Multimedia Systems
DOI 10.1007/s00530-015-0464-7

REGULAR PAPER

Tucker decomposition-based tensor learning for human action recognition

Jianguang Zhang[1,2] · Yahong Han[1,3] · Jianmin Jiang[4]

Received: 3 September 2014 / Accepted: 25 March 2015

Abstract The spatial information is the important cue for human action recognition. Different from the vector representation, the spatial structure of human action in the still images can be preserved by the tensor representation. This paper proposes a robust human action recognition algorithm by tensor representation and Tucker decomposition. In this method, the still image containing human action is represented by a tensor descriptor (Histograms of Oriented Gradients). This representation preserves the spatial information inside the human action. Based on this representation, the unknown tensor parameter is decomposed according to the Tucker tensor decomposition at first, and then the optimization problems can be solved using the alternative optimization method, where at each iteration, the tensor descriptor is projected along one order and the parameter along the corresponding order can be estimated by solving the Ridge Regression problem. The estimated tensor parameter is more discriminative because of effectively using the spacial information along each order. Experiments are conducted using action images from three publicly available databases. Experimental results demonstrate that our method outperforms other methods.

Keywords Tucker decomposition · Histograms of oriented gradients · Action recognition

1 Introduction

Recognition of human actions is a challenging task in computer vision. It is of significant interest in many applications, such as image understanding [1], image annotation [2] and image retrieval [3]. As important cues, spatial information has been shown to be effective for recognizing human actions [4]. The significant spatial information of human action is the shape appearance of human body, e.g., the position of arm in “tennis-serve” is often higher than that in “tennis-forehand”. Similarly, the position of leg in “kicking” is often higher than that in “running”. This spatial structure information is included in the natural representations of visual data. The tensor representation can preserve the spatial information because the tensor can be regarded as natural representations of visual data [5].

Recently there have been more action recognition methods than before using tensor feature descriptors, such as Histograms of Oriented Gradients (HOG) [6] and log-Gabor [7]. HOG is able to characterize the local appearance and shape on still image pretty well by the distribution of local intensity gradients. Due to its robustness, HOG has been successfully applied to the problem of action recognition [8–10]. Therefore, we characterize action image by HOG descriptor in this paper. The basic idea of HOG is to

Communicated by B. Prabhakaran.

✉ Yahong Han
yahong@tju.edu.cn

Jianguang Zhang
lynxzjg@tju.edu.cn

Jianmin Jiang
jianmin.jiang@szu.edu.cn

[1] School of Computer Science and Technology, Tianjin University, Tianjin, China

[2] Department of Mathematics and Computer Science, Hengshui University, Hengshui, China

[3] Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin University, Tianjin, China

[4] School of Computer Science and Software Engineering, Shenzhen University, Shenzhen, China

Published online: 08 April 2015

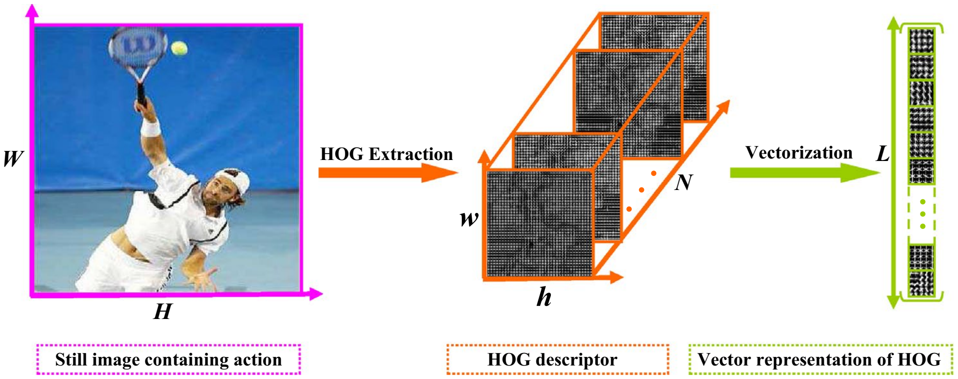


Fig. 1 Examples of still image represented as HOG descriptor (3-order tensor) and its vectorization form. To obtain the HOG descriptor , we first resize the given image to $W \times H$ and then divide the resized image into spatial blocks of equal size as $w \times h$. Within each spatial block, a set of histograms of gradient orientations are calculated. The resulting histograms are normalized by 4 different normalizations. Therefore, dimension of the third order N is 4 times expansion of the number of orientations. Finally, each histogram is organized in a 3-order tensor of dimension $w \times h \times N$. When HOG is reformulated as a vector, the final feature is L-dimensional vector, where $L = w \times h \times N$

compute gradient orientation histograms on a dense grid of uniformly spatial blocks and perform local contrast normalization [6]. As shown in Fig. 1, the HOG descriptor was normally organized in a 3-order tensor representation [10]. How to process this 3-order data is one of the most important topics for both image processing and machine learning.

