Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence

Supervised Nonnegative Tensor Factorization with Maximum-Margin Constraint

Fei Wu and **Xu Tan**
College of Computer Science
Zhejiang University, China
{wufei, tanxu}@zju.edu.cn

Yi Yang
School of Information Technology
and Electrical Engineering
the University of Queensland, Australia
yee.i.yang@gmail.com

Dacheng Tao
Centre for Quantum Computation
and Intelligent Systems
University of Technology, Sydney, Australia
Dacheng.Tao@uts.edu.au

Siliang Tang, Yueting Zhuang
College of Computer Science
Zhejiang University, China
{siliang, yzhuang}@zju.edu.cn

Abstract

Non-negative tensor factorization (NTF) has attracted great attention in the machine learning community. In this paper, we extend traditional non-negative tensor factorization into a supervised discriminative decomposition, referred as Supervised Non-negative Tensor Factorization with Maximum-Margin Constraint (SNTFM^2). SNTFM^2 formulates the optimal discriminative factorization of non-negative tensorial data as a coupled least-squares optimization problem *via* a maximum-margin method. As a result, SNTFM^2 not only faithfully approximates the tensorial data by additive combinations of the basis, but also obtains a strong generalization power to discriminative analysis (in particular for classification in this paper). The experimental results show the superiority of our proposed model over state-of-the-art techniques on both toy and real world data sets.

Introduction

In many real-world applications, data intrinsically come in the form of *tensors*, or multi-dimensional arrays. For example, a gray image can be represented spontaneously as a 2-nd order tensor. In the last decade, decompositions and low-rank approximations of tensors have been studied extensively in ample fields, including computer vision, bioinformatics, neuroscience, and data mining (Kolda and Bader 2009; Tao et al. 2007; Wu, Liu, and Zhuang 2009). Meanwhile, the non-negativity constraint has been proved to be indispensable and useful when dealing with non-negative data in face recognition (Wang et al. 2005), biological analysis (Kim and Park 2007), psychometric (Murakami and Kroonenberg 2003) and gait recognition (Tao et al. 2006). As a result, many researchers focused on Non-negative Tensor Factorization (NTF) in the past few years (Shashua and Hazan 2005; Kim and Park 2012; Mørup, Hansen, and Arnfred 2008; Friedlander and Hatz 2008). NTF, as a more general form of the well-studied Non-negative Matrix Factorization (NMF), aims to obtain a parts-based representation of high-dimensional data object, so that the target data can be expressed by multi-linear combination of non-negative components. In comparison with NMF, in NTF the structural information is reserved in the data, while the vectorization of the object tensor in NMF may result in information loss. However, most of algorithms proposed for NTF or NMF act as unsupervised manners that cannot exploit the inherent discriminative priors (corresponding class labels) of the data objects. This knowledge is in fact useful in many real world applications, such as images with tags etc.

This paper is dedicated to developing a supervised tensor-based factorization, referred as Supervised Non-negative Tensor Factorization with Maximum-Margin constraint (SNTFM^2). SNTFM^2 extends traditional non-negative tensor factorization into a supervised decomposition *via* a maximum-margin method (specifically a support vector machine), which is formulated by coupling the approximation of tensorial data (in terms of faithful reconstruction using additive combinations of the basis) with a maximum margin constraint (in terms of a generalized discriminative power). Maximum margin classifiers such as support vector machines (SVMs) (Cherkassky 1997) and Maximum-margin Markov Networks (M3N) (Roller 2004), have been successfully applied in a wide range of classification problems. Maximum-margin methods commonly construct an optimal separating hyperplane that maximizes the margin (i.e. the distance between the hyperplane and the nearest data point of each class) by mapping the input space into an associated reproducing kernel Hilbert space. It has been shown that such methods are appealing due to the existence of strong generalizations, derived from the well-known *kernel trick* of the learning algorithm. (Kumar, Kotsia, and Patras 2012) proposes a method in which the acquired projections are chosen so that they maximize the discriminative ability through a maximum margin method. (Das Gupta and Xiao 2011) extends the maximum-margin NMF into a kernel (non-linear) one. However, their work are limited to the factorization of vectorized data instead of tensorial data objects.

This work appreciates both the decomposition as well as

the classification task. The decomposition and classification are usually handled independently, while our framework naturally integrates them together as a whole. In contrast to traditional classification methods, our procedure of decomposition has the potential to enhance the classification performance, by identifying the additive basis and discriminative features. Moreover, our work has paid more attention to the computing efficiency, as the data set represented by a tensor usually is much larger than a matrix. In the proposed algorithm, we devise the optimization framework into groups and update one column vector at a time, so that variables can be updated simultaneously.

Our proposed SNTFM^2 has the following two *aspects*: a) the proposed framework jointly deals with the task of decomposition and classification which utilizes the discriminative labels and achieves a greater generalized discriminating power; and b) an efficient optimization approach to solve the formulation. The experimental results show the superiority of our proposed model over state-of-the-art techniques on both toy and real world data sets.

