Knowl Inf Syst (2006)
DOI 10.1007/s10115-006-0050-6

Knowledge and Information Systems

REGULAR PAPER

Dacheng Tao · Xuelong Li · Xindong Wu · Weiming Hu · Stephen J. Maybank

Supervised tensor learning

Received: 30 November 2005 / Revised: 10 July 2006 / Accepted: 18 August 2006

Abstract Tensor representation is helpful to reduce the small sample size problem in discriminative subspace selection. As pointed by this paper, this is mainly because the structure information of objects in computer vision research is a reasonable constraint to reduce the number of unknown parameters used to represent a learning model. Therefore, we apply this information to the vector-based learning and generalize the vector-based learning to the tensor-based learning as the supervised tensor learning (STL) framework, which accepts tensors as input. To obtain the solution of STL, the alternating projection optimization procedure is developed. The STL framework is a combination of the convex optimization and the operations in multilinear algebra. The tensor representation helps reduce the overfitting problem in vector-based learning. Based on STL and its alternating projection optimization procedure, we generalize support vector machines, minimax probability machine, Fisher discriminant analysis, and distance metric learning, to support tensor machines, tensor minimax probability machine, tensor Fisher discriminant analysis, and the multiple distance metrics learning, respectively. We also study the iterative procedure for feature extraction within STL. To examine the effectiveness of STL, we implement the tensor minimax probability machine for image classification. By comparing with minimax probability machine, the tensor version reduces the overfitting problem.

D. Tao (✉) · X. Li · S. J. Maybank
School of Computer Science and Information Systems, Birkbeck, University of London, London, UK
E-mail: dacheng.tao@gmail.com

X. Wu
Department of Computer Science, University of Vermont, Burlington, VT, USA

W. Hu
National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, P.R. China

Keywords Convex optimization · Supervised learning · Tensor · Alternating projection

1 Introduction

In computer vision research, many objects are naturally represented by multidimensional arrays, i.e., tensors [17], such as the gray face image shown in Fig. 1 in face recognition [7, 48], the color image shown in Fig. 2 in scene image classification [40, 41], and the video shot shown in Fig. 3 in motion categorization [11, 26]. However, in current research, the original tensors (images and videos) are always scanned into vectors, thus discarding a great deal of useful structural information [36, 38, 52, 53], which is helpful to reduce the small sample-size (SSS) problem in subspace selection methods, e.g., linear discriminant analysis (LDA).

To utilize this structure information, many dimension reduction algorithms [17, 29, 38, 46, 51, 52] based on the multilinear subspace method (MLSM) have been developed for data representation [17, 29, 46, 51], pattern classification [38, 36, 52], and network abnormal detection [33]. This structure information of objects in computer vision research is a reasonable constraint to reduce the number of unknown parameters used to represent a learning model. MLSM finds a sequence of linear transformation matrices $U_i \in R^{L_i \times L_i'}$ ($L_i' < L_i$, $1 \leq i \leq M$) to transform a big-size tensor $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ to a small-size

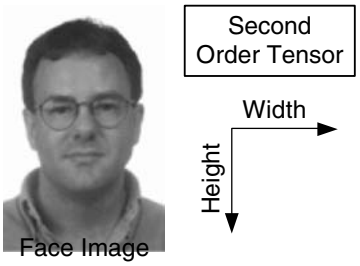


Fig. 1 A gray face image is a second-order tensor, which is also a matrix. Two indices are required for pixel locations. The face image comes from http://www.merl.com/projects/images/face-rec.gif

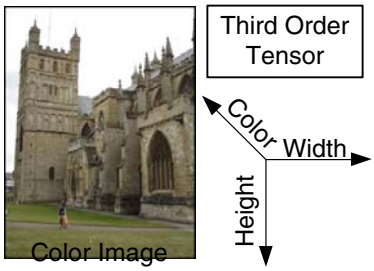


Fig. 2 A color image is a third-order tensor, which is also a data cuboid, because three indices are required to locate elements. Two indices are used for pixel locations and one index is used to local the color information (e.g., R, G, and B)

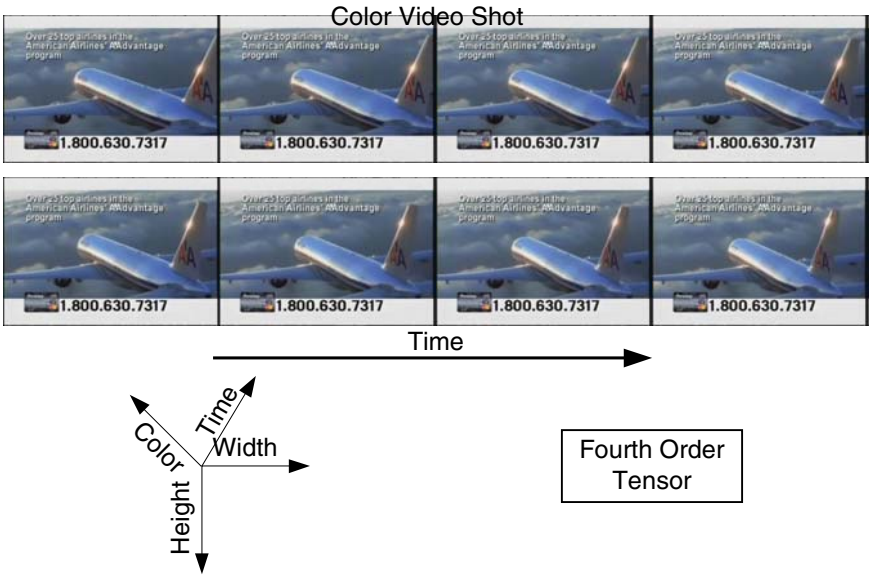


Fig. 3 A color video shot is a fourth-order tensor. Four indices are used to locate elements. Two indices are used for pixel locations; one index is used to local the color information; and the other index is used to represent the time varying. The video shot comes from http://www-nlpir.nist.gov/projects/trecvid/

tensor $\mathbf{Y} \in R^{L_1' \times L_2' \times \cdots \times L_M'}$, i.e., $\mathbf{Y} = \mathbf{X} \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times \cdots \times_M U_M^{\mathrm{T}}$. For example, if we have a big second order tensor (i.e., a big matrix) $\mathbf{X} \in R^{L_1 \times L_2}$, in MLSM we need to find two linear transformation matrices $U_1 \in R^{L_1 \times L_1'}$ ($L_1' < L_1$) and $U_2 \in R^{L_2 \times L_2'}$ ($L_2' < L_2$) to transform the big matrix to a small matrix according to $\mathbf{Y} = \mathbf{X} \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}}$, i.e., $\mathbf{Y} = U_1^{\mathrm{T}} \mathbf{X} U_2$. After the transformation, the original datum dimension is reduced from $L_1 \times L_2$ to $L_1' \times L_2'$, i.e., $\mathbf{Y} \in R^{L_1' \times L_2'}$.

The structure information can also be utilized to vector-based learning to reduce the overfitting problem when measurements are limited. In vector-based learning [6, 9], a projection vector $\vec{w} \in R^L$ and a bias $b \in R$ are learnt to determine the class label of a measurement $\vec{x} \in R^L$ according to a linear decision function $y(\vec{x}) = \mathrm{sign}[\vec{w}^{\mathrm{T}} \vec{x} + b]$. The $\vec{w}$ and b are obtained based on a learning model, e.g., minimax probability machine (MPM) [16, 31], based on N training measurements associated with labels $\{\vec{x}_i \in R^L, y_i\}$, where y_i is the class label, $y_i \in \{+1, -1\}$, and $1 \leq i \leq N$.

The supervised tensor learning (STL) [36] is developed to extend the vector-based learning algorithms to accept tensors as input. That is, we learn a series of projection vectors $\vec{w}_k|_{k=1}^M \in R^{L_k}$ and a bias $b \in R$ to determine the class label $\{+1, -1\}$ of a measurement $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ according to a multilinear decision function $y(\mathbf{X}) = \mathrm{sign}[\mathbf{X} \prod_{k=1}^M \times_k \vec{w}_k + b]$. The projection vectors $\vec{w}_k|_{k=1}^M$ and the bias b are obtained from a learning model, e.g., tensor minimax probability machine (TMPM), based on N training measurements associated with labels $\{\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}, y_i\}$, where y_i is the class label, $y_i \in \{+1, -1\}$, and

$1 \leq i \leq N$. To obtain the solution of the algorithms under STL framework, we develop the alternating projection optimization procedure. Based on STL and its alternating projection optimization procedure, we illustrate several examples, which are support tensor machines (STMs), tensor minimax probability machine (TMPM), tensor Fisher discriminant analysis (TFDA), multiple distance metrics learning (MDML).

This paper is organized as follows. Section 2 introduces tensor algebra. Section 3 gives the relatiobship between LSM and MLSM. In Section 4, the convex optimization is briefly reviewed and a framework-level formula of the convex optimization-based learning is introduced. In Section 5, we develop the supervised tensor learning (STL) framewok, which is an extension of the convex optimization-based learning. The alternating projection method is also developed to obtain the solution to an STL-based learning algorithm. In Section 6, we develop a number of tensor extensions of many popular learning machines, such as the support vector machines (SVM) [5, 27, 28, 34, 35, 45], the minimax probability machine (MPM) [16, 31], the Fisher discriminant analysis (FDA) [6, 8, 14], and the distance metric learning (DML) [49]. In Section 7, an iterative feature extraction model is given as an extension of the STL framework. Experiments in Section 8 based on TMPM show that tensor representation is helpful to reduce the overfitting problem in vector-based learning. Section 9 provides conclusions.

2 Tensor algebra

This section contains the fundamental materials on tensor algebra [17], which are relevant to this paper. Tensors are arrays of numbers that transform in certain ways under different coordinate transformations. The order of a tensor $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$, represented by a multidimensional array of real numbers, is M. An element of $\mathbf{X}$ is denoted as $\mathbf{X}_{l_1, l_2, \ldots, l_M}$, where $1 \leq l_i \leq L_i$ and $1 \leq i \leq M$. The i^{th} dimension (or mode) of $\mathbf{X}$ is of size L_i. A scalar is a zeroth-order tensor; a vector is a first-order tensor; and a matrix is a second-order tensor. A third-order tensor as an example is shown in Fig. 4. In the tensor terminology, we have the following definitions.

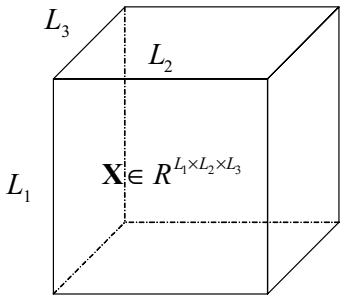


Fig. 4 A third-order tensor $\mathbf{X} \in R^{L_1 \times L_2 \times L_3}$

Definition 2.1 *(Tensor Product or Outer Product) The tensor product* $\mathbf{X} \otimes \mathbf{Y}$ *of a tensor* $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ *and another tensor* $\mathbf{Y} \in R^{L'_1 \times L'_2 \times \cdots \times L'_{M'}}$ *is defined by*

$$(\mathbf{X} \otimes \mathbf{Y})_{l_1 \times l_2 \times \cdots \times l_M \times l'_1 \times l'_2 \times \cdots \times l'_{M'}} = \mathbf{X}_{l_1 \times l_2 \times \cdots \times l_M} \mathbf{Y}_{l'_1 \times l'_2 \times \cdots \times l'_{M'}} \tag{1}$$

for all index values.

For example, the tensor product of two vectors $\vec{x}_1 \in R^{L_1}$ *and* $\vec{x}_2 \in R^{L_2}$ *is a matrix* $X \in R^{L_1 \times L_2}$*, i.e.,* $X = \vec{x}_1 \otimes \vec{x}_2 = \vec{x}_1 \vec{x}_2^{\mathrm{T}}$.

Definition 2.2 *(Mode-d Matricizing or Matrix Unfolding) The mode-d matricizing or matrix unfolding of an* Mth*-order tensor* $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ *is the set of vectors in* R^{L_d} *obtained by keeping the index* i_d *fixed and varying the other indices. Therefore, the mode-d matricizing or matrix unfolding of an* Mth*-order tensor is a matrix* $X_{(d)} \in R^{L_d \times \bar{L}_d}$*, where* $\bar{L}_d = (\prod_{i \neq d} L_i)$*. We denote the mode-d matricizing of* $\mathbf{X}$ *as* $\mathrm{mat}_d(\mathbf{X})$ *or briefly* $X_{(d)}$.

Definition 2.3 *(Tensor Contraction) The contraction of a tensor is obtained by equating two indices and summing over all values of the repeated indices. Contraction reduces the tensor order by 2. A notation is the Einstein's summation convention.*[1] *For example, the tensor product of two vectors* $\vec{x}, \vec{y} \in R^N$ *is* $Z = \vec{x} \otimes \vec{y}$*; and the contraction of* Z *is* $Z_{ii} = \vec{x} \cdot \vec{y} = \vec{x}^{\mathrm{T}} \vec{y}$*, where the repeated indices imply summation. The value of* Z_{ii} *is the inner product of* $\vec{x}$ *and* $\vec{y}$*. In general, for tensors* $\mathbf{X} \in R^{L_1 \times \cdots \times L_M \times L'_1 \times \cdots \times L'_{M'}}$ *and* $\mathbf{Y} \in R^{L_1 \times \cdots \times L_M \times L''_1 \times \cdots \times L''_{M''}}$*, the contraction on the tensor product* $\mathbf{X} \otimes \mathbf{Y}$ *is*

$$\begin{aligned} &[\![\mathbf{X} \otimes \mathbf{Y}; (1:M)\,(1:M)]\!] \\ &\quad = \sum_{l_1=1}^{L_1} \cdots \sum_{l_M=1}^{L_M} (\mathbf{X})_{l_1 \times \cdots \times l_M \times l'_1 \times \cdots \times l'_{M'}} (\mathbf{Y})_{l_1 \times \cdots \times l_M \times l''_1 \times \cdots \times l''_{M''}}. \end{aligned} \tag{2}$$

In this paper, when the convention is conducted on all indices but the index i *on the tensor product of* $\mathbf{X}$ *and* $\mathbf{Y}$ *in* $R^{L_1 \times L_2 \times \cdots \times L_M}$*, we denote this procedure as*

$$\begin{aligned} [\![\mathbf{X} \otimes \mathbf{Y}; \bar{i}\,\bar{i}]\!] &= [\![\mathbf{X} \otimes \mathbf{Y}; (1:i-1, i+1:M)\,(1:i-1, i+1:M)]\!] \\ &= \sum_{l_1=1}^{L_1} \cdots \sum_{l_{i-1}=1}^{L_{i-1}} \sum_{l_{i+1}=1}^{L_{i+1}} \cdots \sum_{l_M=1}^{L_M} \\ &\quad \times (\mathbf{X})_{l_1 \times \cdots \times l_{i-1} \times l_i \times l_{i+1} \times \cdots \times l_M} (\mathbf{Y})_{l_1 \times \cdots \times l_{i-1} \times l_i \times l_{i+1} \times \cdots \times l_M} \\ &= \mathrm{mat}_i(\mathbf{X})\, \mathrm{mat}_i^{\mathrm{T}}(\mathbf{Y}) = X_{(i)} Y_{(i)}^{\mathrm{T}}, \end{aligned} \tag{3}$$

and $[\![\mathbf{X} \otimes \mathbf{Y}; \bar{i}\,\bar{i}]\!] \in R^{L_i \times L_i}$.

[1] *When any two subscripts in a tensor expression are given the same symbol, it is implied that the convention is formed.*–A. Einstein, Die Grundlage der Allgemeinen Relativitatstheorie, Ann. Phys., 49:769, 1916.

Definition 2.4 *(Mode-d product) The mode-d product* $\mathbf{X} \times_d U$ *of a tensor* $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ *and a matrix* $U \in R^{L'_d \times L_d}$ *is an* $L_1 \times L_2 \times \cdots \times L_{d-1} \times L'_d \times L_{d+1} \times \cdots \times L_M$ *tensor defined by*

$$(\mathbf{X} \times_d U)_{l_1 \times l_2 \times \cdots \times l_{d-1} \times l'_d \times l_{d+1} \times \cdots \times l_M} = \sum_{l'_d} \left(\mathbf{X}_{l_1 \times l_2 \times \cdots \times l_{d-1} \times l_d \times l_{d+1} \times \cdots \times l_M} U_{l'_d \times l_d} \right), \tag{4}$$

for all index values. The mode-d product is a type of contraction.

Based on the definition of Mode-d product, we have

$$(\mathbf{X} \times_d U) \times_t V = (\mathbf{X} \times_t V) \times_d U, \tag{5}$$

where $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$, $U \in R^{L'_d \times L_d}$, *and* $V \in R^{L'_t \times L_t}$. *Therefore,* $(\mathbf{X} \times_d U) \times_t V$ *can be simplified as* $\mathbf{X} \times_d U \times_t V$.

