A Nonparallel Support Tensor Machine for Binary Classification based Large Margin Distribution and Iterative Optimization

Authors: Du Zhuolin, Song Yisheng, Yisheng Song

Date: 2025-07-17T00:00:00+00:00

Abstract

Based on the tensor-based large margin distribution and the nonparallel support tensor machine, we establish a novel classifier for binary classification problem in this paper, termed the Large Margin Distribution based NonParallel Support Tensor Machine (LDM-NPSTM). The proposed classifier has the following advantages: First, it utilizes tensor data as training samples, which helps to comprehensively preserve the inherent structural information of high-dimensional data, thereby improving classification accuracy. Second, this classifier not only considers traditional empirical risk and structural risk but also incorporates the marginal distribution information of the samples, further enhancing its classification performance. To solve this classifier, we use alternative projection algorithm. Specifically, building on the formulation where in the proposed LDM-NPSTM, the parameters defining the separating hyperplane form a tensor (tensorplane) constrained to be the sum of rank-one tensors, the corresponding optimization problem is solved iteratively using alternative projection algorithm. In each iteration, the parameters related to the projections along a single tensor mode are estimated by solving a typical Support Vector Machine-type optimization problem. Finally, the efficiency and performance of the proposed model and algorithm are verified through theoretical analysis and some numerical examples.

Full Text

Preamble

A Nonparallel Support Tensor Machine for Binary Classification Based on Large Margin Distribution and Iterative Optimization

Zhuolin Du, Yisheng Song
School of Mathematical Sciences, Chongqing Normal University, Chongqing,

401331, P.R. China.

Email: duzhuolin728@163.com (Du); yisheng.song@cqnu.edu.cn (Song)

Abstract

Based on tensor-based large margin distribution and the nonparallel support tensor machine, we establish a novel classifier for binary classification problems in this paper, termed the Large Margin Distribution based NonParallel Support Tensor Machine (LDM-NPSTM). The proposed classifier offers several advantages: First, it utilizes tensor data as training samples, which helps comprehensively preserve the inherent structural information of high-dimensional data, thereby improving classification accuracy. Second, the classifier not only considers traditional empirical risk and structural risk but also incorporates the marginal distribution information of the samples, further enhancing its classification performance. To solve this classifier, we employ an alternative projection algorithm. Specifically, building on the formulation where the parameters defining the separating hyperplane form a tensor (tensorplane) constrained to be the sum of rank-one tensors, the corresponding optimization problem is solved iteratively using the alternative projection algorithm. In each iteration, the parameters related to the projections along a single tensor mode are estimated by solving a typical Support Vector Machine-type optimization problem. Finally, the efficiency and performance of the proposed model and algorithm are verified through theoretical analysis and numerical examples.

Keywords. Nonparallel support tensor machine; margin distribution; CANDECOMP/PARAFAC (CP) decomposition.

AMS subject classifications. 62H30, 15A63, 90C55.

Introduction

A significant number of real-world datasets, especially those involving image data, are frequently represented in tensor format. For instance, a grayscale face image [**?**] can be modeled as a second-order tensor (or matrix), while color images [**?**, **?**], grayscale video sequences [**?**], gait contour sequences [**?**], and hyperspectral cubes [**?**] are typically expressed as third-order tensors. Additionally, color video sequences are often represented as fourth-order tensors [**?**, **?**]. The tensor-based data representation, with its multi-dimensional structure that complicates the capture of spatial and temporal relationships, high dimensionality prone to the curse of dimensionality and overfitting, and substantial storage and computational requirements, uniquely complicates feature extraction and representation, thereby rendering it a fundamental challenge in the design of classifier models.

One of the most representative and successful classification algorithms is the Support Vector Machine (SVM) [**?**, **?**, **?**], which has been successfully applied to a variety of real-world pattern recognition problems, such as text classification

[?, ?], image classification [?, ?], feature extraction [?, ?, ?], web mining [?], and function estimation [?, ?]. The central idea of SVM is to find the optimal separating hyperplane between positive and negative examples. The optimal hyperplane is defined as the one that gives maximum margin between the training examples that are closest to the hyperplane. Different from traditional SVM, in 2007, Jayadeva et al. [?] proposed Twin SVM (TWSVM), which also aims at generating two nonparallel planes such that each plane is closer to one of the two classes and is as far as possible from the other. Notably, the formulation of TWSVMs is very much in line with standard SVMs. However, TWSVMs seek to solve two dual QPPs of smaller size rather than solving a single dual QPP with a large number of parameters in conventional SVM. As a result, the algorithm achieves a processing speed roughly fourfold faster compared to traditional SVM [?]. While the aforementioned classification methods focus on maximizing the minimum margin, research by Gao et al. [?, ?] has indicated that doing so does not necessarily guarantee improved generalization performance. Instead, the distribution of margins has been shown to play a more critical role. Here, the margin distribution is defined by the margin mean and the margin variance. Therefore, in order to improve the generalization performance of SVM, Zhou et al. [?] characterized margin distribution through its mean and variance, leading to the development of the Large Margin Distribution Machine (LDM), which builds upon the SVM framework. The effectiveness of LDM has been proved in theory and experiments.