Notations and Preliminaries

A *tensor* is a multidimensional array (Kolda and Bader 2009), whose *order* is the number of dimensions. Matrices (tensors of order two) will be denoted by boldface capital letters, e.g., $\mathbf{A}$, vectors (tensor of order one) by boldface lowercase letters, e.g., $\mathbf{a}$, scalars (tensors of order zero) by lowercase letters, e.g., a. The higher-order tensors (order three or higher) are denoted by Euler script calligraphic letters, e.g., an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$. The (i,j)-th element of a matrix $\mathbf{A}$ is denoted by a_{ij}, while the j-th column of a matrix $\mathbf{A}$ is denoted by $\mathbf{a}_j$. Similarly, the elements of an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$ will be denoted by $x_{i_1 i_2 \ldots i_N}, i_l = 1, 2, \ldots, I_l, l = 1, 2, \ldots, N$.

The n-mode matricization, also known as unfolding or flattening, of an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, denoted by $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times (I_1 \ldots I_{n-1} I_{n+1} \ldots I_N)}$, $(n = 1, 2, \ldots, N)$ is the process of reordering the elements of an N-way array into a matrix.

The n-mode product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ with a matrix $\mathbf{U} \in \mathbb{R}^{J \times I_n}$, denoted by $\mathcal{X} \times_n \mathbf{U}$, is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$. Elementwise, we have

$$(\mathcal{X} \times_n \mathbf{U})_{i_1 \ldots i_{n-1} j i_{n+1} \ldots i_N} = \sum_{i_n=1}^{I_n} x_{i_1 i_2 \ldots i_N} u_{j i_n}.$$

In terms of unfolded tensors, the n-mode product can be expressed as

$$\mathcal{Y} = \mathcal{X} \times_n \mathbf{U} \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{U}\mathbf{X}_{(n)}.$$

The rank-one tensor is an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ if it can be written as the outer product of N vectors:

$$\mathcal{X} = \mathbf{x}^{(1)} \circ \mathbf{x}^{(2)} \circ \cdots \circ \mathbf{x}^{(N)},$$

where the symbol "$\circ$" represents the vector outer product.

Non-negative Tensor Factorization

There are mainly two popular kinds of tensor decomposition approaches: CANDECOMP/PARAFAC decomposition (Kiers 2000), or CP for short, and Tucker decomposition (Kolda and Bader 2009).

CP decomposition CP decomposition aims to expressing a tensor as the sum of a finite number of rank-one tensors

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \circ \mathbf{u}_r^{(2)} \cdots \circ \mathbf{u}_r^{(N)} \triangleq [\![\mathbf{U}^{(1)}, \mathbf{U}^{(2)}, \ldots, \mathbf{U}^{(N)}]\!].$$

The operator "$\circ$" is the outer product of vectors and the factor matrices $\mathbf{U}^{(k)} = [\mathbf{u}_1^{(k)}, \ldots, \mathbf{u}_R^{(k)}] \in \mathbb{R}^{I_k \times R}, k = 1, 2, \ldots, N$.

Tucker decomposition Tucker decomposition can be viewed as a form of higher-order PCA. It decomposes a tensor into a core tensor multiplied (or transformed) by a matrix along each mode as follows:

$$\begin{aligned}\mathcal{X} &\approx \mathcal{C} \times_1 \mathbf{U}_1 \cdots \times_N \mathbf{U}_N \\ &= \sum_{r_1=1}^{R_1} \cdots \sum_{r_N=1}^{R_N} c_{r_1 \ldots r_N} \mathbf{u}_{r_1} \circ \cdots \circ \mathbf{u}_{r_N} \\ &\triangleq [\![\mathcal{C}; \mathbf{U}_1, \mathbf{U}_2, \ldots, \mathbf{U}_N]\!],\end{aligned}$$

where $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the original tensor. $\mathbf{U}_1 \in \mathbb{R}^{I_1 \times R_1}, \ldots, \mathbf{U}_N \in \mathbb{R}^{I_N \times R_n}$ are the factor matrices, which are usually columnwise orthogonal, and can be expressed as the principle components in each mode. $\mathcal{C} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ is called the core tensor, which accounts for all possible linear interactions between the components of each mode.

It is interesting to note that the CP decomposition can be regard as a special case of Tucker decomposition (Kolda and Bader 2009), where $R_1 = R_2 = \cdots = R_N$ and the core tensor is superdiagonal, which means every mode of the tensor is of the same size and its elements remain constant under any permutation of the indices.

Although there are some other tensor decompositions proposed afterwards, they can be regarded as variants or combinations of CP and Tucker, such as (Harshman and Lundy 1996; Martınez-Montes et al. 2004; Harshman, Hong, and Lundy 2003).

It is desirable to impose nonnegativity constraint on tensor factorizations and thereby facilitate easier interpretation when analyzing non-negative data. NTF has been widely studied (Friedlander and Hatz 2008; Kim and Park 2012; Liu, Wonka, and Ye 2012).