In the past decades, numerous vector-based algorithms have been proposed, such as K-Nearest Neighbors (KNN) [11], Support Vector Machine (SVM) [12] and Ridge Regression (RR) [13]. These algorithms assume that the inputs are all vectors. When they are utilized to handle the tensor data, the tensor data need to be transformed into vector data at first. One common way is stacking elements of tensor data in a more or less arbitrary order. For instance, as shown in the Fig. 1, the HOG descriptor of size $w \times h \times N$ is reformulated as a $L \times 1$ vector and then the vector-based algorithm can be utilized. Although the performances of vector-based approaches are prominent in many cases, they may create many issues in processing tensor data. First, when tensor data are reformulated as a vector, we often obtain a vector of potentially very high dimensionality. For example, for a tensor data of size $32 \times 32 \times 32$, the reformulated vector is 32768 dimensional. With the increase of dimensionality, we may confront some problems, such as overfitting, or large memory requirements. Second, when a tensor is expanded as a vector, the spatial structure and correlation in shape appearance are disregarded. For instance, as shown in Fig. 1, in each $w \times h$ spaced block of HOG descriptor, we can visualize the shape appearance along each orientation. On the other hand, from all spaced blocks of HOG descriptor, we can obtain the spatial relationship among all orientations. However, these spatial information cannot be found from the vector representation of HOG.

To solve these problems, several algorithms that used tensor representations have been recently proposed. In [14], the face image is considered as 2-order tensor for face recognition. Other traditional vector-based representation subspace learning methods have been extended to tensor representation, such as Tensor Principal Component Analysis [15], Tensor Linear Discriminant Analysis [16] and Multi-linear Discriminant Analysis [17]. However, we cannot directly utilize these methods for action recognition because their purposes are to learn a subspace of tensor data and then employ another vector-based classifier to process images. In [18], the images are represented as 2-order tensors and directly used to learn two groups of classification vectors. The 2-order tensor represents an image in its natural tensor form. Thus, this representation can preserve the spatial correlation of an image and avoid the problems about high dimensionality. However, we cannot handle the 3-order or higher order descriptor (i.e., HOG) of image using this method. In [19], a supervised tensor learning (STL) framework is presented, in which the higher order tensor data can be directly processed. According to rank-1 tensor [20], tensor parameter is decomposed into one factor vector along one order. Therefore, the spatial information along each order of tensor input can be used to estimate the factor vectors. However, the use of only one factor vector for each order may lead to loss of discriminative information.

In [5], the HOG descriptor is viewed as original 3-order tensor representation. A tensor regression learning (TRL) frame is proposed to process the 3-order HOG tensor directly, in which the parameter is decomposed into N rank-1 tensors using the CANDECOMP/PARAFAC (CP) decomposition [21]. N parameter vectors along all orders are estimated using the spatial information along each order of tensor input. After CP decomposition, the rank of tensor is N, which is also the number of rank-1 tensors. An important issue of tensor CP decomposition is that the rank N cannot be confirmed. If there are too many rank-1 tensors, the included information may be noise and redundant, otherwise this representation is incomplete. So it is difficult for tensor CP decommission to effectively use the discriminative spatial information along each order. Another decomposition strategy is the Tucker decomposition [22], which is considered as higher order Principal Component Analysis. For the Tucker decomposition, each tensor is represented as the product of a core tensor and factor matrices along all orders. There are two advantages in Tucker decomposition. First, compared with the CP decomposition that needs to evaluate the rank to approximate the initial tensor, we can obtain the more exact decomposition result of tensor using Tucker decomposition. The other benefit is that we can achieve the goal of spatial information selection along each order by adjusting the dimension of the core tensor.

To make full use of the spatial information by treating HOG as itself without vectorization, we propose a new tensor learning frame for processing 3-order HOG tensor directly. In the proposed frame, the HOG descriptor and parameter are regarded as tensor form. Motivated by the advantages of tensor Tucker decomposition, we decompose the 3-order parameter tensor into a core tensor multiplied by matrices along 3 orders at first. Then in each round of the alternating optimization algorithm utilized in this paper, the core tensor is transformed into the core matrix along one order and the decomposed matrix associated with this order can be estimated by solving the Ridge Regression (RR) problem while fixing other and core matrix. After solving for matrices along 3 orders, we can compute the core tensor using the same process. At each round, we select the discriminative spatial information along one order by adjusting the dimension of the core tensor. The procedure is repeated until convergence. The proposed frame can efficiently explore the spatial information of HOG tensor during the learning process. The recognition performance of proposed method is thereby enhanced subsequently.