In recent years, there has been growing interest in extending traditional vector- or matrix-based machine learning algorithms to better handle tensor data [?, ?]. This shift is motivated by the need to effectively process high-dimensional datasets, such as those encountered in image and video analysis. In 2005, Tao et al. [?] proposed a Supervised Tensor Learning (STL) scheme by replacing vector inputs with tensor inputs and decomposing the corresponding weight vector into a rank-1 tensor, which is trained by the alternating projection optimization method. Based on this learning scheme, in 2007, Tao et al. [?] further extended the standard linear SVM to a tensorial format known as the Support Tensor Machine (STM). This adaptation allows for more effective classification of tensor data by leveraging its inherent structure.

Following this development, Zhang et al. [?] generalized the vector-based learning algorithm TWSVM to the tensor-based method Twin STM (TWSTM) and implemented the classifier for microcalcification cluster detection. Compared with TWSVM, the tensor version reduces the overfitting problem significantly. Additionally, Khemchandani et al. developed a least squares variant of STM, termed Proximal STM (PSTM) [?], where the classifier is obtained by solving a system of linear equations rather than a quadratic programming problem at each iteration of the PSTM algorithm as compared to the STM algorithm. This modification enhances computational efficiency while maintaining classification performance. Tensor-based algorithms, on the other hand, decompose the whole problem into several smaller and simpler subproblems, each defined over specific tensor modes and characterized by lower dimensionality. This decomposition

has been shown to reduce the degree of overfitting that appears in vector-based learning techniques, particularly when few training samples are available [**?**].

In this paper, we propose a novel framework, termed Nonparallel Support Tensor Machine based on Large Margin Distribution (LDM-NPSTM), aimed at further enhancing the generalization performance of the Twin Support Tensor Machine (TWSTM). Drawing on the strengths of Large Margin Distribution theory [**?**] and TWSTM [**?**], our approach integrates their core principles to address classification challenges more effectively. Specifically, we characterize the margin distribution using first-order (margin mean) and second-order (margin variance) statistics, with the core objective of maximizing the margin mean while minimizing the margin variance to improve classification robustness. To ensure a more rigorous model structure, we incorporate a regularization term into the LDM-NPSTM framework, balancing empirical risk and structural complexity. For model optimization, we adopt an iterative solution based on CANDECOMP/PARAFAC (CP) decomposition. In each iteration, parameters corresponding to projections along a single tensor mode are estimated by solving a typical SVM-type optimization problem. Notably, the inverse matrix involved in the dual problem is inherently nonsingular, eliminating the need for additional assumptions and simplifying the computational process.

The remainder of this paper is structured as follows: In Section 2, we introduce the notations consistently used throughout the paper and provide a concise overview of fundamental concepts, including those related to SVM, TWSVM, TWSTM, and LDM. Section 3 elaborates on our proposed framework, the LDM-NPSTM, with detailed formulations and a discussion of its key advantages. Experimental results that demonstrate the effectiveness of the LDM-NPSTM are discussed in Section 4. Finally, concluding remarks are given in Section 5.

2 Preliminaries

In this section, we first introduce some notation and basic definitions used throughout the paper, and then briefly review related works.

2.1 Notation and Basic Definitions

An m-th order tensor is defined as a collection of measurements indexed by m indices, with each index corresponding to a mode. Vectors are considered first-order tensors, while matrices represent second-order tensors [**?**].

In this paper, we will utilize lowercase letters (e.g., x) to denote scalars, boldface lowercase letters (e.g., $\mathbf{x}$) and boldface capital letters (e.g., $\mathbf{X}$) to represent vectors and matrices, respectively. Tensors of order 3 or higher will be denoted by boldface Euler script calligraphic letters (e.g., $\mathcal{X}$). Furthermore, we denote the set of all mth-order n-dimensional real tensors as $T_{m,n}$. The i-th element of a vector $\mathbf{x} \in \mathbb{R}^n$ is denoted by x_i, $i = 1, 2, \ldots, n$. In a similar way, the elements of an m-th order tensor $\mathcal{X}$ will be denoted by $x_{i_1 i_2 \ldots i_m}$, where $i_j = 1, 2, \ldots, n_j$

for $j = 1, \ldots, m$. Moreover, we summarize some notations used throughout the paper in Table 1 .

In the following, we introduce some notation and definitions of tensors and matrices in the area of multilinear algebra [**?**, **?**].

Definition 2.1. (Inner product) Given tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$, the inner product of $\mathcal{X}$ and $\mathcal{Y}$ is defined as

$$\langle \mathcal{X}, \mathcal{Y} \rangle := \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_M=1}^{I_M} x_{i_1 i_2 \ldots i_M} y_{i_1 i_2 \ldots i_M}.$$

Definition 2.2. (Frobenius norm) The Frobenius norm of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ is defined as $\|\mathcal{A}\|_F := \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$.