Given a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the NTF of $\mathcal{X}$, in terms of the Tucker decomposition model is as follows:

$$\begin{aligned}\min_{\mathbf{U}_1, \ldots, \mathbf{U}_N} : \quad & \|\mathcal{X} - \mathcal{C} \times_1 \mathbf{U}_1 \cdots \times_N \mathbf{U}_N\|^2 \\ s.t. \quad & \mathbf{U}_n \geq 0, 1 \leq n \leq N.\end{aligned}$$

It is obvious that the goal of NTF (given by Tucker decomposition model) is to decompose N-order $\mathcal{X}$ with a nonnegative constraint into a set of $N+1$ constitutive factors $\mathcal{C}$, $\mathbf{U}_1, \ldots, \mathbf{U}_N$ that can be combined to form an approximation of $\mathcal{X}$.

The Framework of SNTFM^2

Given an N-order tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_n}$, without loss of generality, we assume the s-th mode denotes the samples, while I_s denotes the number of samples. The class label of the i-th sample is denoted as $y_i \in \{-1, 1\}, i = 1, 2, \ldots, I_s$. In SNTFM^2, we pursue a discriminative decomposition by coupling NTF and a maximum margin classifier (in particular SVM is conducted) for classification. We cast the mission as learning a prediction function $f : \mathcal{X} \mapsto \mathbf{Y}$ that maps the space of input samples $\mathcal{X}$ to the space of binary classification labels $\mathbf{Y}$ based on a training set of input/output pairs.

The non-negative discriminative tensor factorization by Tucker decomposition model for classification can be written as follows:

$$\begin{aligned} \min_{\mathbf{U}_1,\ldots,\mathbf{U}_N} : \quad & \|\mathcal{X} - \mathcal{C} \times_1 \mathbf{U}_1 \cdots \times_N \mathbf{U}_N\|^2 + \Omega(\mathcal{X}) \\ s.t. \quad & \mathbf{U}_n \geq 0, 1 \leq n \leq N, \end{aligned} \tag{1}$$

where $\Omega(\mathcal{X})$ is a regularized penalty, e.g., imposed on the priors of classification for $\mathcal{X}$.

We can write the objective function in Eq.(1) in terms of unfolded tensors as follows:

$$\min_{\mathbf{U}_n} \|\mathbf{X}_{(n)} - \mathbf{U}_n \mathbf{B}_{(n)}\|^2 + \Omega(\mathbf{X}_{(n)}), \tag{2}$$

where

$$\begin{aligned} \mathbf{B}_{(n)} = (\mathcal{C} \times_1 \mathbf{U}_1 \times_2 \cdots \times_{n-1} \mathbf{U}_{n-1} \\ \times_{n+1} \mathbf{U}_{n+1} \times_{n+2} \cdots \times_N \mathbf{U}_N)_{(n)}. \end{aligned} \tag{3}$$

To simplify the optimization problem, we take transpose and separate the objective function into I_n columns of the matrix $\mathbf{U}_n^T$ resulting in I_n independent optimization problems:

$$\begin{aligned} \min_{\mathbf{u}_i} : \quad & \|\mathbf{x}_i - \mathbf{B}_{(n)}^T \mathbf{u}_i\|^2 + \Omega(\mathbf{x}_i) \\ s.t. \quad & \mathbf{u}_i \geq 0, 1 \leq i \leq I_n, \end{aligned} \tag{4}$$

where

$$\mathbf{X}_{(n)} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{I_n}]^T,$$

$$\mathbf{U}_n = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{I_n}]^T.$$

After the non-negative decomposition, now we transfer the classification of tensorial data from the $\{\mathcal{X}, \mathbf{Y}\}$ into $\{\mathbf{u}_i, \mathbf{Y}\}(1 \leq i \leq I_n)$ with a predefined penalty $\Omega(\mathcal{X})$. Moreover, we identify the optimization framework into groups and update one column vector at a time, so that variables can be updated simultaneously.

As stated before, the goal of our proposed approach attempts to find a non-negative decomposition for a tensorial data as well as learn a classifier in the factorized space. Since each principal component $\mathbf{u}_i (1 \leq i \leq I_s)$ in the s-th mode is corresponding to each sample, we reformulate the optimal factorization of non-negative tensorial data in their s-th mode as a coupling least-squares optimization problem via a maximum-margin method, the reformulation can be denoted as a following minimization problem:

$$\begin{aligned} \min_{\mathbf{u}_i^{(s)}} : \quad & \gamma \|\mathbf{x}_i^{(s)} - \mathbf{B}_{(s)}^T \mathbf{u}_i^{(s)}\|^2 + \\ & \left(\mathbf{w}^T \mathbf{w} + \tau \sum_{i=1}^{I_s} L(y_i, \mathbf{w} \cdot \mathbf{u}_i^{(s)} + b) \right) \\ s.t. \quad & \mathbf{u}_i^{(s)} \geq 0, 1 \leq i \leq I_s, \end{aligned} \tag{5}$$

where γ and τ are parameters to control the approximate error and classification loss respectively, and b is the bias term that can be viewed as a component of the weight vector $\mathbf{w}$. $L(y, t) = \max(0, 1 - yt)^p$ could be any loss function, specially, quadratic loss when $p = 2$.