We name the proposed method Tucker Ridge Regression (TuRR). The contributions of this paper are as follows:

1. We utilize the 3-order HOG descriptor to represent the human action in still image and propose a tensor learning framework to directly process this descriptor. Unlike vector-based regression methods, we can effectively preserve the spatial information of human action by tensor representation.
2. To fully exploit the discriminative spatial information of the HOG tensor, we propose the Tucker decomposition to decompose the parameter tensor, which is more robust than that using the CP decomposition of parameter tensor. Moreover, the efficiency is guaranteed by avoiding the generation of high-dimensional vectors.

The rest of the paper is organized as follows. Section 2 covers some preliminaries including relative notation, basic definitions and a brief review of Tucker tensor decomposition. In Sect. 3, we introduce the Tucker Ridge Regression that are able to directly handle tensor representation of HOG descriptor. The efficiency of the proposed algorithm is demonstrated on three action image databases in Sect. 4. Finally, conclusion are drawn in Sect. 5, followed by acknowledgments respectively.

2 Notations and preliminaries

In this section, we will briefly describe some useful notations and concepts of tensorial algebra that will be used throughout this paper and that are consistent with those presented in [20]. A tensor is a multidimensional array. Vector and matrix can be considered as 1-order and 2-order tensors, respectively. In this paper, scalars, vectors, matrices and tensors are denoted by lowercase letters (e.g., x), boldface lowercase letters (e.g., $\mathbf{x}$), capital letters (e.g., X) and Euler script calligraphic letters (e.g., $\mathcal{X}$), respectively.

We denote the ith element of a vector $\mathbf{x}$ as x_i, the (i, j)th element of a matrix X as $x_{i,j}$ and the jth column of a matrix X as x_j. In a similar way, we denote the elements of the M-order tensor $\mathcal{X} \in IR^{I_1 \times I_2 \times \ldots I_M}$ as $x_{i_1 i_2 \ldots i_M}$, $i_l = 1, 2, \ldots, I_l$, $l = 1, 2, \ldots, M$.

The matricization of a tensor is the process of transforming a tensor into a matrix by reordering the elements of a tensor into a matrix. In this paper, we consider only the useful special case of order-n matricization. The order-n matricization of an M-order tensor $\mathcal{X} \in IR^{I_1 \times I_2 \ldots \times I_M}$, denoted by $X_{(n)} \in IR^{I_n \times (I_1 \ldots I_{n-1} I_{n+1} \ldots I_M)}$ arranges the nth order fibers to be the columns of the resulting matrix. Tensor element $x_{(i_1, i_2, \ldots, i_M)}$ maps to matrix element $x_{(i_n, j)}$, where $j = 1 + \sum_{k=1, k \neq n}^{M} (i_k - 1) J_k$ with $J_k = \prod_{l=1, l \neq n}^{k-1} I_l$. Similarly, the vectorization of a matrix X is to stack its elements into a vector, which is denoted by vec(X).

The inner product of two tensors of the same size $\mathcal{X}, \mathcal{Y} \in IR^{I_1 \times I_2 \ldots \times I_M}$ is the sum of the products of their elements, defined by

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

From the tensor matricization equivalents, we have

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \langle X_{(j)}, Y_{(j)} \rangle = \text{trace}\left(X_{(j)} Y_{(j)}^T\right) = \text{vec}\left(X_{(j)}\right)^T \text{vec}\left(Y_{(j)}\right) \tag{2}$$

The order-lproduct between a tensor $\mathcal{X} \in IR^{I_1 \times I_2 \times \cdots \times I_M}$ and a matrix $U \in IR^{J \times I_l}$ is an important operation denoted by $\mathcal{X} \times_l U$ which yields a tensor $\mathcal{Y}$ of size $I_1 \times \cdots \times I_{l-1} \times J \times I_{l+1} \times \cdots \times I_M$ having as elements:

$$y_{i_1 \ldots, i_{l-1} j i_{l+1} \ldots i_M} = \sum_{i_l=1}^{I_l} x_{i_1 i_2 \ldots i_M} u_{j i_l} \tag{3}$$

The Kronecker product of two matrices $A \in IR^{I \times J}$ and $B \in IR^{K \times L}$, denoted by $A \otimes B$, is defined as:

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

The CP tensor decomposition factorizes an M-order tensor $\mathcal{W} \in IR^{I_1 \times I_2 \times \cdots \times I_M}$ into a sum of N rank-1 tensors, denoted as:

$$\mathcal{W} \approx \sum_{n=1}^{N} u_n^1 \circ u_n^2 \circ \ldots u_n^M \triangleq \left[\left[U^1, U^2, \ldots U^M\right]\right] \tag{4}$$

where $\circ$ is the outer product of vectors, the matrices $U^k = [u_1^k, \ldots, u_N^k] \in IR^{I_k \times N}, \quad k = 1, 2, \ldots, M.$ Because the rank N cannot be determined, we only obtain the approximate decomposition result using CP tensor decomposition.