Furthermore,

$$(\mathbf{X} \times_d U) \times_t V = \mathbf{X} \times_d (VU), \tag{6}$$

where $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$, $U \in R^{L'_d \times L_d}$, $V \in R^{L''_d \times L'_d}$, *and* VU *is the standard matrix product between* V *and* U.

To simplify the notation in this paper, we denote

$$\mathbf{X} \times_1 U_1 \times_2 U_2 \times \cdots \times_M U_M \triangleq \mathbf{X} \prod_{k=1}^{M} \times_k U_k, \tag{7}$$

and

$$\mathbf{X} \times_1 U_1 \times \cdots \times_{i-1} U_{i-1} \times_{i+1} U_{i+1} \times \cdots \times_M U_M = \mathbf{X} \prod_{d=1; d \neq i}^{M} \times_d U_d \triangleq \mathbf{X} \bar{\times}_i U_i. \tag{8}$$

Definition 2.5 *(Frobenius Norm) The Frobenius norm of a tensor* $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ *is given by*

$$\|\mathbf{X}\|_{\text{Fro}} = \sqrt{[\![\mathbf{X} \otimes \mathbf{X}; (1:M)(1:M)]\!]} = \sqrt{\sum_{l_1=1}^{L_1} \cdots \sum_{l_M=1}^{L_M} \mathbf{X}^2_{l_1 \times \cdots \times l_M}}. \tag{9}$$

The Frobenius norm of a tensor $\mathbf{X}$ *measures the size of the tensor and its square is the energy of the tensor.*

Definition 2.6 *(Rank-1 tensor) An* M*th-order tensor* $\mathbf{X}$ *has rank one if it is the tensor product of* M *vectors* $\vec{u}_i \in R^{L_i}$, *where* $1 \leq i \leq M$

$$\mathbf{X} = \vec{u}_1 \otimes \vec{u}_2 \otimes \cdots \otimes \vec{u}_M = \prod_{k=1}^{M} \otimes \vec{u}_k. \tag{10}$$

The rank of an arbitrary M*th-order tensor* $\mathbf{X}$, *denoted by* $R = \text{rank}(\mathbf{X})$, *is the minimum number of rank-1 tensors that yield* $\mathbf{X}$ *in a linear combination.*

3 The relationship between LSM and MLSM

Suppose: 1) we have a dimension reduction algorithm A_1 , which finds a sequence of linear transformation matrices $U_i \in R^{L_i \times L_i'}$ ($L_i' < L_i$, $1 \leq i \leq M$) to transform a big-size tensor $\mathbf{X} \in R^{L_1 \times L_2 \times \cdots \times L_M}$ to a small-size tensor $\mathbf{Y}_1 \in R^{L_1' \times L_2' \times \cdots \times L_M'}$, i.e., $\mathbf{Y}_1 = \mathbf{X} \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times \cdots \times_M U_M^{\mathrm{T}}$; and 2) we have another dimension reduction algorithm A_2 , which finds a linear transformation matrix $U \in R^{L \times L'}$ ($L = L_1 \times L_2 \times \cdots \times L_M$ and $L' = L_1' \times L_2' \times \cdots \times L_M'$; $L_i' < L_i$) to transform a high-dimensional vector $\vec{x} = \mathrm{vect}(\mathbf{X})$ to a low-dimensional vector $\vec{y}_2 = \mathrm{vect}(\mathbf{Y}_2)$, i.e., $\vec{y}_2 = U^{\mathrm{T}} \vec{x}$, where vect $(\cdot)$ is the vectorization operator; $\vec{x} \in R^L$ and $\vec{y}_2 \in R^{L'}$. According to [55], we know

$$\begin{aligned} \vec{y}_1 &= \mathrm{vect}(\mathbf{Y}_1) \\ &= \mathrm{vect}\left(\mathbf{X} \times_1 U_1^{\mathrm{T}} \times_2 U_2^{\mathrm{T}} \times \cdots \times_M U_M^{\mathrm{T}}\right) \\ &= (U_1 \otimes U_2 \otimes \cdots \otimes U_M)^{\mathrm{T}} \mathrm{vect}(\mathbf{X}) . \end{aligned} \tag{11}$$

Therefore, if $U = U_1 \otimes U_2 \otimes \cdots \otimes U_M$, $\vec{y}_2 = \vec{y}_1$. That is, the algorithm A_1 equals algorithm A_2, if the linear transformation matrix $U \in R^{L \times L'}$ in A_2 equals to $U_1 \otimes U_2 \otimes \cdots \otimes U_M$.[2]

The tensor representation helps to reduce the number of parameters needed to model the data. In A_1, there are $N_1 = \sum_{i=1}^{M} L_i L_i'$ independent parameters. While in A_2, there are $N_2 = \prod_{i=1}^{M} L_i \prod_{i=1}^{M} L_i'$ independent parameters. In statistical learning, we usually require the number of the training measurements is larger than that of the parameters to model these training measurements for linear algorithms. In the training stage of the MLSM-based learning algorithms, we usually use the alternating projection method to obtain a solution, i.e., the linear projection matrices are obtained independently, so we only need about $N_0 = \max_i\{L_i L_i'\}$ training measurements to obtain a solution for MLSM-based learning algorithms. However, we need about N_2 training measurements to obtain a solution for LSM-based learning algorithms. That is, the MLSM-based learning algorithms requires much smaller training measurements than LSM-based learning algorithms, because $N_0 \ll N_2$. Therefore, the tensor representation helps to redeuce the small sample-size (SSS) problem.

It has a long history to reduce the number of parameters to model the data by adding constraints. Take the strategies in Gaussian distribution estimation as an example[3]: when the data consist of only a few training measurements embedded in a high-dimensional space, we always add some constraints to the covariance matrix, for example by requiring the covariance matrix to be a diagonal matrix. Therefore, to better characterize or classify natural data, a scheme should preserve as many as possible of the original constraints. When the training measurements

[2] In (11), we conduct the reshape operation on $\mathbf{U} = U_1 \otimes U_2 \otimes \cdots \otimes U_M$. That is, originally $\mathbf{U}$ lies in $R^{L_1 \times L_1' \times L_2 \times L_2' \times \cdots \times L_M \times L_M'}$ and after the reshape operation $\mathbf{U}$ is transformed to V in $R^{(L_1 \times L_2 \times \cdots \times L_M) \times (L_1' \times L_2' \times \cdots \times L_M')}$. Then, we can apply the transpose operation on V.

[3] Constraints in MLSM/STL are justified by the form of the data. However, constraints in the example are ad hoc.

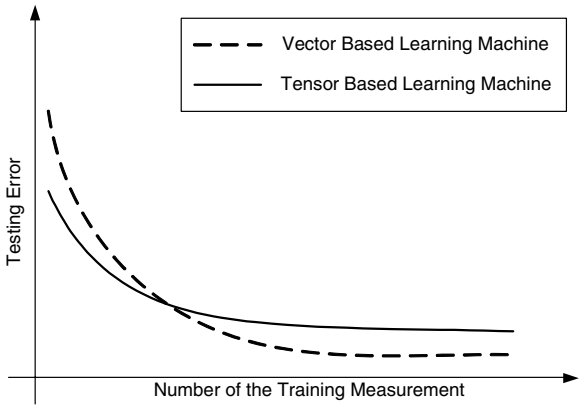


Fig. 5 Tensor-based learning machine versus the vector-based learning machine

are limited, these constraints help to give reasonable solutions to classification problems.

Based on the discussions above, we have the following results:

1) when the number of the training measurements is limited, the vectorization operation always leads to the SSS problem. That is, for small-size training set, we need to use the MLSM-based learning algorithms, because the LSM-based learning algorithms will overfit the data. The vectorization of a tensor into a vector makes it hard to keep track of the information in spatial constraints. For example, two 4-neighbor connected pixels in an image may be hugely separated from each other after a vectorization;
2) when the number of the training measurements is large, the MLSM-based learning algorithms will underfit the data. In this case, the vectorization operation for the data is helpful because it increases the number of parameters to model the data.

Similarly, if we choose to use the vector-based learning algorithms, the vectorization operation $\text{vect}(\cdot)$ is applied to a general tensor $\mathbf{X}$ and forms a vector $\vec{x} = \text{vect}(\mathbf{X}) \in R^L$, where $L = L_1 \times L_2 \times \cdots \times L_M$. The vectorization eliminates the structure information of a measurement in its original format. However, the information is helpful to reduce the number of parameters in a learning model and results in alleviating the overfitting problem. Usually, the testing error decreases with respect to the increasing number of the training measurements. When the number of the training measurements is limited, the tensor-based learning machine performs better than the vector-based learning machine. Otherwise, the vector-based learning machine outperforms the tensor-based learning machine, as shown in Fig. 5.

4 Convex optimization-based learning

Learning models are always formulated as optimization problems [50, 54]. Therefore, mathematical programming [50, 54] is the heart of the machine learning

research [28]. Recently, mathematical programming has been applied for semisupervised learning [1, 18]. In this section, we first introduce the fundamentals of convex optimization and then give out a general formulation for convex optimization-based learning.

A mathematical programming problem [3, 50, 54] has the form or it can be transformed to this form

$$\begin{bmatrix} \min\limits_{\vec{w}} & f_0(\vec{w}) & \\ & f_i(\vec{w}) \leq 0, & 1 \leq i \leq m \\ \text{s.t.} & h_i(\vec{w}) = 0, & 1 \leq i \leq p \end{bmatrix} \tag{12}$$

where $\vec{w} = [w_1, w_2, \ldots, w_n]^{\mathrm{T}} \in R^n$ is the optimization variable in Eq. (12); the function $f_0 : R^n \to R$ is the objective function; the functions $f_i : R^n \to R$ are inequality constraint functions; and the functions $h_i : R^n \to R$ are equality constraint functions. A vector $\vec{w}^*$ is a solution to the problem if f_0 achieves its minimum among all possible vectors, i.e., vectors satisfy all constraint functions ($f_i|_{i=1}^m$ and $h_i|_{i=1}^p$).

When the objective function $f_0(\vec{w})$ and the inequality constraint functions $f_i(\vec{w})|_{i=1}^m$ satisfy

$$\begin{array}{l} f_i(\alpha\vec{w}_1 + \beta\vec{w}_2) \leq \alpha f_i(\vec{w}_1) + \beta f_i(\vec{w}_2) \\ \alpha, \beta \in R_+ \cdots and \cdots \alpha + \beta = 1 \\ w_1, \vec{w}_2 \in R^n \end{array} \tag{13}$$

(i.e., $f_i(\vec{w})|_{i=0}^m$ are convex functions) and the equality constraint functions $h_i(\vec{w})|_{i=1}^p$ are affine (i.e., $h_i(\vec{w}) = 0$ can be simplified as $\vec{a}_i^{\mathrm{T}}\vec{w} = b_i$), the mathematical programming problem defined in Eq. (12) is named the convex optimization problem. Therefore, a convex optimization problem [3] is defined by

$$\begin{bmatrix} \min\limits_{\vec{w}} & f_0(\vec{w}) & \\ & f_i(\vec{w}) \leq 0, & 1 \leq i \leq m \\ \text{s.t.} & \vec{a}_i^{\mathrm{T}}\vec{w} = b_i, & 1 \leq i \leq p \end{bmatrix} \tag{14}$$

where $f_i(\vec{w})|_{i=0}^m$ are convex functions. The domain D of the problem in Eq. (14) is the intersection of the domains of $f_i(\vec{w})|_{i=0}^m$, i.e., $D = \cap_{i=0}^m \mathrm{dom} f_i$. The point $\vec{w}^*$ in D is the optimal solution of Eq. (14) if and only if

$$\nabla^{\mathrm{T}} f_0(\vec{w}^*)(\vec{w} - \vec{w}^*) \geq 0, \quad \forall \vec{w} \in D \tag{15}$$

The convex optimization problem defined in Eq. (14) consists of a large number of popular special cases, such as the linear programming (LP) [44], the linear fractional programming (LFP) [3], the quadratic programming (QP) [21], the quadratically constrained quadratic programming (QCQP) [19], the second-order cone programming (SOCP) [19], the semidefinite programming (SDP) [43], and the geometric programming (GP) [4]. All of these special cases have been widely applied in different areas, such as computer networks, machine learning, computer vision, psychology, the health research, the automation research, and economics.

The significance of a convex optimization problem is that the solution is unique (i.e., the locally optimal solution is also the globally optimal solution), so the convex optimization has been widely applied to machine learning for many years, such as LP [44] in the linear programming machine (LPM) [22, 30], QP [21] in the support vector machines (SVM) [5, 10, 27, 28, 34, 35, 45], SDP [43] in the distance metric learning (DML) [49] and the kernel matrix learning [15], and SOCP [19] in minimax probability machine (MPM) [16, 31]. This section reviews some basic concepts for supervised learning based on convex optimization, such as SVM, MPM, Fisher discriminant analysis (FDA) [6, 8, 14], and DML.

Now, we introduce LP, QP, QCQP, SOCP, and SDP, which have been widely used to model learning problems.

The LP is defined by

$$\begin{bmatrix} \min\limits_{\vec{w}} & \vec{c}^{\mathrm{T}}\vec{w} \\ \text{s.t.} & G\vec{w} \le \vec{h} \\ & A\vec{w} = \vec{b} \end{bmatrix}, \tag{16}$$

where $G \in R^{m\times n}$ and $A \in R^{p\times n}$. That is, the convex optimization problem degenerates to LP when the objective and constraint functions in the convex optimization problem defined in Eq. (14) are all affine.

The QP is defined by

$$\begin{bmatrix} \min\limits_{\vec{w}} & \frac{1}{2}\vec{w}^{\mathrm{T}}P\vec{w} + \vec{q}^{\mathrm{T}}\vec{w} + r \\ & G\vec{w} \le \vec{h} \\ \text{s.t.} & \\ & A\vec{w} = \vec{b} \end{bmatrix} \tag{17}$$

where $P \in S_+^n$, $G \in R^{m\times n}$ and $A \in R^{p\times n}$. Therefore, the convex optimization problem degenerates to QP when the objective function in Eq. (14) is convex quadratic and the constraint functions in Eq. (14) are all affine.

If the inequality constraints are not affine but quadratic, Eq. (17) transfroms to QCQP, i.e.,

$$\begin{bmatrix} \min\limits_{\vec{w}} & \frac{1}{2}\vec{w}^{\mathrm{T}}P_0\vec{w} + \vec{q}_0^{\mathrm{T}}\vec{w} + r_0 \\ & \frac{1}{2}\vec{w}^{\mathrm{T}}P_i\vec{w} + \vec{q}_i^{\mathrm{T}}\vec{w} + r_i, 1 \le i \le m \\ \text{s.t.} & \\ & A\vec{w} = \vec{b} \end{bmatrix} \tag{18}$$

where $P_i \in S_+^n$ for $0 \le i \le m$.

SOCP has the form

$$\begin{bmatrix} \min\limits_{\vec{w}} & \vec{f}^{\mathrm{T}}\vec{w} \\ & \|A_i\vec{w} + b_i\|_{\mathrm{Fro}} \le \vec{c}_i^{\mathrm{T}}\vec{w} + d_i, 1 \le i \le m \\ \text{s.t.} & \\ & F\vec{w} = \vec{g} \end{bmatrix} \tag{19}$$

where $A_i \in R^{n_i\times n}$, $F \in R^{p\times n}$, $\vec{c}_i \in R^n$, $\vec{g} \in R^p$, $b_i \in R^{n_i}$, and $d_i \in R$. The constraint with the form $\|A\vec{w} + b\| \le \vec{c}^{\mathrm{T}}\vec{w} + d$ is called the second-order cone constraint. When $\vec{c}_i = 0$ for all $1 \le i \le m$, SOCP transforms to QCQP.

Recently, SDP has become an important technique in machine learning and many SDP-based learning machines have been developed. SDP minimizes a linear function subject to a matrix semidefinite constraint

$$\left[\begin{array}{ll} \min\limits_{\vec{w}} & \mathbf{E}^{\mathrm{T}}\vec{w} \\ \text{s.t.} & F(\vec{w}) = F_0 + \sum\limits_{i=1}^{n} w_i F_i \geq 0 \end{array}\right] \tag{20}$$

where $F_i \in S^m$ for all $0 \leq i \leq n$ and $\vec{c} \in R^n$.