The traditional solution for SVM classifiers is generally obtained in the dual domain (Cherkassky 1997; Schölkopf and Smola 2002). However, since the weight vector $\mathbf{w}$ and the components $\mathbf{u}_i^{(s)}$ are inherently coupled in Eq.(5), it is complicated to obtain the dual formulation of Eq.(5). Inspired by the idea of primal optimizations of non-linear SVMs (Bottou and Weston 2007), the well-known *kernel trick* is introduced here to implicitly capture the non-linear structures. Therefore, we replace $\mathbf{w}$ with a functional form $f(\mathbf{x})$ as follows

$$f(\mathbf{x}) = \sum_{i=1}^{I_s} \alpha_i \mathbf{k}(\mathbf{x}_i, \mathbf{x}), \tag{6}$$

where $\mathbf{k}(\mathbf{x}, \mathbf{y})$ is a kernel as given by the representer theorem (Kimeldorf and Wahba 1970). After replacing $\mathbf{w}$ by $f(\mathbf{x})$, Eq.(5) is revised as follows:

$$\begin{aligned} \min_{\mathbf{u}_i^{(s)}} : \quad & \gamma \|\mathbf{x}_i^{(s)} - \mathbf{B}_{(s)}^T \mathbf{u}_i^{(s)}\|^2 + \lambda \sum_{i,j=1}^{I_s} \alpha_i \alpha_j \mathbf{k}(\mathbf{u}_i^{(s)}, \mathbf{u}_j^{(s)}) \\ & + \sum_{i=1}^{I_s} L(y_i, \sum_{j=1}^{I_s} \mathbf{k}(\mathbf{u}_i^{(s)}, \mathbf{u}_i^{(s)}) \alpha_i) \\ s.t. \quad & \mathbf{u}_i^{(s)} \geq 0, \end{aligned} \tag{7}$$

where $\lambda = 1/\tau$ is the weight between the loss function and the margin and γ is the relative weight between the generative and the discriminative components.

Writing the kernel matrix $\mathbf{K}$, such that $k_{ij} = \mathbf{k}(\mathbf{u}_i, \mathbf{u}_j)$, and $\mathbf{k}_i$ is the i^{th} column of $\mathbf{K}$, we get

$$\begin{aligned} \min_{\mathbf{u}_i^{(s)}; \boldsymbol{\alpha}} : \quad & \gamma \|\mathbf{x}_i^{(s)} - \mathbf{B}_{(s)}^T \mathbf{u}_i^{(s)}\|^2 + \lambda \boldsymbol{\alpha}^T \mathbf{K} \boldsymbol{\alpha} \\ & + \sum_{i=1}^{I_s} L(y_i, \mathbf{K}_i^T \boldsymbol{\alpha}) \\ s.t. \quad & \mathbf{u}_i^{(s)} \geq 0. \end{aligned} \tag{8}$$

For other modes except the s-th mode (denoted as $\hat{s}$, i.e, $\hat{s} \in \{1, \ldots, s-1, s+1, \ldots, N\}$) of $\mathcal{X}$, since there's no supervised information to be used, we can utilize the original non-negative tensor factorization method to obtain the principle components of other modes. Like NMF, NTF usually

induces a sparse decomposition of the tensorial data (Hoyer 2004) and since the sparseness given by traditional NTF is somewhat of a side-effect rather than a goal, we cannot in any way control the degree to which the representation is sparse. As a result, it is essential to control sparseness of NTF explicitly.

Here, we adopt a sparse solution in (Mørup, Hansen, and Arnfred 2008; Liu, Wonka, and Ye 2012) to get a sparse decompositions of the $\hat{s}$-th mode. The sparse factorization with ℓ_1 norm penalty for the $\hat{s}$-th mode is:

$$\begin{aligned} \min_{\mathbf{u}_i^{(\hat{s})}} : \quad & \|\mathbf{x}_i^{(\hat{s})} - \mathbf{B}_{(\hat{s})}^T \mathbf{u}_i^{(\hat{s})}\|^2 + \eta_{\hat{s}} |\mathbf{u}_i^{(\hat{s})}| \\ s.t. \quad & \mathbf{u}_i^{(\hat{s})} \geq 0, \hat{s} \neq s, \hat{s} \in \{1, \ldots, s-1, s+1, \ldots, N\}, \end{aligned} \tag{9}$$

where $\eta_{\hat{s}}$ controls the sparsity of decomposition of the s-th mode for $\mathcal{X}$. One optimum solution of Eq.(9) can be computed as follows (Liu, Wonka, and Ye 2012):

$$u_{ij}^{(\hat{s})} = \begin{cases} \frac{t - \eta_{\hat{s}}}{b_j b_j^T}, & t > \eta_{\hat{s}} \\ 0, & otherwise \end{cases} \tag{10}$$

where $u_{ij}^{(\hat{s})}$ is the elements of $\mathbf{u}_i^{(\hat{s})}$,

$$t = \mathbf{b}_j (\mathbf{B}_{(\hat{s})}^T \mathbf{u}_i^{(\hat{s})} - \mathbf{b}_j^T \mathbf{x}_i)$$

$$\mathbf{B}_{(\hat{s})} = [\mathbf{b}_1^T, \mathbf{b}_2^T, \ldots \mathbf{b}_{R_{\hat{s}}}^T]^T.$$

Optimization of Discriminant Components and Classification Coefficients

Now we discuss the solution of Eq. (8). As the optimization problem in Eq. (8) is convex, many of well-known methods can be conducted to obtain the solution, such as gradient descent and Newton's method. Gradient descent is simple, but has been observed to converge slowly in practice. While Newton's method typically enjoys a faster convergence rate but the computation of the invert Hessian may be expensive when the size of Hessian is large. Moreover, we can't guarantee the Hessian is invertible for any kernels. Hence, we adopt conjugate gradient here for updating, which does not need to compute the invert of the Hessian and can achieve a reasonable solution within only a couple of steps.