Remark 2.1. Given two same-sized tensors $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ and $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$, the distance between tensors $\mathcal{A}$ and $\mathcal{B}$ is defined as $\|\mathcal{A} - \mathcal{B}\|_F$. Note that the Frobenius norm of the difference between two tensors equals the Euclidean distance of their vectorized representations [**?**].

Definition 2.3. (Outer product) We use $\otimes$ to denote tensor outer product; that is, for any two tensors $\mathcal{A} \in T_{m,n}$ and $\mathcal{B} \in T_{p,n}$ is given by:

$$\mathcal{A} \otimes \mathcal{B} = (a_{i_1 \ldots i_m} b_{i_1 \ldots i_p}) \in T_{m+p,n}.$$

According to this definition, it is easy to check that

$$\mathbf{x} \otimes \cdots \otimes \mathbf{x} = (x_{i_1} \cdots x_{i_k}) \in T_{k,n}.$$

Definition 2.4. (CP decomposition) Given $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$, if there exist $\mathbf{u}_r^{(1)} \in \mathbb{R}^{I_1}, \mathbf{u}_r^{(2)} \in \mathbb{R}^{I_2}, \ldots, \mathbf{u}_r^{(M)} \in \mathbb{R}^{I_M}$ such that

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(M)}$$

where R is a positive integer, we call this a tensor CANDECOMP/PARAFAC (CP) decomposition of $\mathcal{X}$.

Definition 2.5. (Matricization) The matricization (also known as unfolding or flattening of a tensor) is the reordering of the tensor elements into a matrix. The n-mode matricization of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$, denoted by $\mathbf{X}_{(n)} \in \mathbb{R}^{I_n \times (\prod_{k \neq n} I_k)}$, arranges the n-mode fibers to become the columns of the final matrix. Each tensor element $(i_1, i_2, \ldots, i_M)$ maps to the matrix element (i_n, j), where

$$j = 1 + \sum_{k=1, k \neq n}^{M} (i_k - 1) J_k \quad \text{with} \quad J_k = \prod_{l=1, l \neq n}^{k-1} I_l.$$

A more general treatment of matricization can be found in [**?**].

Definition 2.6. (Matrix Kronecker product) The Kronecker product of matrices $\mathbf{A} \in \mathbb{R}^{I\times J}$ and $\mathbf{B} \in \mathbb{R}^{K\times L}$ is denoted by $\mathbf{A} \otimes \mathbf{B}$. The result is a matrix of size $(IK) \times (JL)$ and defined as

$$\mathbf{A} \otimes \mathbf{B} = \begin{bmatrix} a_{11}\mathbf{B} & a_{12}\mathbf{B} & \cdots & a_{1J}\mathbf{B} \\ a_{21}\mathbf{B} & a_{22}\mathbf{B} & \cdots & a_{2J}\mathbf{B} \\ \vdots & \vdots & \ddots & \vdots \\ a_{I1}\mathbf{B} & a_{I2}\mathbf{B} & \cdots & a_{IJ}\mathbf{B} \end{bmatrix}.$$

Definition 2.7. (Matrix Khatri-Rao product) Given matrices $\mathbf{A} \in \mathbb{R}^{I\times K}$ and $\mathbf{B} \in \mathbb{R}^{J\times K}$, their Khatri-Rao product is denoted by $\mathbf{A} \odot \mathbf{B}$. The result is a matrix of size $(IJ) \times K$ defined as

$$\mathbf{A} \odot \mathbf{B} = [\mathbf{a}_1 \otimes \mathbf{b}_1 \quad \mathbf{a}_2 \otimes \mathbf{b}_2 \quad \cdots \quad \mathbf{a}_K \otimes \mathbf{b}_K].$$

Remark 2.2. If matrices $\mathbf{A}$ and $\mathbf{B}$ in Definition 2.7 are vectors, i.e., $\mathbf{a}$ and $\mathbf{b}$, then the Khatri-Rao and Kronecker products are identical, i.e., $\mathbf{a} \otimes \mathbf{b} = \mathbf{a} \odot \mathbf{b}$.

2.2 Related Works

Support Vector Machines (SVMs) form a class of supervised machine learning algorithms that train the classifier function using pre-labeled data. Specifically, for a given training set $\{(\mathbf{x}_i, y_i) \mid i = 1, \cdots, m\}$, where data points $\mathbf{x}_i \in \mathbb{R}^n$ and class labels $y_i \in \{-1, 1\}$, the objective of the support vector machine problem is to identify a hyperplane $\mathbf{w}^\top \mathbf{x} + b = 0$, where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$, in such a way that the two different classes of data points are separated with maximal separation margin and minimal classification loss. The standard SVM problem can be formulated as the following convex quadratic program [**?**]:

$$\begin{aligned} \min_{\mathbf{w},b,\xi_i} \quad & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, 2, \ldots, m. \end{aligned}$$

where ξ_i is a slack variable, and $C > 0$ is a penalty parameter that represents the loss weight.