As the end of this section, we provide a general formula for convex optimization-based learning as

$$\left[\begin{array}{ll} \min\limits_{\vec{w},b,\vec{\xi}} & f(\vec{w}, b, \vec{\xi}) \\ \text{s.t.} & y_i c_i \left(\vec{w}^{\mathrm{T}}\vec{x}_i + b\right) \geq \xi_i, \quad 1 \leq i \leq N \end{array}\right] \tag{21}$$

where $f : R^{L+N+1} \rightarrow R$ is a criterion (convex function) for classification; $c_i : R^{L+N+1} \rightarrow R$ for all $1 \leq i \leq N$ are convex constraint functions; $\vec{x}_i \in R^L$ $(1 \leq i \leq N)$ are training measurements and their class labels are given by $y_i \in \{+1, -1\}$; $\vec{\xi} = [\xi_1, \xi_2, \ldots, \xi_N]^{\mathrm{T}} \in R^N$ are slack variables; and $\vec{w} \in R^L$ and $b \in R$ determine the classification hyperplane, i.e., $y(\vec{x}) = \mathrm{sign}[\vec{w}^{\mathrm{T}}\vec{x} + b]$. By defining different classification criteria f and convex constraint functions $c_i|_{i=1}^N$, we can obtain a large number of learning machines, such as SVM, MPM, FDA, and DML. We detail this in the next section.

5 Supervised tensor learning: a framework

STL extends the vector-based learning algorithms to accept general tensors as input. In STL, we have N training measurements $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ represented by tensors associated with class label information $y_i \in \{+1, -1\}$. We want to separate the positive measurements $(y_i = +1)$ from the negative measurements $(y_i = -1)$ based on a criterion. This extension is obtained by replacing $\vec{x}_i \in R^L$ $(1 \leq i \leq N)$ and $\vec{w} \in R^L$ with $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ $(1 \leq i \leq N)$ and $\vec{w}_k \in R^{L_k}$ $(1 \leq k \leq M)$ in (21), respectively. Therefore, STL is defined by

$$\left[\begin{array}{ll} \min\limits_{\vec{w}_k|_{k=1}^M, b, \vec{\xi}} & f\left(\vec{w}_k|_{k=1}^M, b, \vec{\xi}\right) \\ \text{s.t.} & y_i c_i \left(\mathbf{X}_i \prod\limits_{k=1}^{M} \times_k \vec{w}_k + b\right) \geq \xi_i, \quad 1 \leq i \leq N \end{array}\right] \tag{22}$$

There are two different points between the vector-based learning and the tensor-based learning: 1) the training measurements are represented by vectors in vector-based learning, while they are represented by tensors in tensor-based learning; and 2) the classification decision function is defined by $\vec{w} \in R^L$ and $b \in R$ in vector-based learning $(y(\vec{x}) = \mathrm{sign}[\vec{w}^{\mathrm{T}}\vec{x} + b])$, while the classification decision function is defined by $\vec{w}_k \in R^{L_k}$ $(1 \leq k \leq M)$ and $b \in R$ in tensor-based

learning, i.e., $y(\mathbf{X}) = \text{sign}[\mathbf{X}\prod_{k=1}^{M}\times_k \vec{w}_k + b]$. In vector-based learning, we have the classification hyperplane, i.e., $\vec{w}^{\mathrm{T}}\vec{x} + b = 0$. While in tensor-based learning, we define the classification tensorplane, i.e., $\mathbf{X}\prod_{k=1}^{M}\times_k \vec{w}_k + b = 0$.

The Lagrangian for STL defined in Eq. (22) is

$$\begin{aligned} & L\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}, \vec{\alpha}\right) \\ &= f\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}\right) - \sum_{i=1}^{N}\alpha_i\left(y_i c_i\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) - \xi_i\right) \\ &= f\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}\right) - \sum_{i=1}^{N}\alpha_i y_i c_i\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) + \vec{\alpha}^{\mathrm{T}}\vec{\xi} \end{aligned} \tag{23}$$

with Lagrangian multipliers $\vec{\alpha} = [\alpha_1, \alpha_2, \ldots, \alpha_N]^{\mathrm{T}} \geq 0$.

The solution is determined by the saddle point of the Lagrangian

$$\max_{\vec{\alpha}}\min_{\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}} L\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}, \vec{\alpha}\right) \tag{24}$$

The derivative of $L(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}, \vec{\alpha})$ with respect to $\vec{w}_j$ is

$$\begin{aligned} \partial_{\vec{w}_j} L &= \partial_{\vec{w}_j} f - \sum_{i=1}^{N}\alpha_i y_i \partial_{\vec{w}_j} c_i\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) \\ &= \partial_{\vec{w}_j} f - \sum_{i=1}^{N}\alpha_i y_i \frac{dc_i}{dz}\partial_{\vec{w}_j}\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) \\ &= \partial_{\vec{w}_j} f - \sum_{i=1}^{N}\alpha_i y_i \frac{dc_i}{dz}\left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right), \end{aligned} \tag{25}$$

where $z = \mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b$. The derivative of $L(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}, \vec{\alpha})$ with respect to b is

$$\begin{aligned} \partial_b L &= \partial_b f - \sum_{i=1}^{N}\alpha_i y_i \partial_b c_i\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) \\ &= \partial_b f - \sum_{i=1}^{N}\alpha_i y_i \frac{dc_i}{dz}\partial_b\left(\mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b\right) \\ &= \partial_b f - \sum_{i=1}^{N}\alpha_i y_i \frac{dc_i}{dz}, \end{aligned} \tag{26}$$

where $z = \mathbf{X}_i\prod_{k=1}^{M}\times_k \vec{w}_k + b$.

To obtain a solution to STL, we need to set $\partial_{\vec{w}_j} L = 0$ and $\partial_b L = 0$.

Table 1 Alternating projection for the supervised tensor learning

Input:	The training measurements $\mathbf{X}_i\|_{i=1}^N \in R^{L_1 \times L_2 \times \cdots \times L_M}$, and the associated class lable $y_i = \{+1, -1\}$.
Output:	The parameters in classification tensorplane $\vec{w}_k\|_{k=1}^M \in R^{L_k}$ and $b \in R$, such that the STL objective function $f(\vec{w}_k\|_{k=1}^M, b, \vec{\xi})$ defined in Eq. (22) is minimized.
Step 1:	Set $\vec{w}_k\|_{k=1}^M$ equal to random unit vectors in R^{L_k};
Step 2:	Carry out steps 3–5 iteratively until convergence;
Step 3:	For $j = 1$ to M:
Step 4:	Obtain $\vec{w}_j \in R^{L_j}$ by optimizting $\left[\begin{array}{l} \min\limits_{\vec{w}_j, b, \vec{\xi}} f(\vec{w}_j, b, \vec{\xi}) \\ \text{s.t. } y_i c_i\left[\vec{w}_j^{\mathrm{T}}\left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) + b\right] \geq \xi_i, 1 \leq i \leq N \end{array}\right]$;
Step 5:	End;
Step 6:	Convergence checking: if $\sum_{k=1}^M \left[\left\|\vec{w}_{k,t}^{\mathrm{T}} \vec{w}_{k,t-1}\right\| \left(\left\|\vec{w}_{k,t}\right\|_{\mathrm{Fro}}^{-2}\right) - 1\right] \leq \varepsilon$, the calculated $\vec{w}_k\|_{k=1}^M$ have converged. Here $\vec{w}_{k,t}$ is the current projection vector and $\vec{w}_{k,t-1}$ is the previous projection vector. ;
Step 7:	End.

According to Eq. (25), we have

$$\partial_{\vec{w}_j} L = 0 \Rightarrow \partial_{\vec{w}_j} f = \sum_{i=1}^{N} \alpha_i y_i \frac{dc_i}{dz}\left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) \tag{27}$$

According to Eq. (26), we have

$$\partial_b L = 0 \Rightarrow \partial_b f = \sum_{i=1}^{N} \alpha_i y_i \frac{dc_i}{dz} \tag{28}$$

Based on Eq. (27), we find the solution to $\vec{w}_j$ depends on $\vec{w}_k$ ($1 \leq k \leq M$, $k \neq j$). That is, we cannot obtain the solution to STL directly. The alternating projection provides a cue to have a solution to STL. The key idea in the alternating projection optimization for STL is to obtain the $\vec{w}_j$ with the given $\vec{w}_k$ ($1 \leq k \leq M$, $k \neq j$) in an iterative way. The algorithm is given in Table 1. The convergence issue is proved in Theorem 1.

The alternating projection procedure to obtain a solution in STL is illustrated in Fig. 6. In this figure, the training measurements are represented by third-order tensors. The following three steps are conducted iteratively to obtain the solution for STL:

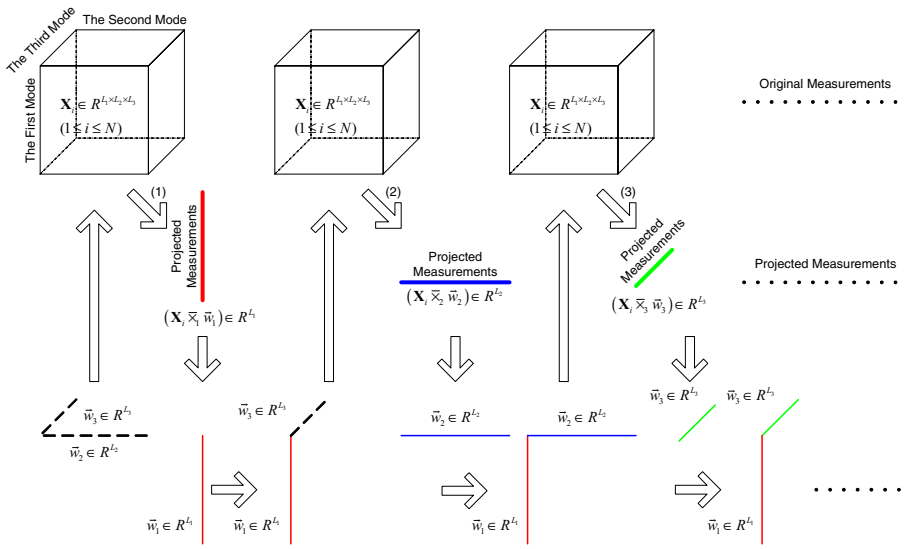


Fig. 6 The third-order tensor example for the alternating projection in STL

1) Generate the second projection vector $\vec{w}_2$ and third projection vectors $\vec{w}_3$ randomly according to the Step 1 in Table 1; project the original training measurements (third-order tensors) $\mathbf{X}_i \in R^{L_1 \times L_2 \times L_3} (1 \leq i \leq N)$ through $\vec{w}_2$ and $\vec{w}_3$ as $(\mathbf{X}_i \bar{\times}_1 \vec{w}_1) \in R^{L_1}$; and calculate the first projection vector $\vec{w}_1$ according to the Step 4 in Table 1 based on the projected training measurements $(\mathbf{X}_i \bar{\times}_1 \vec{w}_1)$;
2) Project the original training measurements $\mathbf{X}_i|_{i=1}^N$ to the calculated first projection vector $\vec{w}_1$ and the original third projection vector $\vec{w}_3$; and calculate the second projection vector $\vec{w}_2$ according to the Step 4 in Table 1 based on the projected training measurements $(\mathbf{X}_i \bar{\times}_2 \vec{w}_2)$;
3) Project the original training measurements $\mathbf{X}_i|_{i=1}^N$ by the previous $\vec{w}_1$ and $\vec{w}_2$; and calculate $\vec{w}_3$ through the Step 4 in Table 1 based on the projected training measurements $(\mathbf{X}_i \bar{\times}_3 \vec{w}_3)$.

Theorem 1 *The alternating projection optimization procedure for STL converges.*

Proof Formally, the alternating projection method never increases the function value $f(\vec{w}_k|_{k=1}^M, b, \vec{\xi})$ of STL between two successive iterations, because it can be interpreted as a type of a monotonic algorithm. We can define a continuous function:

$$f : \bar{u}_1 \times \bar{u}_2 \times \cdots \times \bar{u}_M \times R \times R^N = \overset{M}{\underset{k=1}{\times}} \bar{u}_k \times R \times R^N \rightarrow R$$

where $\vec{w}_d \in \bar{u}_d$ and $\bar{u}_d$ is the set, which includes all possible $\vec{w}_d$. The bias $b \in R$ and the slack variables $\vec{\xi} \in R^N$.

With the definition, f has M different mappings:

$$g(\vec{w}_d^*, b_d^*, \vec{\xi}_d^*) \stackrel{\Delta}{=} \arg \min_{\vec{u}_d \in \bar{u}_d, b, \vec{\xi}} f\left(\vec{w}_d|_{d=1}^M, b, \vec{\xi}\right)$$
$$= \arg \min_{\vec{u}_d \in \bar{u}_d, b, \vec{\xi}} f\left(\vec{w}_d, b, \vec{\xi}; \vec{w}_l|_{l=1}^{d-1}, \vec{w}_l|_{l=d+1}^M\right),$$

The mapping can be calculated with the given $\vec{w}_l|_{l=1}^{d-1}$ in the t^{th} iteration and $\vec{w}_l|_{l=d+1}^M$ in the $(t-1)^{\text{th}}$ iteration of the for-loop in Step 4 in Table 1.

If each $\bar{u}_d$ for all $d \in \{1, 2, \ldots M\}$ is closed, each $g(\vec{w}_d^*, b_d^*, \vec{\xi}_d^*)$ for all $d \in \{1, 2, \ldots M\}$ is closed.

Given an initial $\vec{w}_d \in \bar{u}_d$ $(1 \le d \le M)$, the alternating projection generates a sequence of items $\{\vec{w}_{d,t}^*, b_{d,t}^*, \vec{\xi}_{d,t}^*; 1 \le d \le M\}$ via

$$g\left(\vec{w}_{d,t}^*, b_{d,t}^*, \vec{\xi}_{d,t}^*\right) = \arg \min_{\vec{u}_d \in \bar{u}_d, b, \vec{\xi}} f\left(\vec{w}_d, b, \vec{\xi}; \vec{w}_{l,t}|_{l=1}^{d-1}, \vec{w}_{l,t-1}|_{l=d+1}^M\right)$$

with each $d \in \{1, 2, \cdots M\}$. The sequence has the following relationship:

$$\begin{aligned} a_1 &= f(\vec{w}_{1,1}^*, b_{1,1}^*, \vec{\xi}_{1,1}^*) \ge f(\vec{w}_{2,1}^*, b_{2,1}^*, \vec{\xi}_{2,1}^*) \\ &\ge \cdots \ge f(\vec{w}_{M,1}^*, b_{M,1}^*, \vec{\xi}_{M,1}^*) \ge f(\vec{w}_{1,2}^*, b_{1,2}^*, \vec{\xi}_{1,2}^*) \\ &\ge \cdots \ge f(\vec{w}_{1,t}^*, b_{1,t}^*, \vec{\xi}_{1,t}^*) \ge f(\vec{w}_{2,t}^*, b_{2,t}^*, \vec{\xi}_{2,t}^*) \\ &\ge \cdots \ge f(\vec{w}_{1,T}^*, b_{1,T}^*, \vec{\xi}_{1,T}^*) \ge f(\vec{w}_{2,T}^*, b_{2,T}^*, \vec{\xi}_{2,T}^*) \\ &\ge \cdots \ge f(\vec{w}_{M,T}^*, b_{M,T}^*, \vec{\xi}_{M,T}^*) = a_2. \end{aligned}$$

where $T \to +\infty$. Here, both a_1 and a_2 are limited values in the R space. The alternating projection in STL can be illustrated by a composition of M subalgorithms defined as

$$\Omega_d : \left(\vec{w}_d|_{d=1}^M, b, \vec{\xi}\right) \mapsto \prod_{l=1}^{d-1} \times_l \vec{w}_l \times \text{Map}(\vec{w}_d, b, \vec{\xi}) \prod_{l=d+1}^{M} \times_l \vec{w}_l.$$

It follows that $\Omega \doteq \Omega_1 \circ \Omega_2 \circ \cdots \circ \Omega_M = \circ_{d=1}^M \Omega_d$ is a closed algorithm whenever all $\bar{u}_d$ are compact. All subalgorithms $g(\vec{w}_d^*, b_d^*, \vec{\xi}_d^*)$ decrease the value of f, so it should be clear that Ω is monotonic with respect to f. Consequently, we can say that the alternating projection method to optimize STL defined in Eq. (22) converges. □

6 Supervised tensor learning: Examples

Based on the proposed STL and its alternating projection training algorithm, a large number of tensor-based learning algorithms can be obtained by combining STL with different learning criteria, such as SVM, MPM, FDA, and DML.

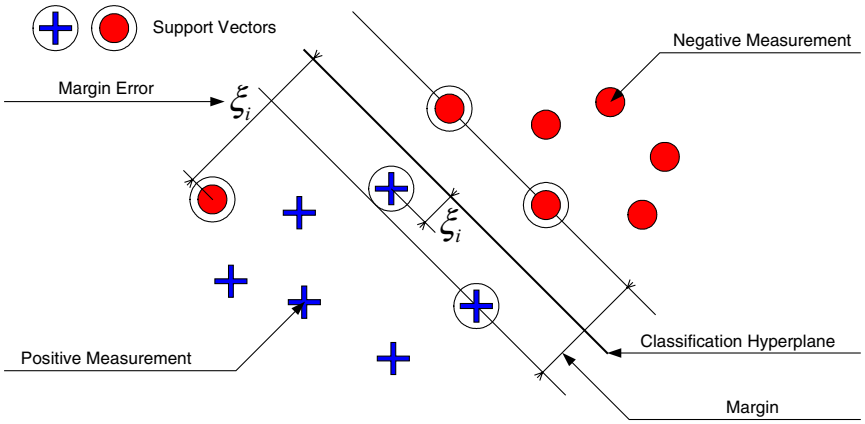


Fig. 7 SVM maximizes the margin between the positive and negative training measurements

6.1 Support vector machine versus support tensor machine

SVM [5, 10, 27, 28, 34, 35, 45] finds a classification hyperplane, which maximizes the margin between the positive measurements and the negative measurements, as shown in Fig. 7.