Update $\boldsymbol{\alpha}$ The first order gradient of Eq. (8) with respect to $\boldsymbol{\alpha}$ is:

$$\nabla_{\boldsymbol{\alpha}} = (2\lambda \mathbf{K}\boldsymbol{\alpha} + \sum_{i=1}^{I_s} \mathbf{k}_i \frac{\partial L}{\partial t}|_{t=\mathbf{k}_i^T \alpha}). \tag{11}$$

As mentioned before, L can be any loss function. Different loss functions such as the Huber and Hinge loss can be seamlessly incorporated into our proposed model. Here, we consider an easier case, i.e., the L_2 penalization of the training errors:

$$L(y_i, f(\mathbf{u}_i^{(s)})) = \max(0, 1 - y_i f(\mathbf{u}_i^{(s)}))^2. \tag{12}$$

Similar to (Bottou and Weston 2007), for a given value of the vector $\boldsymbol{\alpha}$, we call a point $\mathbf{u}_i^{(s)}$ the support vector when $y_i f(\mathbf{u}_i^{(s)}) < 1$, that is, if the loss on this point is nonzero. After reordering the points such that the first n_v points are support vectors, the gradient with respect to $\boldsymbol{\alpha}$ is:

$$\nabla_{\boldsymbol{\alpha}} = 2(\lambda \mathbf{K}\boldsymbol{\alpha} + \mathbf{K}\mathbf{I}^0(\mathbf{K}\boldsymbol{\alpha} - \mathbf{Y})), \tag{13}$$

where $\mathbf{I}^0$ is the $I_s \times I_s$ diagonal matrix with the first n_v entries (number of support vectors) being 1 and others 0, given by

$$\mathbf{I}^0 = \begin{bmatrix} \mathbf{I}_{n_v} & 0 \\ 0 & 0 \end{bmatrix}.$$

The Hessian, with respect to $\boldsymbol{\alpha}$ is

$$\mathbf{H}_{\boldsymbol{\alpha}} = 2(\lambda \mathbf{K} + \mathbf{K}\mathbf{I}^0 \mathbf{K}). \tag{14}$$

Update $\mathbf{u}_i^{(s)}$ Different from previous work on combining SVM cost with NMF, such as (Kumar, Kotsia, and Patras 2012), our proposed method is extended to one *kernelized* approach for tensorial data. Kernelized SVM is appealing due to their good generalization performance.

In this paper, we conduct the inner product kernel as follows:

$$\mathbf{k}(\mathbf{u}_i^{(s)}, \mathbf{u}_j^{(s)}) = \mathbf{u}_i^{(s)T} \mathbf{u}_j^{(s)}. \tag{15}$$

For this kernel the gradient and the Hessian with respect to $\mathbf{u}_i^{(s)}$ can be written as

$$\begin{aligned} \nabla_{\mathbf{u}_i^{(s)}} = & -2\gamma \mathbf{B}_{(s)} \mathbf{x}_i^{(s)} + 2\gamma (\mathbf{B}_{(s)} \mathbf{B}_{(s)}^T) \mathbf{u}_i^{(s)} + 2\lambda \alpha_i \sum_{j=1}^{I_s} \alpha_j \mathbf{u}_j^{(s)} \\ & + 2 \left(\sum_{j=1}^{n_v} l_j \alpha_j \mathbf{u}_j^{(s)} [i \in n_v] + \alpha_i \sum_{j=1}^{n_v} l_j \mathbf{u}_j^{(s)} \right), \end{aligned} \tag{16}$$

$$\mathbf{H}_{\mathbf{u}_i^{(s)}} = 2\gamma (\mathbf{B}_{(s)} \mathbf{B}_{(s)}^T) + (2\lambda \alpha_i^2 + 4 l_i \alpha_i [i \in n_v]) \mathbf{I}_{ns}, \tag{17}$$

where $\mathbf{I}_{ns}$ is an identity matrix of size I_s and $[i \in n_s]$ is an indicator function indicating that the term is present in the equations only when the index i belongs to the set of support vectors.

Choice of Core tensor A key issue in applying the tensor decomposition is how to construct the core tensor. An inappropriate core tensor will make the decomposition infeasible. For example, given a tensor of size $1000 \times 10 \times 3$, a corresponding high-mode core tensor of size $20 \times 20 \times 20$ obviously leads to redundant computing, while a low-mode core tensor of size $3 \times 3 \times 3$ may result in a low accuracy. As a result, it's imperative to devise a core tensor carefully when dealing with unbalanced tensors.

According to (Liu, Wonka, and Ye 2012), we let the core tensor $\mathcal{C}$ satisfy the following requirements, that will make the core tensor akin to an identity tensor.