Moreover, the standard SVM model requires solving a single large-scale optimization problem, which can be computationally intensive. To address this issue, Khemchandani et al. proposed Twin SVM (TWSVM) [**?**]. TWSVM generates two non-parallel planes such that each plane is closer to one of the two classes and is as far as possible from the other. This approach allows TWSVM to solve a pair of smaller-sized quadratic programming problems (QPPs) rather than a single large QPP, resulting in a computational speed that is approximately four times faster than that of traditional SVMs. The optimization

problems in TWSVM can be formulated as the following pair of quadratic programming problems:

$$\begin{aligned} \min_{\mathbf{w}^{(1)},b^{(1)},\mathbf{q}} \quad & \frac{1}{2}(\mathbf{A}\mathbf{w}^{(1)}+\mathbf{e}_1 b^{(1)})^\top(\mathbf{A}\mathbf{w}^{(1)}+\mathbf{e}_1 b^{(1)})+c_1\mathbf{e}^\top\mathbf{q} \\ \text{s.t.} \quad & -(\mathbf{B}\mathbf{w}^{(1)}+\mathbf{e}_2 b^{(1)})+\mathbf{q}\geq \mathbf{e}_2, \quad \mathbf{q}\geq 0 \end{aligned}$$

$$\begin{aligned} \min_{\mathbf{w}^{(2)},b^{(2)},\mathbf{q}} \quad & \frac{1}{2}(\mathbf{B}\mathbf{w}^{(2)}+\mathbf{e}_2 b^{(2)})^\top(\mathbf{B}\mathbf{w}^{(2)}+\mathbf{e}_2 b^{(2)})+c_2\mathbf{e}^\top\mathbf{q} \\ \text{s.t.} \quad & -(\mathbf{A}\mathbf{w}^{(2)}+\mathbf{e}_1 b^{(2)})+\mathbf{q}\geq \mathbf{e}_1, \quad \mathbf{q}\geq 0 \end{aligned}$$

where $c_1, c_2 > 0$ are penalty parameters, $\mathbf{e}_1$ and $\mathbf{e}_2$ are vectors of ones of appropriate dimensions, and $\mathbf{q}$ is the vector of slack variables to deal with linearly nonseparable problems.

Both SVM and TWSVM are vector-based learning algorithms that accept vectors as inputs. In practice, real-world image and video data are more naturally represented as matrices (second-order tensors) or higher-order tensors. Therefore, Zhang et al. [**?**] generalized the vector-based learning algorithm TWSVM to the tensor-based method Twin Support Tensor Machines (TWSTM), which accepts general tensors as input. The following formulation for TWSTM can be established:

$$\begin{aligned} \min_{\mathbf{w}_k^{(1)},b^{(1)},\mathbf{q}} \quad & \frac{1}{2}\left\|\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(1)}+\mathbf{e}_1 b^{(1)}\right\|^\top\left\|\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(1)}+\mathbf{e}_1 b^{(1)}\right\|+c_1\mathbf{e}^\top\mathbf{q} \\ \text{s.t.} \quad & -\left(\mathcal{B}\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(1)}+\mathbf{e}_2 b^{(1)}\right)+\mathbf{q}\geq\mathbf{e}_2, \quad \mathbf{q}\geq 0 \end{aligned}$$

$$\begin{aligned} \min_{\mathbf{w}_k^{(2)},b^{(2)},\mathbf{q}} \quad & \frac{1}{2}\left\|\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(2)}+\mathbf{e}_2 b^{(2)}\right\|^\top\left\|\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(2)}+\mathbf{e}_2 b^{(2)}\right\|+c_2\mathbf{e}^\top\mathbf{q} \\ \text{s.t.} \quad & \mathcal{A}\prod_{k=1}^{M}\times_k\mathbf{w}_k^{(2)}+\mathbf{e}_1 b^{(2)}+\mathbf{q}\geq\mathbf{e}_1, \quad \mathbf{q}\geq 0 \end{aligned}$$

where c_1 and c_2 are penalty parameters, $\mathbf{e}_1$ and $\mathbf{e}_2$ are vectors of appropriate dimensions, and $\mathbf{q}$ is the vector of slack variables to deal with linearly nonseparable problems.

The objective function of the above model is based on minimizing the margin. From the perspective of structural risk, Gao et al. [**?**] have verified that the margin distribution is more important than minimum margins in optimizing generalization performance. By characterizing the margin distribution in terms of margin mean and margin variance, Zhou et al. proposed the Large Margin Distribution Machine (LDM) on the basis of SVM, which focuses on optimizing the distances from the center of the other category. The following formulation

for LDM can be established:

$$\begin{aligned} \min_{\mathbf{w},\xi_i} \quad & \mathbf{w}^\top \mathbf{w} + \lambda_1 \hat{\gamma} - \lambda_2 \bar{\gamma} + C \sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i \mathbf{w}^\top \mathbf{x}_i \geq 1 - \xi_i, \quad \xi_i \geq 0, \end{aligned}$$

where λ_1 and λ_2 are the parameters for trading off the margin variance $\hat{\gamma}$, the margin mean $\bar{\gamma}$, and the model complexity.