Suppose there are N training measurements $\vec{x}_i \in R^L$ $(1 \leq i \leq N)$ associated with the class lables $y_i \in \{+1, -1\}$. The traditional SVM [5, 45], i.e., soft-margin SVM, finds a projection vector $\vec{w} \in R^L$ and a bias $b \in R$ through

$$\left[\begin{array}{ll} \min\limits_{\vec{w},b,\vec{\xi}} & J_{C-\mathrm{SVM}}(\vec{w}, b, \vec{\xi}) = \dfrac{1}{2}\|\vec{w}\|_{\mathrm{Fro}}^2 + c\sum\limits_{i=1}^{N}\xi_i \\ \text{s.t.} & y_i[\vec{w}^{\mathrm{T}}\vec{x}_i + b] \geq 1 - \xi_i, 1 \leq i \leq N \\ & \vec{\xi} \geq 0 \end{array}\right] \tag{29}$$

where $\vec{\xi} = [\xi_1, \xi_2, \ldots, \xi_N]^{\mathrm{T}} \in R^N$ is the vector of all slack variables to deal with the linearly nonseparable problem. The ξ_i $(1 \leq i \leq N)$ is also called the marginal error for the ith training measurement, as shown in Fig. 7. The margin is $2/\|\vec{w}\|_{\mathrm{Fro}}$. When the classification problem is linearly separable, we can set $\vec{\xi} = 0$. The decision function for classification is $y(\vec{x}) = \mathrm{sign}[\vec{w}^{\mathrm{T}}\vec{x} + b]$.

The Lagrangian of Eq. (29) is

$$\begin{aligned} &L(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\kappa}) \\ &= \frac{1}{2}\|\vec{w}\|_{\mathrm{Fro}}^2 + c\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i(y_i[\vec{w}^{\mathrm{T}}\vec{x}_i + b] - 1 + \xi_i) - \sum_{i=1}^{N}\kappa_i\xi_i \\ &= \frac{1}{2}\vec{w}^{\mathrm{T}}\vec{w} + c\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i y_i \vec{w}^{\mathrm{T}}\vec{x}_i - b\vec{\alpha}^{\mathrm{T}}\vec{y} + \sum_{i=1}^{N}\alpha_i - \vec{\alpha}^{\mathrm{T}}\vec{\xi} - \vec{\kappa}^{\mathrm{T}}\vec{\xi} \end{aligned} \tag{30}$$

with Lagrangian multipliers $\alpha_i \geq 0$, $\kappa_i \geq 0$ for $1 \leq i \leq N$. The solution is determined by the saddle point of the Lagrangian

$$\max_{\vec{\alpha},\vec{\kappa}} \min_{\vec{w},b,\vec{\xi}} L(\vec{w}, b, \vec{\xi}, \vec{\alpha}, \vec{\kappa}) \tag{31}$$

This can be achieved by

$$\begin{aligned} \partial_{\vec{w}} L = 0 &\Rightarrow \vec{w} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \\ \partial_b L = 0 &\Rightarrow \vec{\alpha}^{\mathrm{T}} \vec{y} = 0 \\ \partial_{\vec{\xi}} L = 0 &\Rightarrow c - \vec{\alpha} - \vec{\kappa} = 0. \end{aligned} \tag{32}$$

Based on Eq. (32), we can have the dual problem of Eq. (29),

$$\left[\begin{array}{ll} \max\limits_{\vec{\alpha}} & J_D(\vec{\alpha}) = -\dfrac{1}{2} \sum\limits_{i=1}^{N} \sum\limits_{j=1}^{N} y_i y_j \vec{x}_i^{\mathrm{T}} \vec{x}_j \alpha_i \alpha_j + \sum\limits_{i=1}^{N} \alpha_i \\ \text{s.t.} & \vec{\alpha}^{\mathrm{T}} \vec{y} = 0 \\ & 0 \leq \vec{\alpha} \leq c \end{array} \right] \tag{33}$$

Set $P = [y_i y_j \vec{x}_i^{\mathrm{T}} \vec{x}_j]_{1 \leq i,j \leq N}$, $\vec{q} = \vec{1}_{N \times 1}$, $A = \vec{y}$, $\vec{b} = 0$, $G = [I_{N \times N}, -I_{N \times N}]^{\mathrm{T}}$, and $\vec{h} = [c\vec{1}_{N \times 1}^{\mathrm{T}}, \vec{0}_{N \times 1}^{\mathrm{T}}]^{\mathrm{T}}$ in Eq. (17), we can see that the dual problem of Eq. (29) in SVM is a QP.

In the soft-margin SVM defined in Eq. (29), the constant c determines the tradeoff between 1) maximizing the margin between the positive and negative measurements and 2) minimizing the training error. The constant c is not intuitive. Therefore, [28, 27] developed the ν-SVM by replacing the unintuitive parameter c with an intuitive parameter ν as

$$\left[\begin{array}{ll} \min\limits_{\vec{w},b,\vec{\xi},\rho} & J_{\nu-\mathrm{SVM}}(\vec{w}, b, \vec{\xi}, \rho) = \dfrac{1}{2} \|\vec{w}\|_{\mathrm{Fro}}^2 + \dfrac{1}{N} \sum\limits_{i=1}^{N} \xi_i - \nu\rho \\ & y_i[\vec{w}^{\mathrm{T}} \vec{x}_i + b] \geq \rho - \xi_i, \quad 1 \leq i \leq N \\ \text{s.t.} & \vec{\xi} \geq 0, \\ & \rho \geq 0 \end{array} \right] \tag{34}$$

The significance of ν in ν-SVM defined in Eq. (34) is that it controls the number of support vectors and the marginal errors.

Suykens and Vandewalle [33, 34] simplified the soft-margin SVM as the least squares SVM,

$$\left[\begin{array}{ll} \min\limits_{\vec{w},b,\vec{\varepsilon}} & J_{\mathrm{LS-SVM}}(\vec{w}, b, \vec{\varepsilon}) = \dfrac{1}{2} \|\vec{w}\|_{\mathrm{Fro}}^2 + \dfrac{\gamma}{2} \vec{\varepsilon}^{\mathrm{T}} \vec{\varepsilon} \\ \text{s.t.} & y_i[\vec{w}^{\mathrm{T}} \vec{x}_i + b] = 1 - \varepsilon_i, \quad 1 \leq i \leq N \end{array} \right] \tag{35}$$

Here, the penalty $\gamma > 0$. There are two different points between the soft-margin SVM defined in Eq. (29) and the least squares SVM defined in Eq. (35): 1)

inequality constraints are replaced by equality constraints; and 2) the loss $\sum_{i=1}^{N} \xi_i$ ($\xi_i \geq 0$) is replaced by square loss. The two modifications enable the solution of the least-square SVM to be conveniently obtained compared to the soft-margin SVM.

According to the statistical learning theory, we know a learning machine performs well when the number of the training measurements is larger than the complexity of the model. Moreover, the complexity of the model and the number of the parameters to describe the model are always in direct proportion. In computer vision research, the objects are usually represented by general tensors and the number of the training measurements is limited. Therefore, it is reasonable to have the tensor extension of SVM, i.e., the support tensor machine (STM). Based on Eq. (29) and STL defined in Eq. (22), it is not difficult to obtain the tensor extension of the soft-margin SVM, i.e., the soft-margin STM.

Suppose we have training measurements $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ $(1 \leq i \leq N)$ and their corresponding class labels $y_i \in \{+1, -1\}$. The decision function is a multilinear function $y(\mathbf{X}) = \text{sign}[\mathbf{X} \prod_{k=1}^{M} \times_k \vec{w}_k + b]$, where the projection vectors $\vec{w}_k \in R^{L_k}$ $(1 \leq k \leq M)$ and the bias b in soft-margin STM are obtained from

$$\left[\begin{array}{ll} \min\limits_{\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}} & J_{\text{C-STM}}\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}\right) = \frac{1}{2}\left\| \bigotimes_{k=1}^{M} \vec{w}_k \right\|_{\text{Fro}}^2 + c\sum\limits_{i=1}^{N} \xi_i \\ \text{s.t.} & y_i\left[\mathbf{X}_i \prod\limits_{k=1}^{M} \times_k \vec{w}_k + b\right] \geq 1 - \xi_i, \quad 1 \leq i \leq N \\ & \vec{\xi} \geq 0 \end{array}\right] \tag{36}$$

Here, $\vec{\xi} = [\xi_1, \xi_2, \ldots, \xi_N]^{\text{T}} \in R^N$ is the vector of all slack variables to deal with the linearly nonseparable problem.

The Lagrangian for this problem is

$$\begin{aligned} & L\left(\vec{w}_k|_{k=1}^{M}, b, \vec{\xi}, \vec{\alpha}, \vec{\kappa}\right) \\ & = \frac{1}{2}\left\| \bigotimes_{k=1}^{M} \vec{w}_k \right\|_{\text{Fro}}^2 + c\sum_{i=1}^{N} \xi_i \\ & \quad - \sum_{i=1}^{N} \alpha_i \left(y_i \left[\mathbf{X}_i \prod_{k=1}^{M} \times_k \vec{w}_k + b \right] - 1 + \xi_i \right) - \sum_{i=1}^{N} \kappa_i \xi_i \\ & = \frac{1}{2} \prod_{k=1}^{M} \vec{w}_k^{\text{T}} \vec{w}_k + c\sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i y_i \left(\mathbf{X}_i \prod_{k=1}^{M} \times_k \vec{w}_k \right) \\ & \quad - b\vec{\alpha}^{\text{T}} \vec{y} + \sum_{i=1}^{N} \alpha_i - \vec{\alpha}^{\text{T}} \vec{\xi} - \vec{\kappa}^{\text{T}} \vec{\xi} \end{aligned} \tag{37}$$

with Lagrangian multipliers $\alpha_i \geq 0$, $\kappa_i \geq 0$ for $1 \leq i \leq N$. The solution is determined by the saddle point of the Lagrangian

$$\max_{\vec{\alpha},\vec{\kappa}} \min_{\vec{w}_k|_{k=1}^M,b,\vec{\xi}} L\left(\vec{w}_k|_{k=1}^M, b, \vec{\xi}, \vec{\alpha}, \vec{\kappa}\right) \tag{38}$$

This can be achieved by

$$\begin{aligned} &\partial_{\vec{w}_j} L = 0 \Rightarrow \vec{w}_j = \frac{1}{\prod\limits_{k=1}^{k\neq j} \vec{w}_k^{\mathrm{T}} \vec{w}_k} \sum_{i=1}^{N} \alpha_i y_i \left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) \\ &\partial_b L = 0 \Rightarrow \vec{\alpha}^{\mathrm{T}} \vec{y} = 0 \\ &\partial_{\vec{\xi}} L = 0 \Rightarrow c - \vec{\alpha} - \vec{\kappa} = 0 \end{aligned} \tag{39}$$

The first equation in Eq. (39) shows that the solution of $\vec{w}_j$ depends on $\vec{w}_k (1 \leq k \leq M, k \neq j)$. That is, we cannot obtain the solution for soft-margin STM directly. This point has been emphasized in the STL framework. Therefore, we use the proposed alternating projection method in STL to obtain the solution of soft-margin STM. To have the alternating projection method for soft-margin STM, we need to replace the Step 4 in Table 1 by the following optimization problem,

$$\left[\begin{array}{ll} \min\limits_{\vec{w}_j,b,\vec{\xi}} & J_{\mathrm{C-STM}}(\vec{w}_j, b, \vec{\xi}) = \frac{\eta}{2} \left\|\vec{w}_j\right\|_{\mathrm{Fro}}^2 + c \sum\limits_{i=1}^{N} \xi_i \\ & y_i[\vec{w}_j^{\mathrm{T}} \left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) + b] \geq 1 - \xi_i, 1 \leq i \leq N \\ \text{s.t.} & \vec{\xi} \geq 0 \end{array}\right] \tag{40}$$

where $\eta = \prod_{1\leq k\leq M}^{k\neq j} \|\vec{w}_k\|_{\mathrm{Fro}}^2$.

The problem defined in Eq. (40) is the standard soft-margin SVM defined in Eq. (29).

Based on ν-SVM defined in Eq. (34) and STL defined in Eq. (22), we can also have the tensor extension of the ν-SVM, i.e., ν-STM,

$$\left[\begin{array}{ll} \min\limits_{\vec{w}_k|_{k=1}^M,b,\vec{\xi},\rho} & J_{\nu-\mathrm{STM}}\left(\vec{w}_k|_{k=1}^M, b, \vec{\xi}, \rho\right) = \frac{1}{2} \left\| \bigotimes\limits_{k=1}^{M} \vec{w}_k \right\|_{\mathrm{Fro}}^2 + \frac{1}{N} \sum\limits_{i=1}^{N} \xi_i - \nu\rho \\ & y_i\left[\mathbf{X}_i \prod\limits_{k=1}^{M} \times_k \vec{w}_k + b\right] \geq \rho - \xi_i, 1 \leq i \leq N \\ \text{s.t.} & \vec{\xi} \geq 0, 1 \leq i \leq N \\ & \rho \geq 0 \end{array}\right] \tag{41}$$

Here, $\nu \geq 0$ is a constant. The Lagrangian for this problem is

$$\begin{aligned} & L\left(\vec{w}_k|_{k=1}^M, b, \vec{\xi}, \rho, \vec{\alpha}, \vec{\kappa}, \tau\right) \\ & = \frac{1}{2}\left\| \bigotimes_{k=1}^M \vec{w}_k \right\|_{\text{Fro}}^2 + \frac{1}{N}\sum_{i=1}^N \xi_i - \nu\rho - \tau\rho \\ & \quad - \sum_{i=1}^N \alpha_i \left(y_i \left[\mathbf{X}_i \prod_{k=1}^M \times_k \vec{w}_k + b \right] - \rho + \xi_i \right) - \vec{\kappa}^{\mathrm{T}}\vec{\xi} \\ & = \frac{1}{2}\prod_{k=1}^M \vec{w}_k^{\mathrm{T}}\vec{w}_k + \frac{1}{N}\sum_{i=1}^N \xi_i - \sum_{i=1}^N \alpha_i y_i \left(\mathbf{X}_i \prod_{k=1}^M \times_k \vec{w}_k \right) \\ & \quad - b\vec{\alpha}^{\mathrm{T}}\vec{y} + \rho\sum_{i=1}^N \alpha_i - \vec{\alpha}^{\mathrm{T}}\vec{\xi} - \vec{\kappa}^{\mathrm{T}}\vec{\xi} - \nu\rho - \tau\rho, \end{aligned} \tag{42}$$

with Lagrangian multipliers $\tau \geq 0$ and $\alpha_i \geq 0$, $\kappa_i \geq 0$ for $1 \leq i \leq N$. The solution is determined by the saddle point of the Lagrangian

$$\max_{\vec{\alpha},\vec{\kappa},\tau} \min_{\vec{w}_k|_{k=1}^M, b, \vec{\xi}, \rho} L\left(\vec{w}_k|_{k=1}^M, b, \vec{\xi}, \rho, \vec{\alpha}, \vec{\kappa}, \tau\right) \tag{43}$$

Similar to the soft-margin STM, the solution of $\vec{w}_j$ depends on $\vec{w}_k$ ($1 \leq k \leq M, k \neq j$), because

$$\partial_{\vec{w}_j} L = 0 \Rightarrow \vec{w}_j = \frac{1}{\prod_{k=1}^{k\neq j} \vec{w}_k^{\mathrm{T}}\vec{w}_k} \sum_{i=1}^N \alpha_i y_i \left(\mathbf{X}_i \bar{\times}_j \vec{w}_j \right) \tag{44}$$

Therefore, we use the proposed alternating projection method in STL to obtain the solution of ν-STM. To have the alternating projection method for ν-STM, we need to replace the Step 4 in Table 1 by the following optimization problem,

$$\left[\begin{array}{ll} \min\limits_{\vec{w}_j, b, \vec{\xi}, \rho} & J_{\nu-\text{STM}}\left(\vec{w}_j, b, \vec{\xi}, \rho\right) = \dfrac{\eta}{2}\left\|\vec{w}_j\right\|_{\text{Fro}}^2 + \dfrac{1}{N}\sum\limits_{i=1}^N \xi_i - \nu\rho \\ & y_i\left[\vec{w}_j^{\mathrm{T}}\left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) + b\right] \geq \rho - \xi_i, \quad 1 \leq i \leq N \\ \text{s.t.} & \vec{\xi} \geq 0 \\ & \rho \geq 0 \end{array} \right] \tag{45}$$

where $\eta = \prod_{1\leq k\leq M}^{k\neq j} \|\vec{w}_k\|_{\text{Fro}}^2$.