1. Consists of "0"s or "1"s;
2. All rows of $\mathbf{C}_{(n)}$ are orthogonal;
3. $\mathbf{C}_{(n)}$ is of full rank for any n.

The details of the proposed supervised non-negative tensor factorization with maximum margin constraint(SNTFM^2) are summarized in Algorithm 1.

Algorithm 1: The Algorithm of SNTFM^2

Input: The tensorial training data and their corresponding class labels, i.e., $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I_1 \times I_2 \cdots \times I_N}$, where s-th mode denotes the number of samples and $y_i \in \{-1, +1\}, i = 1, 2, \ldots, I_s$, given a core tensor $\mathcal{C}$ and a kernel function $\mathbf{k}$.
Output: The principle decomposition components of each mode $\{\mathbf{U}_1, \ldots, \mathbf{U}_N\}$ and classifier coefficients vector $\boldsymbol{\alpha}$.

```
Initialize {U_1, ..., U_N} and coefficients
vector α.
repeat
    for n = 1 to N do
        Compute B_(n) using Eq. (3).
        if n = s
            Compute kernel matrix K.
            Update α using Eq. (13), Eq. (14).
        end if
        for i = 1 to I_n do
            if n = s
                Update u_i^(s) using Eq.(16), Eq.(17)
            else if n ≠ s
                Update u_i^(ŝ) using Eq. (10).
            end if
        end
    end
until iter ≥ MAXITER or convergence.
```

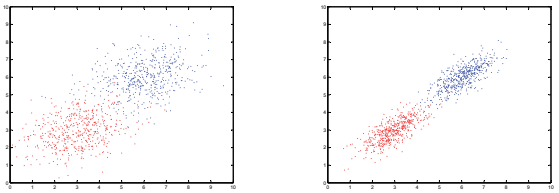
Experiments

Experiments on Toy Data Set

Here the synthetic data is used to discuss the interpretable nature of our proposed SNTFM^2 for the discriminative analysis. We generate two multivariate normal distribution, denoted as ND, with means at $[6, 6]$, $[3, 3]$ and covariances are both $[1.5, 0.4; 0.4, 1]$. Each distribution has 500 data points. The labels for the data points are assigned as $\{-1, 1\}$ according to their different distributions. The original 1000 data points are illustrated in Figure 1(a). In Figure 1(a), some data from two classes are overlapping on some places, so that directly applying SVM or other methods for the classification may not achieve a satisfied performance.

The NTF decomposition has the ability to project the original data onto the space of additive basis. We represent the original data as a 1000×2 tensor, and apply the proposed SNTFM^2, setting the core tensor as an identity matrix of size 2, and the parameters are tuned to achieve the best performance. We can observe from Figure 1(b) that the original data in their projected space by SNTFM^2 decomposition are much more easily to be separated.

We then conduct experimental comparison on some basic binary classification data sets from the UCI Machine Learning Reposity (http://archive.ics.uci.edu/ml/), i.e., Ionosphere, Musk, Pima. In Ionosphere, there are 34 attributes from 351 instances. In Musk, there are 168 attributes from 6598 instances. While, there are 8 attributes from 768 instances in Pima. The comparison methods are SVM, 1-Nearest Neighbor (1NN), Naive Bayes (NB). We randomly pick a half of the data for training, and the rest for testing. The results are reported in Table 1. The results shown in boldface are the best. We can observe that on Pima data, our proposed method works little worse than SVM. The reason is probably that the nonnegative tensor factorization of low-dimension data (i.e.,Pima) is trivial for classification. However, the performance by our approach is comparable with the performance by SVM. While in general, the proposed SNTFM^2 outperforms the other methods.



(a) (b)

Figure 1: (a) The two dimensional data points. The red-colored data points belong to the positive class and the blue-colored data points belong to the negative class. The data from two classes are somehow overlapped. (b) The original data in their projected space by SNTFM^2 decomposition. The data from two classes are easily to be separated.

Table 1: The Comparisons of Classification Accuracy

Data Set	SVM	1NN	NB	SNTFM^2
ND	94.2 %	93.6 %	93.2 %	**96.8** %
Ionosphere	88.6 %	86.1 %	85.9 %	**89.8** %
Musk	83.6 %	81.1 %	66.8 %	**85.7** %
Pima	**77.3** %	65.6 %	74.5 %	77.0 %

As stated before, our proposed algorithm leads to an efficient solution, in which we devise the optimization framework into groups and update one column vector independently at a time, so that variables can be updated simultaneously. To verify the efficiency for tensorial data decomposition by SNTFM^2, we conduct experiments on synthetic data sets. The compared algorithms on synthetic data are sparse nonnegative CP decomposition (abbreviated as NCP) (Mørup et al. 2006), sparse nonnegative Tucker decomposition (abbreviated as NTucker) (Mørup, Hansen, and Arnfred 2008). The synthetic tensor data is of size $200 \times 200 \times 200$. For SNTFM^2 algorithm, the class labels are assigned. We apply the same random initialization, and set the sparseness coefficients $\eta = 0.1$, other parameters are tuned to achieve best performance. We compare their factorization efficiency by different core tensors of size $10 \times 10 \times 10$, $20 \times 20 \times 20$ and $40 \times 40 \times 40$. All the comparisons are implemented on the same computer environment. It can be observed from Table 2 that our proposed SNTFM^2 outperforms the other methods in terms of factorization efficiency.