3 Proposed LMD-NPSTM

In this section, LDM-NPSTM is proposed. Specifically, Subsection 3.1 introduces the structure of the model, optimization strategies, and associated dual problems. Moreover, the detailed implementation algorithm of LDM-NPSTM will be shown in Subsection 3.2.

3.1 Model Construction and Optimization

Consider the binary classification problem in tensor space, where the training set is defined as $T_m = \{(\mathcal{X}_p, y_p) \mid p = 1, 2, \ldots, m_1\} \cup \{(\mathcal{Y}_q, y_q) \mid q = 1, 2, \ldots, m_2\}$. Here, $\mathcal{X}_p, \mathcal{Y}_q \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$ denote the feature tensor of the p-th and q-th sample, respectively, $y_p = 1$ and $y_q = -1$ are their corresponding class labels; and $m = m_1 + m_2$ (where m_1 and m_2 are the numbers of positive and negative samples in T_m, respectively). Let $\mathbf{y}_1 = [1, \cdots, 1]^\top \in \mathbb{R}^{m_1 \times 1}$ be the label vector for all positive samples, where each element corresponds to the label of a positive instance, and $\mathbf{y}_2 = [-1, \cdots, -1]^\top \in \mathbb{R}^{m_2 \times 1}$ denote the label vector for all negative samples, with each element corresponding to the label of a negative instance. The LDM-NPSTM identifies two non-parallel hyperplanes in the feature space:

$$f_1(\mathcal{X}) = \langle \mathcal{W}_1, \mathcal{X} \rangle = 0, \quad f_2(\mathcal{Y}) = \langle \mathcal{W}_2, \mathcal{Y} \rangle = 0,$$

where $\mathcal{W}_1, \mathcal{W}_2 \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_M}$.

Notably, inspired by the work of Zhou et al. [**?**], the bias term does not affect the overall derivation process. Furthermore, unlike the conventional approach that calculates margin mean and variance using the entire dataset, the LDM-NPSTM separates these calculations into positive-class and negative-class components [**?**]. Specifically, these metrics are associated with the distances to their respective hyperplanes, as elaborated below.

The distance from an individual data point to the hyperplane is defined by

$$\gamma_p^+ = y_p \frac{|\langle \mathcal{W}_2, \mathcal{X}_p \rangle|}{\|\mathcal{W}_2\|_F}, \quad p = 1, \ldots, m_1,$$

$$\gamma_q^- = y_q \frac{|\langle \mathcal{W}_1, \mathcal{Y}_q \rangle|}{\|\mathcal{W}_1\|_F}, \quad q = 1, \ldots, m_2,$$

where $y_p \in \{1, -1\}$ and $y_q \in \{1, -1\}$ denote the labels of positive/negative samples, respectively.

The margin means are defined as follows:

$$\bar{\gamma}^+ = \frac{1}{m_1}\sum_{p=1}^{m_1}\gamma_p^+ = \frac{1}{m_1}\sum_{p=1}^{m_1}\frac{|\langle \mathcal{W}_2, \mathcal{X}_p\rangle|}{\|\mathcal{W}_2\|_F},$$

$$\bar{\gamma}^- = \frac{1}{m_2}\sum_{q=1}^{m_2}\gamma_q^- = \frac{1}{m_2}\sum_{q=1}^{m_2}\frac{|\langle \mathcal{W}_1, \mathcal{Y}_q\rangle|}{\|\mathcal{W}_1\|_F}.$$

The margin variances are defined as follows:

$$\hat{\gamma}^+ = \frac{1}{m_1 - 1}\sum_{p=1}^{m_1}\frac{\langle \mathcal{W}_2, \mathcal{X}_p\rangle^2}{\|\mathcal{W}_2\|_F^2}, \quad \hat{\gamma}^- = \frac{1}{m_2 - 1}\sum_{q=1}^{m_2}\frac{\langle \mathcal{W}_1, \mathcal{Y}_q\rangle^2}{\|\mathcal{W}_1\|_F^2}.$$

To effectively capture complex data relationships and enhance classification performance, we assume the weights in subsequent classifiers form a tensor $\mathcal{W}$. This tensor can be decomposed into a sum of R rank-one tensors, as defined by the CP decomposition:

$$\mathcal{W} = \sum_{r=1}^{R}\mathbf{u}_r^{(1)} \otimes \mathbf{u}_r^{(2)} \otimes \cdots \otimes \mathbf{u}_r^{(M)}$$

where $\mathbf{u}_r^{(j)} \in \mathbb{R}^{I_j}$, $j = 1, 2, \ldots, M$, and M is the number of tensor modes. The weight tensor $\mathcal{W}$ generalizes the weight vector $\mathbf{w}$ in SVMs.