The problem defined in Eq. (45) is the standard ν-SVM defined in Eq. (34).

Based on the least squares SVM defined in Eq. (35) and the STL defined in Eq. (22), we can also have the tensor extension of the least-square SVM, i.e.,

least-square STM,

$$\left[\begin{array}{l} \min\limits_{\vec{w}_k|_{k=1}^M, b, \vec{\varepsilon}} J_{LS-STM}\left(\vec{w}_k|_{k=1}^M, b, \vec{\varepsilon}\right) = \dfrac{1}{2}\left\|\bigotimes\limits_{k=1}^M \vec{w}_k\right\|_{\text{Fro}}^2 + \dfrac{\gamma}{2}\vec{\varepsilon}^{\text{T}}\vec{\varepsilon} \\ \text{s.t.} \qquad y_i\left[\mathbf{X}_i \prod\limits_{k=1}^M \times_k \vec{w}_k + b\right] = 1 - \varepsilon_i, 1 \le i \le N \end{array}\right] \tag{46}$$

where $\gamma > 0$ is a constant. Similar to the soft-margin STM and ν-STM, there is no closed-form solution for least squares STM. We use the alternating projection method in STL to obtain the solution of the least squares STM. To have the alternating projection method for the least squares STM, we need to replace the Step 4 in Table 1 by the following optimization problem,

$$\left[\begin{array}{l} \min\limits_{\vec{w}_j, b, \vec{\varepsilon}} J_{LS-STM}\left(\vec{w}_k|_{k=1}^M, b, \vec{\varepsilon}\right) = \dfrac{\eta}{2}\left\|\vec{w}_j\right\|_{\text{Fro}}^2 + \dfrac{\gamma}{2}\vec{\varepsilon}^{\text{T}}\vec{\varepsilon} \\ \text{s.t.} \quad y_i\left[\vec{w}_j^{\text{T}}\left(\mathbf{X}_i \bar{\times}_j \vec{w}_j\right) + b\right] = 1 - \varepsilon_i, \quad 1 \le i \le N \end{array}\right] \tag{47}$$

where $\eta = \prod_{1\le k\le M}^{k\ne j} \|\vec{w}_k\|_{\text{Fro}}^2$.

Theorem 2 *In STM, the decision function is defined by a multilinear function* $y(\mathbf{X}) = \text{sign}[\mathbf{X}\prod_{k=1}^M \times_k \vec{w}_k + b]$ *with* $\|\bigotimes_{k=1}^M \vec{w}_k\|_{\text{Fro}}^2 \le \Lambda^2$ *and* $\|\mathbf{X}\|_{\text{Fro}}^2 \le R^2$. *Let* $\rho > 0$ *and* v *is the fraction of training measurements with margin smaller than* $\rho/|\Lambda|$. *When STM is obtained from* N *training measurements* $\|\mathbf{X}_i\|_{\text{Fro}}^2 \le R^2$ $(1 \le i \le N)$, *sampled from a distribution* P *with probability at least* $1-\delta$ $(0 < \delta < 1)$, *the misclassification probability of a test measurement sampled from* P *is bounded by*

$$v + \sqrt{\frac{\lambda}{N}\left(\frac{R^2\Lambda^2}{\rho^2}\ln^2 N + \ln\frac{1}{\delta}\right)} \tag{48}$$

where λ *is a universal constant.*

Proof This is a direct conclusion from the theorem on the margin error bound introduced in [28]. More information about other error bounds in SVM can be found in [2]. □

6.2 Minimax probability machine versus tensor minimax probability machine

The minimax probability machine (MPM) [16, 31] has become popular. It is reported to outperform the conventional SVM consistently and, therefore, has attracted attention as a promising supervised learning algorithm. MPM focuses on finding a decision hyperplane, which is $H(\vec{w}, b) = \{\vec{x}|\vec{w}^{\text{T}}\vec{x} + b = 0\}$, to separate the positive measurements from the negative measurements (a binary classification problem) with maximal probability with respect to all distributions modeled by given means and covaraince, as shown in Fig. 8. MPM maximizes the probability of correct classification rate (classification accuracy) on the future measurements.

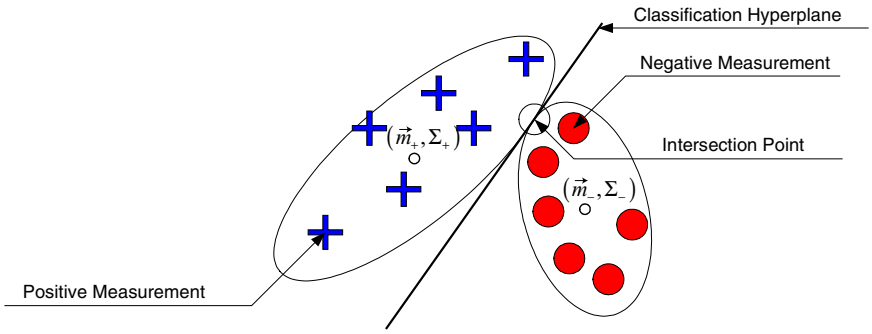


Fig. 8 MPM separates the positive measurements from the negative measurements by maximizing the probability of the correct classification for the future measurements. The intersection point minimizes the maximum of the Mahalanobis distances of the positive and negative measurements, i.e., it has the same Mahalanobis distances to the mean of the positive measurements and the mean of the negative measurements

For Gaussian-distributed measurements, it minimizes the maximum of the Mahalanobis distances of the positive measurements and the negative measurements. With the given positive measurements $\vec{x}_i$ $(y_i = +1)$ and negative measurements $\vec{x}_i$ $(y_i = -1)$, MPM is defined as,

$$\left[\begin{array}{ll} \max\limits_{\vec{w},b,\delta} & J_{\mathrm{MPM}}\left(\vec{w}, b, \delta\right) = \delta \\ \text{s.t.} & \inf\limits_{\vec{x}_i\left(y_i=+1\right)\sim\left(\vec{m}_+,\,\Sigma_+\right)} \Pr\left\{\vec{w}^{\mathrm{T}}\vec{x}_i + b \geq 0\right\} \geq \delta \\ & \inf\limits_{\vec{x}_i\left(y_i=-1\right)\sim\left(\vec{m}_-,\,\Sigma_-\right)} \Pr\left\{\vec{w}^{\mathrm{T}}\vec{x}_i + b \leq 0\right\} \geq \delta \end{array}\right] \tag{49}$$

Here, the notation $\vec{x}_i$ $(y_i = +1) \sim (\vec{m}_+, \Sigma_+)$ means the class distribution of the positive measurements has the mean $\vec{m}_+$ and covariance Σ_+. So does the notation $\vec{x}_i$ $(y_i = -1) \sim (\vec{m}_-, \Sigma_-)$. The classification decision function is given by $y\left(\vec{x}\right) = \mathrm{sign}[\vec{w}^{\mathrm{T}}\vec{x} + b]$.

Recently, based on the powerful Marshall and Olkin's theorem [20], Popescu and Bertsimas [23] proved a probability bound,

$$\sup_{\vec{x}\sim\left(\vec{m},\,\Sigma\right)} \Pr\left\{\vec{x} \in S\right\} = \frac{1}{1+d^2} \quad \text{with} \quad d^2 = \inf_{\vec{x}\in S} \left(\vec{x}-\vec{m}\right)^{\mathrm{T}} \Sigma^{-1} \left(\vec{x}-\vec{m}\right) \tag{50}$$

where $\vec{x}$ stands for a random vector, S is a given convex set, and the supremum is taken over all distributions for $\vec{x}$ with the mean value as $\vec{m}$ and the covariance matrix Σ. Based on this result, Lanckriet et al. [16] reformulated Eq. (49) as:

$$\left[\begin{array}{ll} \max\limits_{\vec{w},b,\kappa} & J_{\mathrm{MPM}}\left(\vec{w}, b, \kappa\right) = \kappa \\ \text{s.t.} & \vec{w}^{\mathrm{T}}\vec{m}_+ + b \geq +\kappa\sqrt{\vec{w}^{\mathrm{T}}\Sigma_+\vec{w}},\ y_i = +1 \\ & \vec{w}^{\mathrm{T}}\vec{m}_- + b \leq -\kappa\sqrt{\vec{w}^{\mathrm{T}}\Sigma_-\vec{w}},\ y_i = -1 \end{array}\right] \tag{51}$$

where the constraint functions in Eq. (51) are second-order cone functions. There MPM is an SOCP. This problem can be further simplified as

$$\begin{bmatrix} \min\limits_{\vec{w}} & J_{\mathrm{MPM}}(\vec{w}) = \sqrt{\vec{w}^{\mathrm{T}}\Sigma_{+}\vec{w}} + \sqrt{\vec{w}^{\mathrm{T}}\Sigma_{-}\vec{w}} \\ \text{s.t.} & \vec{w}^{\mathrm{T}}(\vec{m}_{+} - \vec{m}_{-}) = 1 \end{bmatrix} \tag{52}$$

where b is determined by

$$b* = (\vec{w}*)^{\mathrm{T}}\vec{m}_{+} - \frac{\sqrt{(\vec{w}*)^{\mathrm{T}}\Sigma_{+}(\vec{w}*)}}{\sqrt{(\vec{w}*)^{\mathrm{T}}\Sigma_{+}(\vec{w}*)} + \sqrt{(\vec{w}*)^{\mathrm{T}}\Sigma_{-}(\vec{w}*)}} \tag{53}$$

In computer vision research, many objects are represented by tensors. To match the input requirments in MPM, we need to vectorize the tensors to vectors. When the training measurements are limited, the vectorization will be a disaster. This is because MPM meets the matrix singular problem seriously (the ranks of Σ_{+} and Σ_{-} are deficient). To reduce this problem, we propose the tensor extension of MPM, i.e., tensor MPM (TMPM). TMPM is a combination of MPM and STL.

Suppose we have the training measurements $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ $(1 \le i \le N)$ and their corresponding class labels $y_i \in \{+1, -1\}$. The decision function is a multilinear function $y(\mathbf{X}) = \mathrm{sign}[\mathbf{X}\prod_{k=1}^{M} \times_k \vec{w}_k + b]$, where the projection vectors $\vec{w}_k \in R^{L_k}$ $(1 \le k \le M)$ and the bias b in TMPM are obtained from

$$\begin{bmatrix} \max\limits_{\vec{w}_k|_{k=1}^{M}, b, \kappa} & J_{\mathrm{MPM}}\left(\vec{w}_k|_{k=1}^{M}, b, \kappa\right) = \kappa \\ & \frac{1}{N_{+}}\left(\sum\limits_{i=1}^{N}\left[\mathrm{I}(y_i = +1)\mathbf{X}_i \prod\limits_{k=1}^{M} \times_k \vec{w}_k\right]\right) + b \ge +\kappa \sup\limits_{1 \le l \le M} \sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{+;l}\vec{w}_l} \\ \text{s.t.} & \frac{1}{N_{-}}\left(\sum\limits_{i=1}^{N}\left[\mathrm{I}(y_i = -1)\mathbf{X}_i \prod\limits_{k=1}^{M} \times_k \vec{w}_k\right]\right) + b \le -\kappa \sup\limits_{1 \le l \le M} \sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{-;l}\vec{w}_l} \end{bmatrix} \tag{54}$$

where $\Sigma_{+;l}$ is the covariance matrix of the projected measurements $(\mathbf{X}_i \bar{\times}_{-l} \vec{w}_l)$ for all $y_i = +1$ and $\Sigma_{-;l}$ is the covariance matrix of the projected measurements $(\mathbf{X}_i \bar{\times}_{-l} \vec{w}_l)$ for all $y_i = -1$. The function $\mathrm{I}(y_i = +1)$ is 1 if y_i is $+1$, otherwise 0. The function $\mathrm{I}(y_i = -1)$ is 1 if y_i is -1, otherwise 0. This problem can be simplified as

$$\begin{bmatrix} \max\limits_{\vec{w}_k|_{k=1}^{M}, b, \kappa} & J_{\mathrm{MPM}}\left(\vec{w}_k|_{k=1}^{M}, b, \kappa\right) = \kappa \\ & \mathbf{M}_{+}\prod\limits_{k=1}^{M} \times_k \vec{w}_k + b \ge +\kappa \sup\limits_{1 \le l \le M} \sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{+;l}\vec{w}_l} \\ \text{s.t.} & \mathbf{M}_{-}\prod\limits_{k=1}^{M} \times_k \vec{w}_k + b \le -\kappa \sup\limits_{1 \le l \le M} \sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{-;l}\vec{w}_l} \end{bmatrix} \tag{55}$$

where $\mathbf{M}_+ = \left(1/N_+\right)\sum_{i=1}^{N}[\mathrm{I}(y_i=+1)\mathbf{X}_i]$, $\mathbf{M}_- = \left(1/N_-\right)\sum_{i=1}^{N}[\mathrm{I}(y_i=-1)\mathbf{X}_i]$, N_+ (N_-) is the number of the positive (negative) measurements.

The Lagrangian for this problem is

$$\begin{aligned}
&L\left(\vec{w}_k|_{k=1}^{M}, b, \kappa, \vec{\alpha}\right)\\
&= -\kappa - \alpha_1\left(\mathbf{M}_+\prod_{k=1}^{M}\times_k\vec{w}_k + b - \kappa\sup_{1\le l\le M}\sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{+;l}\vec{w}_l}\right)\\
&\quad + \alpha_2\left(\mathbf{M}_-\prod_{k=1}^{M}\times_k\vec{w}_k + b + \kappa\sup_{1\le l\le M}\sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{-;l}\vec{w}_l}\right)\\
&= -\kappa - \frac{\alpha_1 b}{N_+} + \frac{\alpha_2 b}{N_-} - \alpha_1\mathbf{M}_+\prod_{k=1}^{M}\times_k\vec{w}_k + \alpha_2\mathbf{M}_-\prod_{k=1}^{M}\times_k\vec{w}_k\\
&\quad + \frac{\alpha_1\kappa}{N_+}\sup_{1\le l\le M}\sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{+;l}\vec{w}_l} + \frac{\alpha_2\kappa}{N_-}\sup_{1\le l\le M}\sqrt{\vec{w}_l^{\mathrm{T}}\Sigma_{-;l}\vec{w}_l}
\end{aligned} \tag{56}$$

with Lagrangian multipliers $\alpha_i \ge 0$ ($i = 1, 2$). The solution is determined by the saddle point of the Lagrangian

$$\max_{\vec{\alpha}}\min_{\vec{w}_k|_{k=1}^{M}, b, \kappa} L\left(\vec{w}_k|_{k=1}^{M}, b, \kappa, \vec{\alpha}\right) \tag{57}$$

This can be achieved by setting $\partial_{\vec{w}_j}L = 0$, $\partial_b L = 0$, and $\partial_\kappa L = 0$. It is not difficult to find that the solution of $\vec{w}_j$ depends on $\vec{w}_k$ ($1 \le k \le M$, $k \ne j$). Therefore, there is no closed-form solution for TMPM. We use the proposed alternating projection method in STL to obtain the solution of TMPM. To have the alternating projection method for TMPM, we need to replace the Step 4 in Table 1 by the following optimization problem,

$$\left[\begin{array}{ll}
\max\limits_{\vec{w}_j, b, \kappa} & J_{\mathrm{MPM}}(\vec{w}_j, b, \kappa) = \kappa\\
\text{s.t.} & \vec{w}_j^{\mathrm{T}}(\mathbf{M}_+\bar{\times}_j\vec{w}_j) + b \ge +\kappa\sqrt{\vec{w}_j^{\mathrm{T}}\Sigma_{+;j}\vec{w}_j}\\
 & \vec{w}_j^{\mathrm{T}}(\mathbf{M}_-\bar{\times}_j\vec{w}_j) + b \le -\kappa\sqrt{\vec{w}_j^{\mathrm{T}}\Sigma_{-;j}\vec{w}_j}
\end{array}\right] \tag{58}$$

This problem is the standard MPM defined in Eq. (51).