Experiments on Real World Data Sets

We then conduct experimental comparisons for the task of classification on two public benchmark data sets in the real world: facial images and human motion data, both of which can be expressed in tensors spontaneously.

1. *Facial Images:* We conduct experiments on facial images FG-NET dataset(Agarwal et al. 2010) for the classification according to age. The FG-NET data set comprises of 1002 facial images of 82 people being from 0 to 69 years old. All the acquired images are resized to 40×30 pixels. We randomly select 800 of them as training set and the rest as testing set.

Table 2: The Comparisons of Factorization Efficiency

Method	Time(sec)	Iterations
Core tensor size: $10 \times 10 \times 10$		
SNTFM^2	**16.8642**	**29**
NCP	135.863	146
NTucker	285.955	277
Core tensor size: $20 \times 20 \times 20$		
SNTFM^2	**39.0048**	**44**
NCP	284.566	218
NTucker	182.608	239
Core tensor size: $40 \times 40 \times 40$		
SNTFM^2	**55.3347**	**61**
NCP	97.5724	114
NTucker	2241.18	452

2. *Motion Data:* In the case of motion data, we conduct experiments on classifying the human pose, and carry experiments using publicly available data set, the Carnegie Mellon University's Graphics Lab motion capture database(Piazza et al. 2009). We process the data to form each motion a 3×49 tensor. We randomly choose 48 classes for the experiment, including 50000 motions for training and the rest 19363 for testing.

We evaluate the performance on the task of classification in terms of three kinds of metrics. The first one is accuracy. The second one is area under the ROC curve, called AUC (Fawcett 2006), we use the MacroAUC (average on AUC of all the classes) and the MicroAUC (the global calculation of AUC regardless of classes). The last one is the harmonic mean of precision and recall, called F1 score(Fawcett 2006), the MacroF1 (average on F1 scores of all the classes) and the MicroF1 (the global calculation of F1 regardless of classes).

We compare our proposed SNTFM^2 algorithm with the following five methods, of which two are vector-based methods and the others are tensor-based methods. The parameters of all the methods are tuned using cross-validation:

- **SVM**: the Support Vector Machine, a typical maximum-margin classifier, is applied after vectorizing the input tensor on each class using the one vs. all scheme. We use an open source software LIBSVM (Chang and Lin 2011).
- **NMFSVM**: a non-negative matrix factorization base regularizer for SVM (Das Gupta and Xiao 2011).
- **STM**: the Support Tucker Machine (Kotsia and Patras 2011), a tensor-based model of SVM.
- **LTR**: a modified tensor based logistic regression method for classification (Guo, Kotsia, and Patras 2012).
- **DNTF**: CP decomposition together with a Linear Discriminant Analysis (LDA) approach (Zafeiriou 2009).

All the methods are repeated ten times for ten random training/test partitions, and we report the average results. Table 3 shows the performance in terms of the average Accuracy, MacroAUC, MicroAUC, MacroF1 and MicroF1. The results shown in boldface are best results. From the results in 3, we can make the following observations:

- The proposed SNTFM^2 achieves the best performance of classifications in all of metrics for all the other approaches thanks to its nonnegative and maximum-margin natures.
- The classification performances by tensor-based approaches (e.g., SNTFM^2, STM, LTR and NTF+LDA) have a better performance than those vectorized-based approaches(e.g., SVM and NMFSVM).This is easy to understand a tensor can inherently preserve the structure embedded in original data.

Table 3: Performance comparison of different algorithms in terms of Accuracy, MacroAUC, MicroAUC, MacroF1 and MicroF1. The results shown in boldface are best results.

A. Comparison on Facial Images

Method	Accuracy	MacroAUC	MicroAUC	MacroF1	MicroF1
SNTFM^2	**0.9008**	**0.7345**	**0.7458**	**0.7213**	**0.7618**
SVM	0.8310	0.6551	0.6846	0.6331	0.6784
NMFSVM	0.8821	0.6989	0.7064	0.6871	0.7127
STM	0.8718	0.7253	0.7443	0.7037	0.7435
LTR	0.8848	0.7246	0.7250	0.7119	0.7534
DNTF	0.8894	0.7205	0.7314	0.7028	0.7458

B. Comparison on Motion Data

Method	Accuracy	MacroAUC	MicroAUC	MacroF1	MicroF1
SNTFM^2	**0.8879**	**0.7616**	**0.7857**	**0.7212**	**0.7407**
SVM	0.8216	0.6913	0.7231	0.6602	0.6768
NMFSVM	0.8475	0.7421	0.7527	0.6825	0.6954
STM	0.8549	0.7585	0.7844	0.7093	0.7364
LTR	0.8697	0.7469	0.7761	0.7064	0.7335
DNTF	0.8416	0.7205	0.7322	0.7191	0.7323

Conclusion

This paper proposes a supervised non-negative tensor factorization with maximum-margin constraint (SNTFM^2) for classification. The SNTFM^2 method is attractive and in particularly appropriate for discriminant analysis due to its nonnegativity property and discriminating capability for tensorial data. Moreover, our proposed method is efficient. The comparisons between SNTFM^2 and the state-of-the-art approaches showed that SNTFM^2 achieved the best performance in both toy and real-word data sets.