For mode j ($j = 1, \ldots, M$), we stack the rank-one components $\{\mathbf{u}_r^{(j)}\}_{r=1}^R$ into a matrix:

$$\mathbf{U}^{(j)} = [\mathbf{u}_1^{(j)}, \mathbf{u}_2^{(j)}, \ldots, \mathbf{u}_R^{(j)}] \in \mathbb{R}^{I_j \times R}.$$

The j-th matricization of $\mathcal{W}$ is given by:

$$\mathbf{W}_{(j)} = \mathbf{U}^{(j)}\left(\mathbf{U}^{(M)} \odot \cdots \odot \mathbf{U}^{(j+1)} \odot \mathbf{U}^{(j-1)} \odot \cdots \odot \mathbf{U}^{(1)}\right)^\top = \mathbf{U}^{(j)}\left(\mathbf{U}^{(-j)}\right)^\top$$

Similarly, the j-th matricization of samples $\mathcal{X}$ is given by $\mathbf{X}_{(j)} = \mathbf{V}^{(j)}\left(\mathbf{V}^{(-j)}\right)^\top$.

Further, consider the tensor inner product property:

$$\langle \mathcal{W}, \mathcal{W}\rangle = \mathrm{Tr}\left(\mathbf{W}_{(j)}\mathbf{W}_{(j)}^\top\right) = \mathrm{vec}\left(\mathbf{W}_{(j)}\right)^\top \mathrm{vec}\left(\mathbf{W}_{(j)}\right).$$

Analogous to the above, for individual sample tensors $\mathcal{X}_p$ and $\mathcal{Y}_q$, their inner products with weight tensors $\mathcal{W}_1, \mathcal{W}_2$ satisfy:

$$\langle \mathcal{W}_1, \mathcal{X}_p\rangle = \mathrm{Tr}\left(\mathbf{W}_{1(j)}^\top \mathbf{X}_{p(j)}\right), \quad \langle \mathcal{W}_2, \mathcal{X}_p\rangle = \mathrm{Tr}\left(\mathbf{W}_{2(j)}^\top \mathbf{X}_{p(j)}\right),$$

$$\langle \mathcal{W}_1, \mathcal{Y}_q\rangle = \mathrm{Tr}\left(\mathbf{W}_{1(j)}^\top \mathbf{Y}_{q(j)}\right), \quad \langle \mathcal{W}_2, \mathcal{Y}_q\rangle = \mathrm{Tr}\left(\mathbf{W}_{2(j)}^\top \mathbf{Y}_{q(j)}\right).$$

To guarantee that the matrices in the LDM-NPSTM dual problem are nonsingular, we add a regularization term $\|\mathcal{W}_i\|_F^2$, $i = 1, 2$.

LMD-NPSTM seeks a pair of tensors $\mathcal{W}_1$ and $\mathcal{W}_2$ simultaneously maximizing the mean of positive and negative margin while minimizing the variance of margin:

$$\begin{aligned} \min_{\mathcal{W}_1,\xi_2} \quad & \frac{1}{2}\sum_{p=1}^{m_1}\langle \mathcal{W}_1, \mathcal{X}_p\rangle^2 + \|\mathcal{W}_1\|_F^2 + \lambda_1\hat{\gamma}^- - \lambda_3\bar{\gamma}^- + c_3\mathbf{e}^\top\xi_2 \\ \text{s.t.} \quad & -\sum_{q=1}^{m_2}\langle \mathcal{W}_1, \mathcal{Y}_q\rangle + \xi_q^2 \geq 1, \quad \xi_q^2 \geq 0, \quad q = 1, \dots, m_2, \end{aligned}$$

$$\begin{aligned} \min_{\mathcal{W}_2,\xi_1} \quad & \frac{1}{2}\sum_{q=1}^{m_2}\langle \mathcal{W}_2, \mathcal{Y}_q\rangle^2 + \|\mathcal{W}_2\|_F^2 + \lambda_2\hat{\gamma}^+ - \lambda_4\bar{\gamma}^+ + c_4\mathbf{e}^\top\xi_1 \\ \text{s.t.} \quad & -\sum_{p=1}^{m_1}\langle \mathcal{W}_2, \mathcal{X}_p\rangle + \xi_p^1 \geq 1, \quad \xi_p^1 \geq 0, \quad p = 1, \dots, m_1, \end{aligned}$$

where $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are hyperparameters; c_3 and c_4 weight the penalty on slack variables ξ_1 and ξ_2.