6.3 Fisher discriminant analysis versus tensor fisher discriminant analysis

Fisher discriminant analysis (FDA) [6, 8, 14] as shown in Fig. 9 has been widely applied for classification. Suppose there are N training measurements $\vec{x}_i \in R^L$ ($1 \le i \le N$) associated with the class lables $y_i \in \{+1, -1\}$. There are N_+ positive training measurements and their mean is $\vec{m}_+ = (1/N_+)\sum_{i=1}^{N}[\mathrm{I}(y_i=+1)\vec{x}_i]$; there are N_- negative training measurements and their mean can be calculated from $\vec{m}_- = (1/N_-)\sum_{i=1}^{N}\left[\mathrm{I}(y_i=-1)\vec{x}_i\right]$; the mean of all training measurements

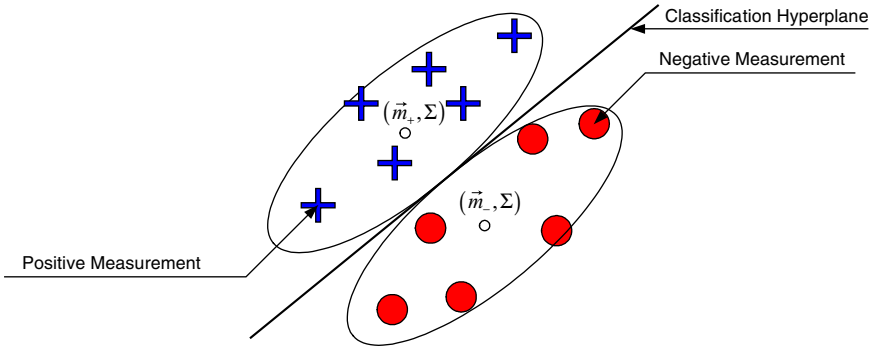


Fig. 9 FDA separates the positive measurements from the negative measurements by maximizing the symmetric Kullback–Leibler divergence between the two classes under the assumption that the two classes share the same covariance matrix

is $\vec{m} = (1/N) \sum_{i=1}^{N} \vec{x}_i$; and the covariance matrix of all training measurements is Σ. FDA finds a direction to separate the class means well while minimizing the variance of the total training measurements. Therefore, two quantities need to be defined, which are: 1) the between-class scatter $S_b = (\vec{m}_2 - \vec{m}_1)(\vec{m}_2 - \vec{m}_1)^{\mathrm{T}}$: measuring the difference between the two classes; and 2) the within-class scatter $S_w = \sum_{i=1}^{N} (\vec{x}_i - \vec{m})(\vec{x}_i - \vec{m})^2 = N\Sigma$: the variance of the total training measurements. The projection direction $\vec{w}$ maximizes

$$\left[\max_{\vec{w}} J_{FDA}(\vec{w}) = \frac{\vec{w}^{\mathrm{T}} S_b \vec{w}}{\vec{w}^{\mathrm{T}} S_w \vec{w}} \right] \tag{59}$$

This problem is simplified as

$$\left[\max_{\vec{w}} J_{FDA}(\vec{w}) = \frac{\left\| \vec{w}^{\mathrm{T}} (\vec{m}_+ - \vec{m}_-) \right\|}{\sqrt{\vec{w}^{\mathrm{T}} \Sigma \vec{w}}} \right] \tag{60}$$

According to [37], we know this procedure is equivalent to maximizing the symmetric Kullback–Leibler divergence (KLD) between the positive and the negative measurements with identical covariances, so that the positive measurements are separated from the negative measurements. Based on the definition of FDA, we know FDA is a special case of the linear discriminant analysis (LDA).

The linear decision function in FDA is $y(\vec{x}) = \mathrm{sign}[\vec{w}^{\mathrm{T}} \vec{x} + b]$, where $\vec{w}$ is the eigenvector of $\Sigma^{-1} (\vec{m}_+ - \vec{m}_-)(\vec{m}_+ - \vec{m}_-)^{\mathrm{T}}$ associated with the largest eigenvalue and the bias b is calculated by

$$b = \frac{N_- - N_+ - (N_+ \vec{m}_+ + N_- \vec{m}_-)^{\mathrm{T}} \vec{w}}{N_- + N_+} \tag{61}$$

The significance [6, 9] of FDA is: FDA is Bayes optimal when the two classes are Gaussian distributed with identical covariances.

When objects are represented by tensors, we need to vectorize the tensors to vectors to match the input requirments in FDA. When the training measurements are limited, the vectorization will be a disaster for FDA. This is because S_w and

S_b are both singular. To reduce this problem, we propose the tensor extension of FDA, i.e., tensor FDA (TFDA). TFDA is a combination of FDA and STL. Moreover, TFDA is a special case of the previous proposed general tensor discriminant analysis (GTDA) [38].

Suppose we have the training measurements $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ $(1 \leq i \leq N)$ and their corresponding class labels $y_i \in \{+1, -1\}$. The mean of the training positive meansurements is $\mathbf{M}_+ = (1/N_+) \sum_{i=1}^{N} [\mathrm{I}(y_i = +1)\mathbf{X}_i]$; the mean of the training negative measurements is given by $\mathbf{M}_- = (1/N_-) \sum_{i=1}^{N} [\mathrm{I}(y_i = -1)\mathbf{X}_i]$; the mean of all training measurements is $\mathbf{M} = (1/N) \sum_{i=1}^{N} \mathbf{X}_i$; and $N_+(N_-)$ is the number of the positive (negative) measurements. The decision function is a multilinear function $y(\mathbf{X}) = \mathrm{sign}[\mathbf{X} \prod_{k=1}^{M} \times_k \vec{w}_k + b]$, where the projection vectors $\vec{w}_k \in R^{L_k}$ $(1 \leq k \leq M)$ and the bias b in TFDA are obtained from

$$\left[\max_{\vec{w}_k|_{k=1}^{M}} J_{\mathrm{TFDA}}\left(\vec{w}_k|_{k=1}^{M}\right) = \frac{\left\| (\mathbf{M}_+ - \mathbf{M}_-) \prod_{k=1}^{M} \times_k \vec{w}_k \right\|^2}{\sum_{i=1}^{N} \left\| (\mathbf{X}_i - \mathbf{M}) \prod_{k=1}^{M} \times_k \vec{w}_k \right\|^2} \right] \tag{62}$$

There is no closed-form solution for TFDA. The alternating projection is applied to obtain the solution for TFDA and we need to replace the Step 4 in Table 1 by the following optimization problem,

$$\left[\max_{\vec{w}_j} J_{\mathrm{TFDA}}(\vec{w}_j) = \frac{\left\| \vec{w}_j^{\mathrm{T}} \left[(\mathbf{M}_+ - \mathbf{M}_-) \bar{\times}_j \vec{w}_j \right] \right\|^2}{\sum_{i=1}^{N} \left\| \vec{w}_j^{\mathrm{T}} [(\mathbf{X}_i - \mathbf{M}) \bar{\times}_j \vec{w}_j] \right\|^2} \right] \tag{63}$$

This problem is the standard FDA. When we have the projection vectors $\vec{w}_k|_{k=1}^{M}$, we can obtain the bias b from

$$b = \frac{N_- - N_+ - (N_+ \mathbf{M}_+ + N_- \mathbf{M}_-) \prod_{k=1}^{M} \times_k \vec{w}_k}{N_- + N_+} \tag{64}$$

6.4 Distance metric learning versus multiple distance metrics learning

Weinberger et al. [49] proposed the distance metric learning (DML) to learn a metric for k-nearest-neighbor (kNN) classification (see Fig. 10). The motivation of DML is simple because the performance of kNN is only related to the metric used for dissimilarity measure. In traditional kNN, this measure is the Euclidean metric, which fails to capture the statistical charateristics of the training measurements. In DML, the metric is obtained to guarantee: 1) k-nearest neighbors of a measurement have the identical label with the measurement; and 2) measurements with different lables are separated from the measurement according to margin maximization.

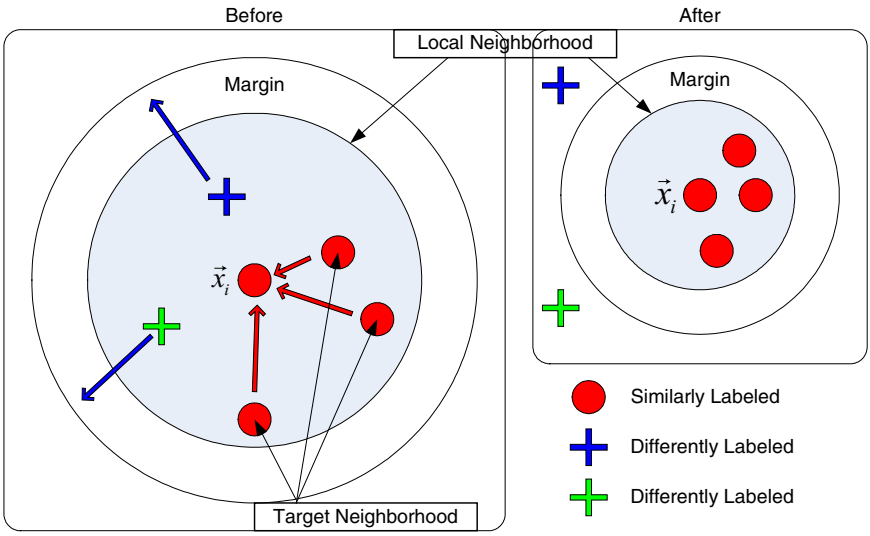


Fig. 10 DML obtains a metric to guarantee: 1) k-nearest neighbors of a measurement have the identical label with the measurement; and 2) measurements with different lables are separated from the measurement according to margin maximization

DML is defined by

$$\left[\begin{array}{ll} \min\limits_{\substack{\Sigma,\xi_{ijl}\\ 1\le i,j,l\le N}} & J_{\mathrm{DML}}(\Sigma,\xi_{ijl}) = \sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}(\vec{x}_i-\vec{x}_j)^{\mathrm{T}}\Sigma(\vec{x}_i-\vec{x}_j) \\ & \qquad + c\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}\,(1-y_{il})\,\xi_{ijl} \\ & (\vec{x}_i-\vec{x}_l)^{\mathrm{T}}\,\Sigma\,(\vec{x}_i-\vec{x}_l)-(\vec{x}_i-\vec{x}_j)^{\mathrm{T}}\Sigma(\vec{x}_i-\vec{x}_j) \\ \text{s.t.} & \quad \ge 1-\xi_{ijl},\ 1\le i,j,l\le N \\ & \xi_{ijl}\ge 0,\ 1\le i,j,l\le N \\ & \Sigma\ge 0 \end{array}\right] \tag{65}$$

where $\eta_{ij}=1$ and $y_{ij}=1$ mean that $\vec{x}_i$ and $\vec{x}_j$ have the same class label, otherwise 0. The constraint function $\Sigma\ge 0$ indicates that the maxtrix Σ is required to be positive semidefinite, so the problem is an SDP. From the learnt distance metric Σ, it is direct to have the linear transformation matrix by decomposing $\Sigma=W^{\mathrm{T}}W$.

The optimization problem defined in Eq. (65) is equivalent to

$$\left[\begin{array}{ll} \min\limits_{\substack{\Sigma,\xi_{ijl}\\ 1\le i,j,l\le N}} & J_{\mathrm{DML}}(\Sigma,\xi_{ijl}) = \mathrm{tr}(A^{\mathrm{T}}\Sigma A) + c\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}\,(1-y_{il})\,\xi_{ijl} \\ & B_{ijl}^{\mathrm{T}}\Sigma B_{ijl} \ge 1-\xi_{ijl},\ 1\le i,j,l\le N \\ \text{s.t.} & \xi_{ijl}\ge 0,\ 1\le i,j,l\le N \\ & \Sigma\ge 0 \end{array}\right] \tag{66}$$

where $A = [\sqrt{\eta_{ij}}(\vec{x}_i - \vec{x}_j)]_{L\times N^2} (1 \le i, j \le N)$ and $B_{ijl} = [\vec{x}_i - \vec{x}_l, \vec{x}_j - \vec{x}_i]_{L\times 2}$

Suppose we have the training measurements $\mathbf{X}_i \in R^{L_1\times L_2\times\cdots\times L_M}$ $(1 \le i \le N)$ and their corresponding class labels $y_i \in \{1, 2, \ldots, n\}$. The multiple distance metrics learning (MDML) learns M metrics $\Sigma_k = W_k^{\mathrm{T}}W_k$ $(1 \le k \le M)$ for $\mathbf{X}_i|_{i=1}^{N}$ to make the measurements, which have the same (different) labels, and are as close (far) as possible. The MDML is defined as

$$\left[\begin{array}{ll} \min\limits_{\substack{W_k|_{k=1}^{M},\xi_{ijl}\\ 1\le i,j,l\le N}} & J_{\mathrm{MDML}}\left(W_k|_{k=1}^{M},\xi_{ijl}|_{1\le i,j,l\le N}\right) \\ & = \left[\begin{array}{l} \sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}\left\|(\mathbf{X}_i-\mathbf{X}_j)\prod\limits_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 \\ +c\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}\,(1-y_{il})\,\xi_{ijl} \end{array}\right] \\ \text{s.t.} & \left\|(\mathbf{X}_i-\mathbf{X}_l)\prod\limits_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 - \left\|(\mathbf{X}_i-\mathbf{X}_j)\prod\limits_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 \ge 1-\xi_{ijl} \\ & \xi_{ijl}\ge 0,\ 1\le i,j,l\le N \\ & W_k^{\mathrm{T}}W_k \ge 0,\ 1\le k\le M \end{array}\right] \tag{67}$$

As described in STL framework, there is also no closed-form solution for MDML. The alternating projection method is applied to obtain the solution for

MDML and we need to replace Step 4 in Table 1 by the following optimization problem,

$$\left[\begin{array}{ll} \min\limits_{\substack{W_p,\xi_{ijl}\\ 1\le i,j,l\le N}} & J_{\mathrm{MDML}}(W_p,\xi_{ijl}|_{1\le i,j,l\le N}) \\ & = \mathrm{tr}\left(A_p^{\mathrm{T}}\Sigma_p A_p\right) + c\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\eta_{ij}\,(1-y_{il})\,\xi_{ijl} \\ \text{s.t.} & \left\|(\mathbf{X}_i-\mathbf{X}_l)\prod\limits_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 - \left\|(\mathbf{X}_i-\mathbf{X}_j)\prod\limits_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 \ge 1-\xi_{ijl} \\ & \xi_{ijl}\ge 0,\ 1\le i,j,l\le N \\ & W_k^{\mathrm{T}}W_k \ge 0,\ 1\le k\le M \end{array}\right] \tag{68}$$

Here,

$$A_p = \sum_{i=1}^{N}\sum_{j=1}^{N}\sqrt{\eta_{ij}}\mathrm{mat}_p((\mathbf{X}_i-\mathbf{X}_j)\bar{\times}_p W_p) \tag{69}$$

and

$$B_{ijl;p} = \mathrm{mat}_p((\mathbf{X}_j-\mathbf{X}_l)\bar{\times}_p W_p) \tag{70}$$

This is because $\|(\mathbf{X}_i-\mathbf{X}_l)\prod_{k=1}^{M}\times_k W_k\|_{\mathrm{Fro}}^2 = \mathrm{tr}(\mathrm{mat}_j^{\mathrm{T}}((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)\Sigma_j \mathrm{mat}_j((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j))$

Derivation

$$\begin{aligned}
&\left\|(\mathbf{X}_i-\mathbf{X}_l)\prod_{k=1}^{M}\times_k W_k\right\|_{\mathrm{Fro}}^2 \\
&= \left[\!\!\left[\left((\mathbf{X}_i-\mathbf{X}_l)\prod_{k=1}^{M}\times_k W_k\right)\otimes\left((\mathbf{X}_i-\mathbf{X}_l)\prod_{k=1}^{M}\times_k W_k\right);(1:M)\,(1:M)\right]\!\!\right] \\
&= \mathrm{tr}[\![([(\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j]\times_j W_j)\otimes([(\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j] \\
&\quad \times_j W_j);(1:M)\,(1:M)]\!] \\
&= \mathrm{tr}(W_j[\![[(\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j]\otimes[(\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j];(\bar{j})(\bar{j})]\!]W_j^{\mathrm{T}}) \\
&= \mathrm{tr}(W_j\mathrm{mat}_j((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)\mathrm{mat}_j^{\mathrm{T}}((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)W_j^{\mathrm{T}}) \\
&= \mathrm{tr}(\mathrm{mat}_j^{\mathrm{T}}((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)W_j^{\mathrm{T}}W_j\mathrm{mat}_j((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)) \\
&= \mathrm{tr}(\mathrm{mat}_j^{\mathrm{T}}((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)\Sigma_j\mathrm{mat}_j((\mathbf{X}_i-\mathbf{X}_l)\bar{\times}_j W_j)).
\end{aligned}$$

This problem defined in Eq. (68) is the standard DML.

