  Calculate $\|\mathcal{B} - \mathcal{C}(:, j, :)\|_F$

end for

Claim that the training image whose coefficient is closest to that of the test image is the match.

Algorithm 5 Solve Tensor Robust PCA by ADMM.

Input: tensor data $\mathcal{X}$, parameter λ.

Initialize: $\mathcal{L}_0 = \mathcal{S}_0 = \mathcal{Y}_0 = 0, \rho = 1.1, \mu_0 = 1\mathrm{e}-3, \mu_{\max} = 1\mathrm{e}10, \epsilon = 1\mathrm{e}-8$

while not converged **do**

1. Update $\mathcal{L}_{k+1}$ by

$$\boldsymbol{C}_{k+1} = \underset{\boldsymbol{\mathcal{L}}}{\operatorname{argmin}} \|\boldsymbol{\mathcal{L}}\|_* + \frac{\mu_k}{2} \left\| \boldsymbol{\mathcal{L}} + \boldsymbol{\mathcal{E}}_k - \boldsymbol{X} + \frac{\boldsymbol{Y}_k}{\mu_k} \right\|_F^2$$

2. Update $\mathcal{E}_{k+1}$ by

$$\mathcal{E}_{k+1} = \underset{\varepsilon}{\operatorname{argmin}} \lambda \|\mathcal{E}\|_1 + \frac{\mu_k}{2} \left\| \mathcal{L}_{k+1} + \mathcal{E} - \mathcal{X} + \frac{\mathcal{Y}_k}{\mu_k} \right\|_F^2$$

3.

$$\boldsymbol{\mathcal{Y}}_{k+1} = \boldsymbol{\mathcal{Y}}_k + \mu_k \left(\mathcal{L}_{k+1} + \mathcal{E}_{k+1} - \boldsymbol{X} \right)$$

4. Update μ_{k+1} by $\mu_{k+1} = \min \left(\rho \mu_k, \mu_{\max} \right)$ 5. Check the convergence conditions

$$\|\mathcal{L}_{k+1} - \mathcal{L}_k\|_\infty \leq \epsilon, \|\mathcal{E}_{k+1} - \mathcal{E}_k\|_\infty \leq \epsilon$$
$$\|\mathcal{L}_{k+1} + \mathcal{E}_{k+1} - \mathcal{X}\|_\infty \leq \epsilon$$

end while

References

[1] Brett W Bader, Tamara G Kolda, et al. Matlab tensor toolbox version 2.5. *Available online, January*, 7, 2012.

[2] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[3] Jie Chen and Yousef Saad. On the tensor svd and the optimal low rank orthogonal approximation of tensors. *SIAM journal on Matrix Analysis and Applications*, 30(4):1709–1734, 2009.

[4] Ning Hao, Misha E Kilmer, Karen Braman, and Randy C Hoover. Facial recognition using tensor-tensor decompositions. *SIAM Journal on Imaging Sciences*, 6(1):437–463, 2013.

[5] Misha E Kilmer, Karen Braman, Ning Hao, and Randy C Hoover. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM Journal on Matrix Analysis and Applications*, 34(1):148–172, 2013.

[6] Misha E Kilmer and Carla D Martin. Factorization strategies for third-order tensors. *Linear Algebra and its Applications*, 435(3):641–658, 2011.

[7] Misha E Kilmer, Carla D Martin, and Lisa Perrone. A third-order generalization of the matrix svd as a product of third-order tensors. *Tufts University, Department of Computer Science, Tech. Rep. TR-2008-4*, 2008.

[8] Tamara G Kolda. Orthogonal tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 23(1):243–255, 2001.

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

[10] Canyi Lu, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. Tensor robust principal component analysis: Exact recovery of corrupted low-rank tensors via convex optimization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5249–5257, 2016.

[11] Karl Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[12] M Alex O Vasilescu and Demetri Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In *European conference on computer vision*, pages 447–460. Springer, 2002.

[13] John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization. In *Advances in neural information processing systems*, pages 2080–2088, 2009.