For model optimization, we adopt an alternating optimization scheme. At the iterations for the j-th mode we solve:

$$\begin{aligned} \min_{\mathbf{W}_{(j)}^1,\xi_2} \quad & \frac{1}{2}\sum_{p=1}^{m_1}\left(\mathrm{Tr}\left(\mathbf{W}_{(j)}^{1\top}\mathbf{X}_{p(j)}\right)\right)^2 + \mathrm{Tr}\left(\mathbf{W}_{(j)}^{1\top}\mathbf{W}_{(j)}^1\right) + \lambda_1\hat{\gamma}^- - \lambda_3\bar{\gamma}^- + c_3\mathbf{e}^\top\xi_2 \\ \text{s.t.} \quad & -\sum_{q=1}^{m_2}\mathrm{Tr}\left(\mathbf{W}_{(j)}^{1\top}\mathbf{Y}_{q(j)}\right) + \xi_q^2 \geq 1, \quad \xi_q^2 \geq 0, \quad q = 1, \dots, m_2, \end{aligned}$$

$$\begin{aligned} \min_{\mathbf{W}_{(j)}^2,\xi_1} \quad & \frac{1}{2}\sum_{q=1}^{m_2}\left(\mathrm{Tr}\left(\mathbf{W}_{(j)}^{2\top}\mathbf{Y}_{q(j)}\right)\right)^2 + \mathrm{Tr}\left(\mathbf{W}_{(j)}^{2\top}\mathbf{W}_{(j)}^2\right) + \lambda_2\hat{\gamma}^- - \lambda_4\bar{\gamma}^- + c_4\mathbf{e}^\top\xi_1 \\ \text{s.t.} \quad & -\sum_{p=1}^{m_1}\mathrm{Tr}\left(\mathbf{W}_{(j)}^{2\top}\mathbf{X}_{p(j)}\right) + \xi_p^1 \geq 1, \quad \xi_p^1 \geq 0, \quad p = 1, \dots, m_1. \end{aligned}$$

Under the CP constraint, we replace $\mathbf{W}_{(j)}$ using $\mathbf{U}^{(j)}(\mathbf{U}^{(-j)})^\top$. To simplify, define $\mathbf{A} = \mathbf{U}^{(-j)\top}\mathbf{U}^{(-j)}$ and $\tilde{\mathbf{U}}^1 = \mathbf{U}_{(j)}^1\mathbf{A}^{1/2}$. Define transformed data matrices:

$$\tilde{\mathbf{X}}_p = \mathbf{X}_{p(j)}\mathbf{U}^{(-j)}\mathbf{A}^{-1/2}, \quad \tilde{\mathbf{Y}}_q = \mathbf{Y}_{q(j)}\mathbf{U}^{(-j)}\mathbf{A}^{-1/2}.$$

The subproblem for $\tilde{\mathbf{U}}^1$ becomes:

$$\begin{aligned}\min_{\mathrm{vec}(\tilde{\mathbf{U}}^1),\xi_2} \quad & \frac{1}{2}\sum_{p=1}^{m_1}\left(\mathrm{vec}(\tilde{\mathbf{U}}^1)^\top \mathrm{vec}(\tilde{\mathbf{X}}_p)\right)^2 + \mathrm{vec}(\tilde{\mathbf{U}}^1)^\top \mathrm{vec}(\tilde{\mathbf{U}}^1) + \frac{\lambda_1}{m_2-1}\sum_{q=1}^{m_2}\left(\mathrm{vec}(\tilde{\mathbf{U}}^1)^\top \mathrm{vec}(\tilde{\mathbf{Y}}_q)\right)^2 \\ & - \frac{\lambda_3}{m_2}\sum_{q=1}^{m_2}\mathrm{vec}(\tilde{\mathbf{U}}^1)^\top \mathrm{vec}(\tilde{\mathbf{Y}}_q) + c_3\mathbf{e}^\top\xi_2 \\ \text{s.t.} \quad & -\mathrm{vec}(\tilde{\mathbf{U}}^1)^\top \mathrm{vec}(\tilde{\mathbf{Y}}_q) + \xi_q^2 \geq 1, \quad \xi_q^2 \geq 0, \quad q = 1, \ldots, m_2.\end{aligned}$$

Similarly for $\tilde{\mathbf{U}}^2$ with $\mathbf{B} = \mathbf{U}^{(-j)\top}\mathbf{U}^{(-j)}$ and $\bar{\mathbf{Y}}_q^{(j)}, \bar{\mathbf{X}}_p^{(j)}$:

$$\begin{aligned}\min_{\mathrm{vec}(\tilde{\mathbf{U}}^2),\xi_1} \quad & \frac{1}{2}\sum_{q=1}^{m_2}\left(\mathrm{vec}(\tilde{\mathbf{U}}^2)^\top \mathrm{vec}(\bar{\mathbf{Y}}_q^{(j)})\right)^2 + \mathrm{vec}(\tilde{\mathbf{U}}^2)^\top \mathrm{vec}(\tilde{\mathbf{U}}^2) + \frac{\lambda_2}{m_1-1}\sum_{p=1}^{m_1}\left(\mathrm{vec}(\tilde{\mathbf{U}}^2)^\top \mathrm{vec}(\bar{\mathbf{X}}_p^{(j)})\right)^2 \\ & - \frac{\lambda_4}{m_1}\sum_{p=1}^{m_1}\mathrm{vec}(\tilde{\mathbf{U}}^2)^\top \mathrm{vec}(\bar{\mathbf{X}}_p^{(j)}) + c_4\mathbf{e}^\top\xi_1 \\ \text{s.t.} \quad & -\mathrm{vec}(\tilde{\mathbf{U}}^2)^\top \mathrm{vec}(\bar{\mathbf{X}}_p^{(j)}) + \xi_p^1 \geq 1, \quad \xi_p^1 \geq 0, \quad p = 1, \ldots, m_1.\end{aligned}$$