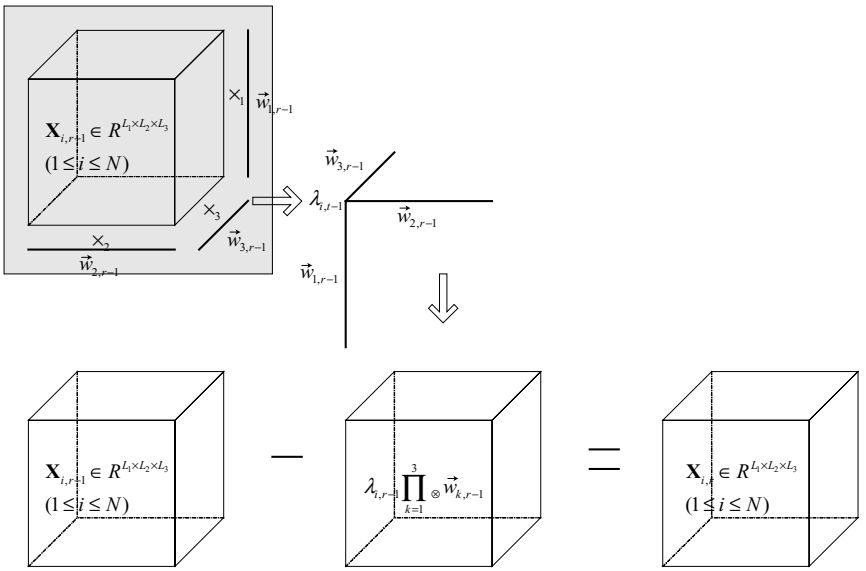


Fig. 11 Iterative feature extraction model for third-order tensors

7 Iterative feature extraction model based on supervised tensor learning

The iterative feature extraction model (IFEM) based on STL is an extension of the STL framework for feature extraction and its procedure is similar to the recursive rank-one tensor approximation developed by Shashua and Levin in [29]. An example of IFEM is given in [39].

Suppose we have the training measurements $\mathbf{X}_i \in R^{L_1 \times L_2 \times \cdots \times L_M}$ $(1 \le i \le N)$ and their corresponding class labels $y_i \in \{+1, -1\}$. IFEM is defined by

$$\mathbf{X}_{i,r} = \mathbf{X}_{i,r-1} - \lambda_{i,r-1} \prod_{k=1}^{M} \times_k \vec{w}_{k,r-1} \tag{71}$$

$$\lambda_{i,r-1} = \mathbf{X}_{i,r-1} \prod_{k=1}^{M} \times_k (\vec{w}_{k,r-1})^{\mathrm{T}} \tag{72}$$

$$\begin{bmatrix} \min\limits_{\vec{w}_{k,r}|_{k=1}^{M}, b, \vec{\xi}} & f\left(\vec{w}_{k,r}|_{k=1}^{M}, b, \vec{\xi}\right) \\ \text{s.t.} & y_i c_i \left(\mathbf{X}_{i,r} \prod\limits_{k=1}^{M} \times_k \vec{w}_{k,r} + b\right) \ge \xi_i, \quad 1 \le i \le N \end{bmatrix} \tag{73}$$

where $\mathbf{X}_{i,1} = \mathbf{X}_i$ and $\lambda_{i,0} = 0$. The $\lambda_{i,r}|_{r=1}^{R}$ (R is the number of the extracted features in IFEM) is used to represent the original tensor $\mathbf{X}_i$.

From the definition of IFEM, which is defined by Eqs. (71)–(73), we know that IFEM can be calculated by a greedy approach. The calculation of $\mathbf{X}_{i,r}|_{i=1}^{N}$

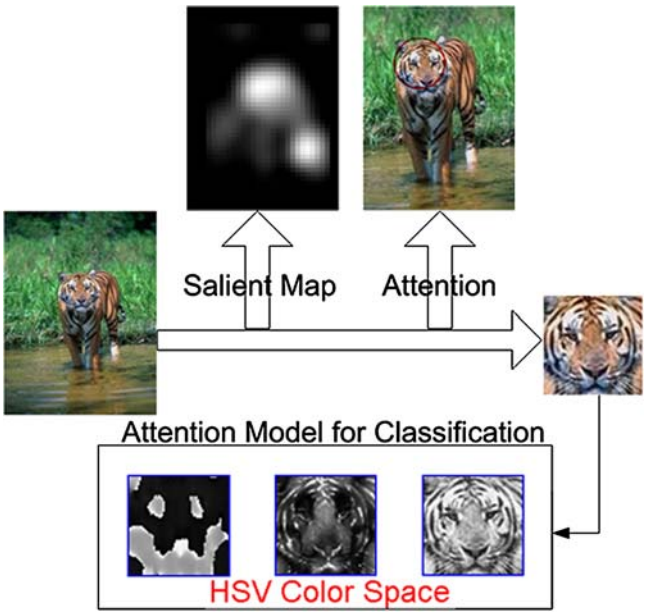


Fig. 12 Attention model for image representation

is based on the given $\mathbf{X}_{i,r-1}|_{i=1}^{N}$ and $\vec{w}_{k,r-1}|_{k=1}^{M}$. With the given $\mathbf{X}_{i,r-1}|_{i=1}^{N}$ and $\vec{w}_{k,r-1}|_{k=1}^{M}$, we can calculate $\lambda_{i,r-1}$ via Eq. (72). The projection vectors $\vec{w}_{k,r}|_{k=1}^{M}$ can be obtained by optimizing Eq. (73) through the alternating projection method in Table 1. The flowchart of the algorithm for feature extraction for third-order tensors is illustrated in Fig. 11.

With IFEM, we can obtain $\vec{w}_{k,r}|_{1\leq k\leq M}^{1\leq r\leq R}$ iteratively. The coordinate values $\lambda_{i,r}|_{r=1}^{R}$ can represent the original tensor $\mathbf{X}_i$. For example, in nearest neighbor-based recognition, the prototype tensor $\mathbf{X}_p$ for each individual class in the database and the testing tensor $\mathbf{X}_t$ to be classified are projected onto the bases to get the prototype vector $\lambda_{p,r}|_{r=1}^{R}$ and the testing vector $\lambda_{t,r}|_{r=1}^{R}$. The testing tensor class is found by minimizing the Euclidean distance $\varepsilon = \sqrt{\sum_{r=1}^{R}(\lambda_{t,r} - \lambda_{p,r})^2}$ over p.

8 Experiment

In this section, we provide experiments for image classification with TMPM and MPM. The experiments demonstrate that tensor representation can reduce the overfitting problem, i.e., STL is a powerful tool for classification in computer vision research.

8.1 TMPM for image classification

To categorize images into groups based on their semantic contents is very important and challenging. Its fundamental task is the binary classification and thus a

hierarchical structure can be built according to a series of binary classifiers. As a result, this semantic image classification can make the growing image repositories easy to search and browse [24, 25]. The image semantic classification is also of great help for many applications.

In this STL-based classification experiment, two groups of images are separated from each other by a trained TMPM, which plays the role of an example of the generalized learning machines within the STL framework. The input (representing features) of TMPM is the region of interest (ROI) within an image, which is exacted by the attention model in [12, 13, 32] and represented as a third-order tensor.

The attention model [12, 13] is capable of reproducing human performances for a number of pop-out tasks [42]. In other words, when a target is different from its surroundings by its unique orientation, color, intensity, or size, it is always the first attended location and easy to be noticed by the observer. Therefore, utilizing the attention model-based ROI to describe an image's semantic information is reasonable.

As shown in Fig. 12, representing an attention region from an image consists of several steps: 1) extracting the salient map as introduced by Itti et al. [12, 13]; 2) finding the most attentive region, whose center has the largest value in the salient map; 3) extracting the attention region by a square, i.e., ROI, in size of 64×64; and 4) finally, representing this ROI in the hue, saturation, and value (HSV) perceptual color space. Consequently, we have a third-order tensor for the image representation.

Note that although we only select a small region from the image, the size of the extracted third-order tensor is already as large as $64 \times 64 \times 3$. If we vectorize this tensor, the dimensionality of the vector will be 12,288. From the following paragraphs, we will be aware that the sizes of the training/testing sets are only of hundreds, which are clearly much smaller than 12,288. Therefore, it always meets the matrix singular problem when a third-order tensor is reformed to comply with the input requirements (vectors) of conventional learning machines. On the contrary, our proposed tensor-oriented supervised learning scheme can avoid this problem directly and meanwhile represent the ROIs much more naturally.

The training set and the testing set for the following experiments are built upon the Corel photo gallery [47], from which 100 images are selected for each of the two categories of measurements as shown in Figs. 13 and 14. These 200 images are then processed to extract the third tensor attention features for TMPM as shown in Figs. 15 and 16.

We choose the *Tiger* category shown in in Fig. 13 and the *Leopard* category shown in Fig. 14 for this binary classification experiment since it is a very difficult task for a machine to distinguish them although a human being can differentiate a tiger from a leopard or vice versa easily. Basically, the characteristics of a classifier cannot be examined adequately when facing a simple problem, for example, classifying grassland pictures from blood pictures. The *Tiger* and *Leopard* classification is carried out in this Section. We choose the top N images as training sets according to the image IDs, while all remaining images are used to form the corresponding testing set.

In our experiments, the introduced third tensor attention ROIs can mostly be found correctly from the images. Some successful results, respectively, being

Fig. 13 Example images from the *Tiger* category

extracted from the *Tiger* category and the *Leopard* category, are shown in Figs. 15 and 16. By this means, the underlying data structures are kept well for the next classification step. However, we should note that the attention model sometimes cannot depict the semantic information of an image. This is mainly because the attention model always locates the region that is different from its surroundings and thus might be cheated when some complex or bright background exists. Some unsuccessful ROIs can also be found from Figs. 15 and 16. It should be emphasized that to keep the following comparative experiments fair and automatic, these wrongly extracted ROIs are not excluded from the training sets. Therefore, it is another challenge for both the conventional learning machine (MPM) and our newly proposed one (TMPM).

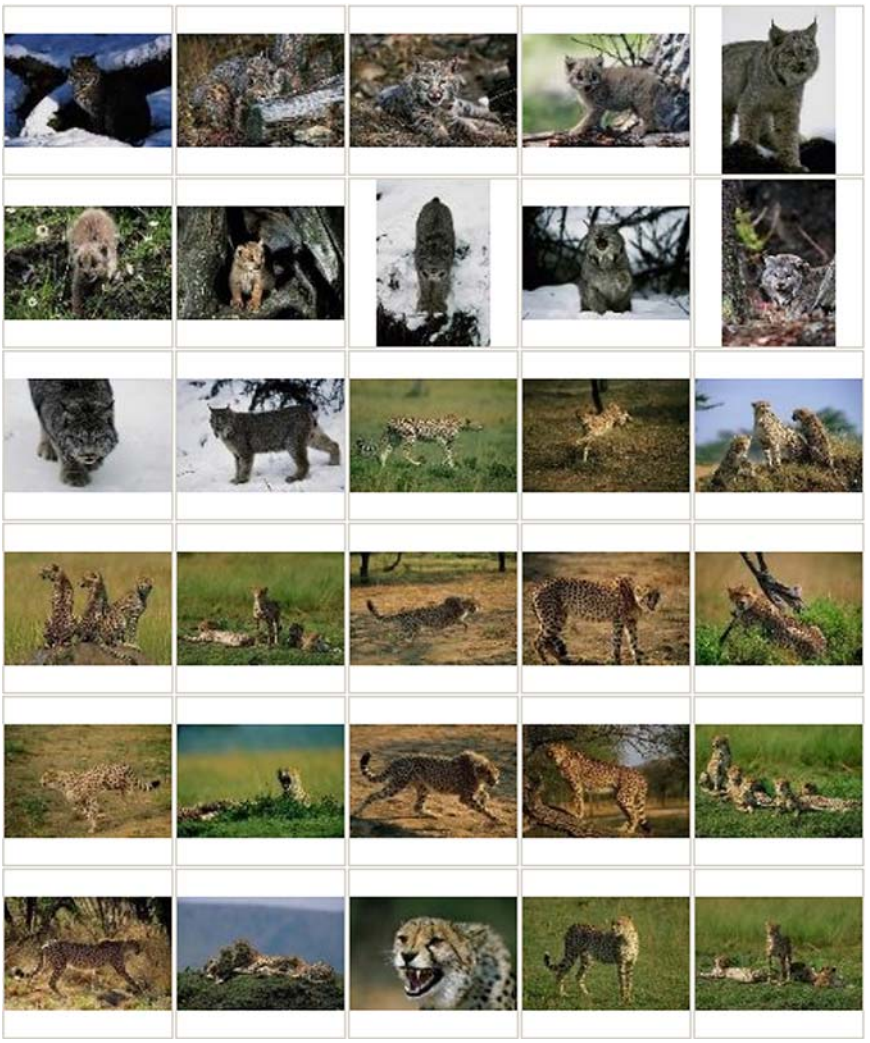


Fig. 14 Example images from the *leopard* category

We carried out the binary classification (*Tiger* and *Leopard*) experiments upon the above training and testing sets. The proposed TMPM is compared with the MPM. The experimental results are shown in Table 2. Error rates for both training and testing are reported according to the increasing size of the training set (STS) from 5 to 30 with a step 5.

From the training error rates in Table 2, it can be seen that the traditional method (MPM) cannot learn a satisfying model for classification when the size of the measurement set is much smaller than the features' dimensionality in the learning state. However, the machine learning algorithm under the proposed STL framework (TMPM) has a good characteristic on the volume control according to the computational learning theory and its real performances.

Fig. 15 One hundred ROIs in the *Tiger* category

Also from Table 2, based on the testing error rates of the comparative experiments, the proposed TMPM algorithm is demonstrated to be more effective to represent the intrinsic discriminative information (in the form of the third-order ROIs). TMPM learns a better classification model for future data classification than MPM and thus has a satisfactory performance on the testing set. It is

Table 2 TMPM versus MPM

	Training error rate		Testing error rate	
STS	TMPM	MPM	TMPM	MPM
5	0.0000	0.4000	0.4600	0.5050
10	0.0000	0.5000	0.4250	0.4900
15	0.0667	0.4667	0.3250	0.4150
20	0.0500	0.5000	0.2350	0.4800
25	0.0600	0.4800	0.2400	0.4650
30	0.1167	0.5000	0.2550	0.4600

Fig. 16 One hundred ROIs in the *Leopard* category

observed that the TMPM error rate is a decreasing function of the size of the training set. This is actually consistent with the statistical learning theory.

We also evaluate TMPM as a sample algorithm of the proposed STL framework. Two important issues in machine learning are studied, namely, the convergence property and the insensitiveness to the initial values.

Fig. 17 shows that as a sample algorithm of the STL framework, TMPM converges efficiently by the alternating projection method. Usually, 20 iterations are enough to achieve the convergence.

Three subfigures in the left column of Fig. 17 show tensor-projected position values of the original general tensors with an increasing number of learning iterations using 10, 20, and 30 training measurements for each class, respectively, from top to bottom. We find that the projected values converge at stable values. Three subfigures in the right column of Fig. 17 show the training error rates and the testing error rates according to the increasing number of learning iterations by 10, 20, and 30 training measurements for each class, respectively, from top to bottom.

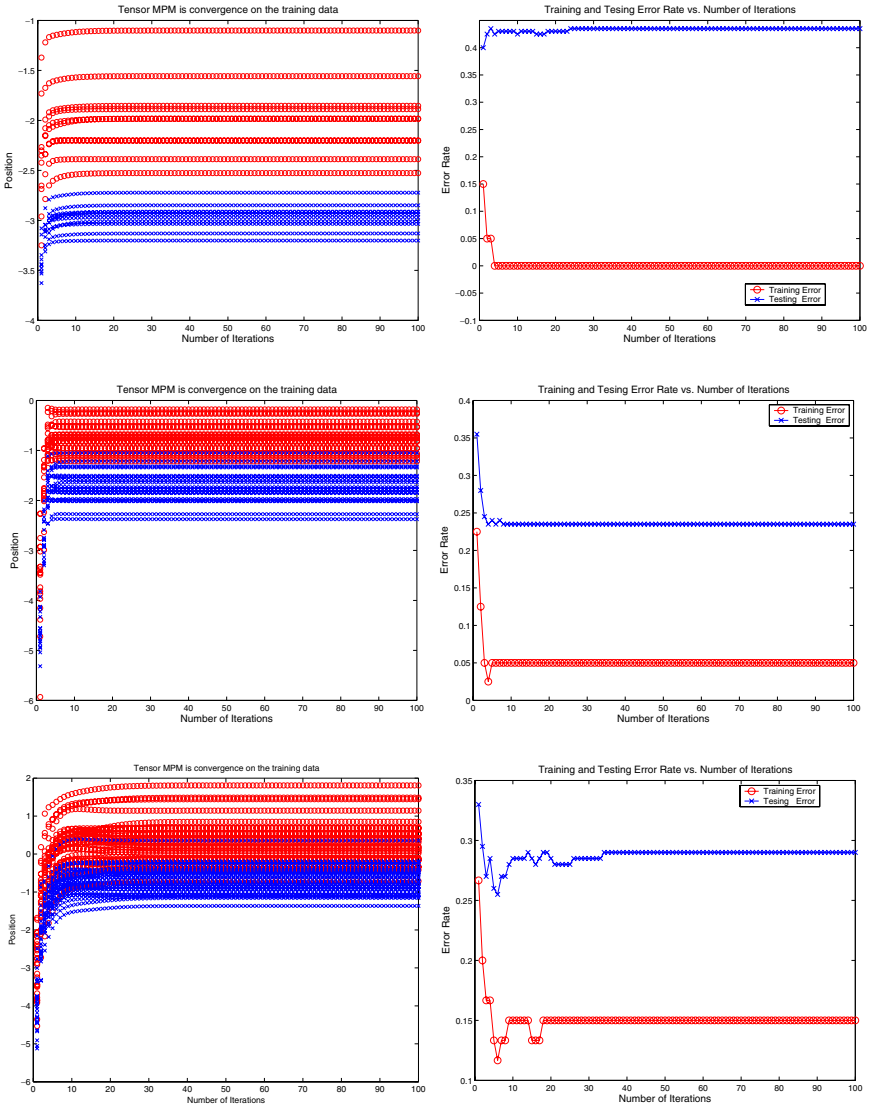


Fig. 17 TMPM converges effectively

Based on all subfigures in Fig. 17, it can be found that the training error and the testing error are converged at some stable values. Therefore, upon these observations, we also empirically justified the convergence of the alternating projection method for TMPM. The theoretical proof is given in Theorem 1.