Acknowledgments

This work is supported by 973 Program (No. 2010CB327900), NSFC (61070068), NCET (NCET-11-0457) and Australian Research Council Discovery Project with number ARC DP-120103730.

References

Agarwal, A.; Triggs, B.; Rhone-Alpes, I.; and Montbonnot, F. 2010. The fg-net aging database, http://www.fgnet.rsunit.com.

Bottou, L., C. O. D. D., and Weston, J. 2007. Training a support vector machine in the primal. *Neural Computation* 19(5):1155–1178.

Chang, C. C., and Lin, C. J. 2011. LIBSVM: A library for support vector machines, http://www.csie.ntu.edu.tw/ cjlin/libsvm. *ACM Transactions on Intelligent Systems and Technology* 2:27:1–27:27.

Cherkassky, V. 1997. *The Nature Of Statistical Learning Theory*, volume 8.

Das Gupta, M., and Xiao, J. 2011. Non-negative matrix factorization as a feature selection tool for maximum margin classifiers. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 2841–2848.

Fawcett, T. 2006. An introduction to roc analysis. *Pattern Recognition Letters* 27(8):861–874.

Friedlander, M., and Hatz, K. 2008. Computing non-negative tensor factorizations. *Optimisation Methods and Software* 23(4):631–647.

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

Harshman, R., and Lundy, M. 1996. Uniqueness proof for a family of models sharing features of tucker's three-mode factor analysis and parafac/candecomp. *Psychometrika* 61(1):133–154.

Harshman, R.; Hong, S.; and Lundy, M. 2003. Shifted factor analysisłpart i: Models and properties. *Journal of chemometrics* 17(7):363–378.

Hoyer, P. 2004. Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research* 5:1457–1469.

Kiers, H. 2000. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics* 14(3):105–122.

Kim, H., and Park, H. 2007. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics* 23(12):1495–1502.

Kim, J., and Park, H. 2012. Fast nonnegative tensor factorization with an active-set-like method. *High-Performance Scientific Computing* 311–326.

Kimeldorf, G., and Wahba, G. 1970. A correspondence between bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics* 495–502.

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

Kotsia, I., and Patras, I. 2011. Support tucker machines. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, 633–640. IEEE.

Kumar, B. G. V.; Kotsia, I.; and Patras, I. 2012. Max-margin non-negative matrix factorization. *Image Vision Comput.* 279–291.

Liu, J.; Wonka, P.; and Ye, J. 2012. Sparse non-negative tensor factorization using columnwise coordinate descent. *Pattern Recognition* 45(1):649–656.

Martınez-Montes, E.; Valdés-Sosa, P.; Miwakeichi, F.; Goldman, R.; and Cohen, M. 2004. Concurrent eeg/fmri analysis by multiway partial least squares. *NeuroImage* 22(3):1023–1034.

Mørup, M.; Hansen, L.; Parnas, J.; and Arnfred, S. 2006. Decomposing the time-frequency representation of eeg using non-negative matrix and multi-way factorization. *Technical University of Denmark Technical Report*.

Mørup, M.; Hansen, L.; and Arnfred, S. 2008. Algorithms for sparse nonnegative tucker decompositions. *Neural computation* 20(8):2112–2131.

Murakami, T., and Kroonenberg, P. 2003. Three-mode models and individual differences in semantic differential data. *Multivariate Behavioral Research* 38(2):247–283.

Piazza, T.; Lundström, J.; Kunz, A.; and Fjeld, M. 2009. Predicting missing markers in real-time optical motion capture. *Modelling the Physiological Human* 125–136.

Roller, B. 2004. Max-margin markov networks. In *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, volume 16, 25. MIT Press.

Schölkopf, B., and Smola, A. J. 2002. *A Short Introduction to Learning with Kernels*.

Shashua, A., and Hazan, T. 2005. Non-negative tensor factorization with applications to statistics and computer vision. In *Proceedings of the 22nd international conference on Machine learning*, 792–799. ACM.

Tao, D.; Li, X.; Maybank, S.; and Wu, X. 2006. Human carrying status in visual surveillance. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, 1670–1677. IEEE.

Tao, D.; Li, X.; Wu, X.; and Maybank, S. 2007. General tensor discriminant analysis and gabor features for gait recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 29(10):1700–1715.

Wang, Y.; Yunde, J.; Hu, C.; and Turk, M. 2005. Non-negative matrix factorization framework for face recognition. *International Journal of Pattern Recognition and Artificial Intelligence* 19(04):495–511.

Wu, F.; Liu, Y.; and Zhuang, Y. 2009. Tensor-based transductive learning for multimodality video semantic concept detection. *Multimedia, IEEE Transactions on* 11(5):868–878.

Zafeiriou, S. 2009. Discriminant nonnegative tensor factorization algorithms. *Neural Networks, IEEE Transactions on* 20(2):217–235.