Theorem 3.1. The optimal solution $\mathrm{vec}(\tilde{\mathbf{U}}^1)^*$ and $\mathrm{vec}(\tilde{\mathbf{U}}^2)^*$ can be expressed as $\mathrm{vec}(\tilde{\mathbf{U}}^1)^* = \mathbf{V}^{(j)}\beta_1$ and $\mathrm{vec}(\tilde{\mathbf{U}}^2)^* = \mathbf{V}^{(j)}\beta_2$, where $\beta_1, \beta_2 \in \mathbb{R}^m$ are coefficient vectors.

Theorem 3.2. The original problem can be transformed into the Wolfe dual problem:

$$\begin{aligned}\max_{\alpha_1} \quad & -\frac{1}{2}\alpha_1^\top\mathbf{H}_1\alpha_1 + \left(\frac{\lambda_3}{m_2}\mathbf{H}_1\mathbf{y}_2 + \mathbf{e}_2\right)^\top\alpha_1 \\ \text{s.t.} \quad & \mathbf{0} \leq \alpha_1 \leq c_3\mathbf{e}_2,\end{aligned}$$

where $\mathbf{H}_1 = \mathbf{M}_1\mathbf{G}_1^{-1}\mathbf{M}_1^\top$, $\mathbf{G}_1 = \mathbf{K}_1^\top\mathbf{K}_1 + c_1\mathbf{K} + \frac{2\lambda_1}{m_2-1}\mathbf{M}_1^\top\mathbf{M}_1$, and $\beta_1 = \mathbf{G}_1^{-1}(\frac{\lambda_3}{m_2}\mathbf{M}_1^\top\mathbf{y}_2 - \mathbf{M}_1^\top\alpha_1)$.

Similarly for α_2:

$$\begin{aligned}\max_{\alpha_2} \quad & -\frac{1}{2}\alpha_2^\top\mathbf{H}_2\alpha_2 + \left(\frac{\lambda_4}{m_1}\mathbf{H}_2\mathbf{y}_1 + \mathbf{e}_1\right)^\top\alpha_2 \\ \text{s.t.} \quad & \mathbf{0} \leq \alpha_2 \leq c_4\mathbf{e}_1,\end{aligned}$$

where $\mathbf{H}_2 = \mathbf{M}_2\mathbf{G}_2^{-1}\mathbf{M}_2^\top$, $\mathbf{G}_2 = \mathbf{K}_2^\top\mathbf{K}_2 + c_2\mathbf{K} + \frac{2\lambda_2}{m_1-1}\mathbf{M}_2^\top\mathbf{M}_2$, and $\beta_2 = \mathbf{G}_2^{-1}(\frac{\lambda_4}{m_1}\mathbf{M}_2^\top\mathbf{y}_1 - \mathbf{M}_2^\top\alpha_2)$.

Theorem 3.3. The expression for the decision function of LDM-NPSTM is:

$$f(\mathcal{X}) = \arg\min_{i=1,2}\frac{|\langle\mathcal{W}_i, \mathcal{X}\rangle|}{\|\mathcal{W}_i\|_F} = \arg\min_{i=1,2}\frac{|\mathbf{K}_i\beta_i|}{\sqrt{\beta_i^\top\mathbf{K}\beta_i}}.$$

3.2 Algorithm and Pseudocode

Algorithm 1 Alternating Projection for LDM-NPSTM

Input: Training tensors $\{\mathcal{X}_p\}$, $\{\mathcal{Y}_q\}$, labels y_i, $N = 5000$. **Output:** Parameters $\mathcal{W}_1$ and $\mathcal{W}_2$.

1. Initialize $\mathcal{W}_1, \mathcal{W}_2$ as sum of random rank-one tensors, $k = 0$.
2. While $\|W_i^{(k)} - W_i^{(k-1)}\| / \|W_i^{(k-1)}\| > \epsilon$ or $k < N$ do:
   - For $j = 1, \ldots, M$ do:
     - Update α_1, α_2 using QPP toolkit.
     - Update β_1, β_2.
     - Update factor matrices $\mathbf{U}_{(j)}^1, \mathbf{U}_{(j)}^2$.
   - End for
   - Update $\mathbf{W}_{(j)}^{1(k)}, \mathbf{W}_{(j)}^{2(k)}$.
   - $k \leftarrow k + 1$.
3. End while

Theorem 3.4. The sequences $\{f_i(\mathbf{U}_i^{(j)}, \xi_{-i}^{(j)})\}$ generated in Algorithm 1 are monotonically non-increasing and converge to limit points.

Note: Figure translations are in progress. See original paper for figures.