Many learning algorithms converge at different destinations with different initial parameter values. This is the so-called local minimal problem. However, the developed TMPM does not slump into this local minimal problem, which is demonstrated by a set of experiments (with 100 different initial parameters, 10 learning iterations, and 20 training measurements). From the theoretical view, the

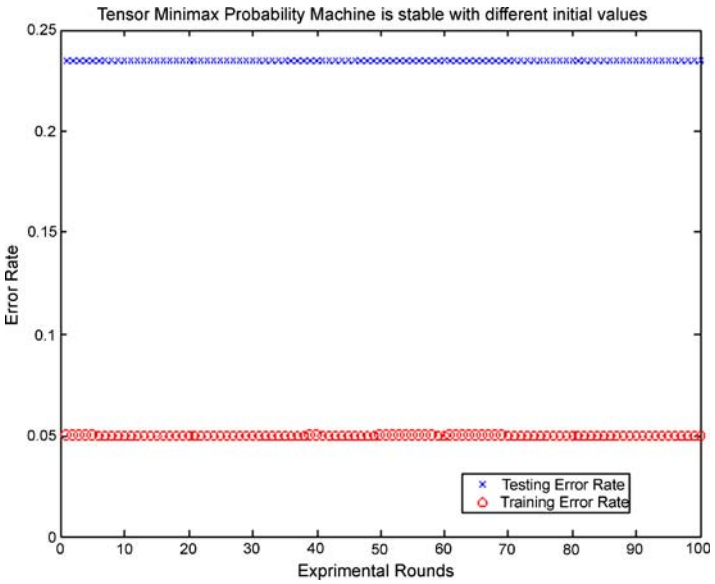


Fig. 18 TMPM is stable with different initial values

TMPM is also a convex optimization problem, so the solution to TMPM is unique. Figure 18 shows this point. The training error rates and the testing error rates are always 0.05 and 0.235, respectively.

9 Conclusion

In this paper, the vector-based learning is extended to accept tensors as input. The result is the supervised tensor learning (STL) framework, which is the multilinear extension of the convex optimization-based learning. To obtain the solution of an STL-based learning algorithm, the alternating projection method is designed.

Based on STL and its alternating projection optimization algorithm, we illustrate several examples. That is, we extend the soft-margin support vector machines (SVM), the ν-SVM, the least squares SVM, the minimax probability machine (MPM), the Fisher discriminant analysis (FDA), the distance metric learning (DML) to their tensor versions, which are the soft-margin support tensor machine (STM), the ν-STM, the least squares STM, the tensor MPM (TMPM), the tensor FDA (TFDA), and the multiple distance metrices learning (MDML). Based on STL, we also introduce a method for feature extraction through an iterative way, i.e., we develop the iterative feature extraction model (IFEM). Finally, we implement TMPM for image classification. By comparing TMPM with MPM, we know TMPM reduces the overfitting problem in MPM.

Acknowledgements We thank the editors and anonymous reviewers for their very useful comments and suggestions.

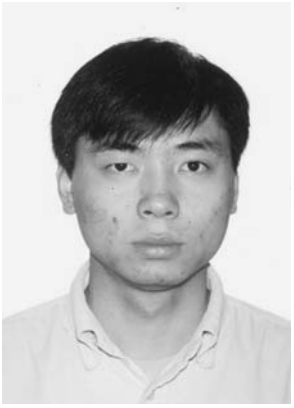
References

1. Amini R, Gallinari P (2005) Semi-supervised learning with an imperfect supervisor. Knowl Inf Syst 8(4):385–413
2. Bartlett P, Shawe-Taylor J (1998) Generalization performance of support vector machines and other pattern classifiers. In: Scholkopf B, Burges CJ, Smola AJ (eds) Advances in kernel methods—support vector learning. MIT Press, Cambridge, MA
3. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge, UK
4. Boyd S, Kim SJ, Vandenberghe L, Hassibi A (2006) A tutorial on geometric programming. Optim Eng
5. Burges JC (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov 2(2):121–167
6. Duda RO, Hart PE, Stork DG (2001) Pattern classification. Wiley, New York
7. Etemad K, Chellappa R (1998) Discriminant analysis for recognition of human face images. J Opt Soc Am A 14(8):1,724–1,733
8. Fisher RA (1938) The statistical utilization of multiple measurements. Ann Eugenics 8:376–386
9. Fukunaga K (1990) Introduction to statistical pattern recognition, 2nd edn. Academic, New York
10. Fung G, Mangasarian OL (2001) Proximal support vector machine classifiers. In: Proceedings of the 7th ACM SIGKDD international conference on knowledge discovery and data mining, San Francisco, CA, pp 77–86
11. Girgensohn A, Foote J (1999) Video classification using transform coefficients. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, vol 6, pp 3045–3048
12. Itti L, Koch C, Niebur E (1998) A model of saliency-based visual attention for rapid scene analysis. IEEE Trans Pattern Anal Mach Intell 20(11):1,254–1,259
13. Itti L, Koch C (2001) Computational modeling of visual attention. Nat Rev Neurosci 2(3):194–203
14. Kim SJ, Magnani A, Boyd S (2005) Robust Fisher discriminant analysis. In: Advances in neural information processing systems. Vancouver and Whistler, British Columbia, Canada
15. Lanckriet G, Cristianini N, Bartlett P, Ghaoui L, Jordan M (2004) Learning the kernel matrix with semidefinite programming. J Mach Learn Res 5:27–72
16. Lanckriet G, Ghaoui L, Bhattacharyya C, Jordan M (2002) A robust minimax approach to classification. J Mach Learn Res 3:555–582
17. Lathauwer LD (1997) Signal processing based on multilinear algebra. Ph.D. Thesis, Katholike Universiteit Leuven, Leuven, Belgium
18. Li T, Ogihara M (2005) Semisupervised learning from different information sources. Knowl Inf Syst 7(3):289–309
19. Lobo M, Vandenberghe L, Boyd S, Lebret H (1998) Applications of second-order cone programming. Linear Algebr Appl 284:193–228
20. Marshall A, Olkin I (1960) Multivariate Chebyshev inequalities. Ann Math Stat 31(4):1,001–1,014
21. Nocedal J, Wright SJ (1999) Numerical optimization. Springer, Berlin
22. Pedroso JP, Murata N (1999) Support vector machines for linear programming: motivation and formulations. BSIS Technical Report 99-2, Riken Brain Science Institute, Wako-shi, Saitama, Japan
23. Popescu I, Bertsimas D (2000) Optimal inequalities in probability theory: a convex optimization approach. Technique Report TM62, Insead
24. Prasad BG, Biswas KK, Gupta SK (2004) Region-based image retrieval using integrated color, shape, and location index. Comput Vis Image Underst 94(1–3):192–233
25. Rui Y, Huang TS, Chang SE (1999) Image retrieval: Current techniques, promising directions and open issues. J Vis Commun Image Represent 10:39–62
26. Salmenkivi M, Mannila H (2005) Using Markov chain Monte Carlo and dynamic programming for event sequence data. Knowl Inf Syst 7(3):267–288
27. Scholkopf B, Smola A, Williamson RC, Bartlett PL (2000) New support vector algorithms. Neural Comput 12:1,207–1,245

28. Scholkopf B, Smola A (2001) Learning with kernels: support vector machines, regularization, optimization, and beyond (Adaptive computation and machine learning). MIT Press, Hawai, Cambridge, MA
29. Shashua A, Levin A (2001) Linear image coding for regression and classification using the tensor-rank principle. In: Proceedings of the IEEE international conference on computer vision and pattern recognition, Hawai, vol 1, pp 42–49
30. Smola A, Friess TT, Scholkopf B (1999) Semiparametric support vector and linear programming machines. Neural Inf Process Syst 11:585–591
31. Strohmann TR, Belitski A, Grudic GZ, DeCoste D (2003) Sparse greedy minimax probability machine classification. In: Advances in neural information processing systems. Vancouver and Whistler, British Columbia, Canada
32. Sun Y, Fisher R (2003) Object-based visual attention for computer vision. Artif Intell 146(1):77–123
33. Sun J, Tao D, Faloutsosy C (2006) Beyond streams and graphs: dynamic tensor analysis. In: Proceedings of the ACM SIGKDD international conference on knowledge discovery and data mining, Philadelphia, PA, USA
34. Suykens JAK, Vandewalle J (1999) Least squares support vector machine classifiers. Neural Process Lett 9(3):293–300
35. Suykens JAK, van Gestel T, De Brabanter J, De Moor B, Vandewalle J (2002) Least squares support vector machines. World Scientific, Singapore
36. Tao D, Li X, Hu W, Maybank SJ, Wu X (2005) Supervised tensor learning. In: Proceedings of the IEEE international conference on data mining, Houston, Texas, USA, pp 450–457
37. Tao D (2006) Discriminative linear and multilinear subspace methods. PhD Thesis, University of London, London
38. Tao D, Li X, Wu X, Maybank SJ (2006) Human carrying status in visual surveillance. In: Proceedings of the IEEE international conference on computer vision and pattern recognition, New York, NY, USA, pp 1,670–1,677
39. Tao D, Li X, Wu X, Maybank SJ (2006) Elapsed time in human gait recognition: a new approach. In: Proceedings of the IEEE international conference on acoustics, speech, and signal processing, Toulouse, France
40. Tao D, Li X, Maybank SJ (2007) Negative samples analysis in relevance feedback. IEEE Trans Knowl Data Eng
41. Torralba AB, Oliva A (1999) Semantic organization of scenes using discriminant structural templates. In: Proceedings of the IEEE international conference on computer vision, Kerkyra, Greece, pp 1,253–1,258
42. Treisman AM, Gelade G (1980) A feature-integration theory of attention. Cogn Psychol 12(1):97–136
43. Vandenberghe L, Boyd S (1996) Semidefinite programming. SIAM Rev 1(38):49–95
44. Vanderbei R (2001) Linear programming: foundations and extensions, 2nd edn. Springer, Berlin
45. Vapnik V (1995) The nature of statistical learning theory. Springer-Verlag, New York
46. Vasilescu MAO, Terzopoulos D (2003) Multilinear subspace analysis for image ensembles. In: Proceedings of the IEEE international conference on computer vision and pattern recognition, vol 2, Madison, WI, pp 93–99
47. Wang JZ, Li L, Wiederhold G (2001) SIMPLIcity: semantics-sensitive integrated matching for picture libraries. IEEE Trans Pattern Anal Mach Intell 23(9):947–963
48. Wechslet H, Phillips J, Bruse V, Soulie F, Hauhg T (eds) (1998) Face recognition: from theory to application. Springer-Verlag, Berlin
49. Weinberger KQ, Blitzer J, Saul LK (2005) Distance metric learning for large margin nearest neighbor classification. Neural Inf Process Syst 18:
50. Winston WL, Goldberg JB, Venkataramanan M (2002) Introduction to mathematical programming: operations research, 4th edn. Duxbury, Pacific Grove, CA, USA
51. Xu D, Yan S, Zhang L, Zhang H-J, Liu Z, Shum H-Y (2005) Concurrent subspaces analysis. In: Proceedings of the IEEE international conference on computer vision and pattern recognition, San Diego, CA, USA, vol 2, pp 203–208
52. Ye J, Janardan R, Li Q (2005) Two-dimensional linear discriminant analysis. In: Advances in neural information processing systems. Vancouver and Whistler, British Columbia, Canada, pp 1,569–1,576

53. Ye J, Li Q (2005) A two-stage linear discriminant analysis via QR-decomposition. IEEE Trans Pattern Anal Mach Intell 27(6):929–941
54. Zangwill WI (1969) Nonlinear programming: a unified approach. Prentice-Hall, Englewood Cliffs, NJ
55. Zhang X (2004) Matrix analysis and applications. Springer, Berlin

Author Biographies



Dacheng Tao received the B.Eng. degree from the University of Science and Technology of China (USTC), the MPhil degree from the Chinese University of Hong Kong (CUHK) and the PhD from the University of London (Birkbeck). He will join the Department of Computing in the Hong Kong Polytechnic University as an assistant professor. His research interests include biometric research, discriminant analysis, support vector machine, convex optimization for machine learning, multilinear algebra, multimedia information retrieval, data mining, and video surveillance. He published extensively at TPAMI, TKDE, TIP, TMM, TCSVT, CVPR, ICDM, ICASSP, ICIP, ICME, ACM Multimedia, ACM KDD, etc. He gained several Meritorious Awards from the Int'l Interdisciplinary Contest in Modeling, which is the highest level mathematical modeling contest in the world, organized by COMAP. He is a guest editor for special issues of the Int'l Journal of Image and Graphics (World Scientific) and the Neurocomputing (Elsevier).

Xuelong Li works at the University of London. He has published in journals (IEEE T-PAMI, T-CSVT, T-IP, T-KDE, TMM, etc.) and conferences (IEEE CVPR, ICASSP, ICDM, etc.). He is an Associate Editor of IEEE T-SMC, Part C, Neurocomputing, IJIG (World Scientific), and Pattern Recognition (Elsevier). He is also an Editor Board Member of IJITDM (World Scientific) and ELCVIA (CVC Press). He is a Guest Editor for special issues of IJCM (Taylor and Francis), IJIG (World Scientific), and Neurocomputing (Elsevier). He co-chaired the 5th Annual UK Workshop on Computational Intelligence and the 6th the IEEE Int'l Conf. on Machine Learning and Cybernetics. He was also a publicity chair of the 7th IEEE Int'l Conf. on Data Mining and the 4th Int'l Conf. on Image and Graphics. He has been on the program committees of more than 50 conferences and workshops.

Xindong Wu is a Professor and the Chair of the Department of Computer Science at the University of Vermont. He holds a Ph.D. in Artificial Intelligence from the University of Edinburgh, Britain. His research interests include data mining, knowledge-based systems, and Web information exploration. He has published extensively in these areas in various journals and conferences, including IEEE TKDE, TPAMI, ACM TOIS, IJCAI, AAAI, ICML, KDD, ICDM, and WWW, as well as 12 books and conference proceedings. Dr. Wu is the Editor-in-Chief of the IEEE Transactions on Knowledge and Data Engineering (by the IEEE Computer Society), the Founder and current Steering Committee Chair of the IEEE International Conference on Data Mining (ICDM), an Honorary Editor-in-Chief of Knowledge and Information Systems (by Springer), and a Series Editor of the Springer Book Series on Advanced Information and Knowledge Processing (AIKP). He is the 2004 ACM SIGKDD Service Award winner.

Weiming Hu received the Ph.D. degree from the Department of Computer Science and Engineering, Zhejiang University. From April 1998 to March 2000, he was a Postdoctoral Research Fellow with the Institute of Computer Science and Technology, Founder Research and Design Center, Peking University. Since April 1998, he has been with the National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences. Now he is a Professor and a Ph.D. Student Supervisor in the laboratory. His research interests are in visual surveillance, neural networks, filtering of Internet objectionable information, retrieval of multimedia, and understanding of Internet behaviors. He has published more than 80 papers on national and international journals, and international conferences.

Stephen J. Maybank received a BA in Mathematics from Kings college, Cambridge in 1976 and a PhD in Computer Science from Birkbeck College, University of London in 1988. He was a research scientist at GEC from 1980 to 1995, first at MCCS, Frimley and then, from 1989, at the GEC Marconi Hirst Research Centre in London. In 1995 he became a lecturer in the Department of Computer Science at the University of Reading and in 2004 he became a professor in the School of Computer Science and Information Systems at Birkbeck College, University of London. His research interests include camera calibration, visual surveillance, tracking, filtering, applications of projective geometry to computer vision and applications of probability, statistics and information theory to computer vision. He is the author of more than 90 scientific publications and one book. He is a Fellow of the Institute of Mathematics and its Applications, a Fellow of the Royal Statistical Society and a Senior Member of the IEEE. For further information see http://www.dcs.bbk.ac.uk/ sjmaybank.