The Tucker tensor decomposition factorizes an M-order tensor $\mathcal{W} \in IR^{I_1 \times I_2 \times \cdots \times I_M}$ into the multiplication of a core tensor with a set of factor matrices along all orders, denotes as:

$$\mathcal{W} = \mathcal{G} \times_1 U_1 \times_2 U_2 \ldots \times_M U_M = [\![\mathcal{G}; U_1, U_2, \ldots, U_M]\!] \tag{5}$$

where $\mathcal{G} \in IR^{R_1 \times R_2 \ldots \times R_M}$ is a core tensor and $\{U_i \in IR^{I_i \times R_i}\}_{i=1}^M$ is a set of factor matrices which are multiplied to the core tensor $\mathcal{G}$ along each order. According to [20], $R_i \leq d_i$ for all $i = 1, 2, \ldots, M$. Compared with the CP decomposition, Tucker decomposition does not need to pre-evaluate the rank N. So we can obtain more accurate decomposition result. It is noted that we can control the dimension of the factor matrix U_i by adjusting the dimension R_i along the ith order of core tensor $\mathcal{G}$ and then select discriminative spatial information to enhance the performance of action recognition.

The Kronecker product of M factor matrices is defined as

$$U_\otimes = U_M \otimes \cdots \otimes U_1 \tag{6}$$

Similarly, the Kronecker product of $M - 1$ factor matrices (skipping the k-th matrix $\widetilde{U}_k$) is given by

$$\widetilde{U}_k = U_M \otimes \cdots \otimes U_{k+1} \otimes U_{k-1} \otimes \cdots \otimes U_1 \tag{7}$$

Therefore, the matricization form of Eq. (5) is

$$\begin{aligned} W_k &= U_k G_k (U_M \otimes \cdots \otimes U_{k+1} \otimes U_{k-1} \otimes \cdots \otimes U_1)^T \\ &= U_k G_k \widetilde{U}_k^T \end{aligned} \tag{8}$$

and the vectorized form of Eq. (8) is

$$\text{vec}(W_k) = U_\otimes \text{vec}(G_k) \tag{9}$$

where G_k is the matricization form of tensor $\mathcal{G}$ along the k th order.

3 The proposed algorithm

In this section, we present the objective function of Tucker Ridge Regression (TuRR) followed by a detailed optimization method for investigating the solution.

3.1 Tucker ridge regression

When the appropriate regularization is added, the least squares loss function gains comparable performance to other complicated loss functions [23]. Therefore, we utilize the least squares loss function for the problem of regression. When the ℓ_2-norm is used for regularization in the vector space, the typical ridge regression can be formulated as:

$$\min_{\mathbf{w}, b} \sum_{i=1}^{n} \left(\langle \mathbf{x}_i, \mathbf{w} \rangle - y_i + b \right)^2 + \lambda \|\mathbf{w}\|_2^2 \tag{10}$$

where $\mathbf{x}_i$ is the vector representation of image descriptor, $\mathbf{w}$ is the parameter vector, b is the bias, and y_i is the regression output of $\mathbf{x}_i$.

To extend the ridge regression from vector to tensor space, let $\mathcal{X} = [\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n] \in IR^{d_1 \times d_2 \times \ldots d_M \times n}$ as the set of training data where the i-th image descriptor $\mathcal{X}_i$ is an M-order tensor and n is the total number of the training samples. We denote the associated class label vectors as $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T \in \{0, 1\}^{n \times 1}$. $y_i = 1$ if the i-th data is a positive example whereas $y_i = 0$ otherwise. The tensor Ridge Regression can be written as:

$$\min_{\mathcal{W}, b} \sum_{i=1}^{n} (\langle \mathcal{X}, \mathcal{W} \rangle - y_i + b)^2 + \lambda \|\mathcal{W}\|_F^2 \tag{11}$$

where $\mathcal{W} \in IR^{d_1 \times d_2 \ldots \times d_M}$ is the parameter tensor, λ denotes the regularization parameter and b is the bias term. We focus on learning the parameter tensor $\mathcal{W}$ and bias term b.

In this paper, in order to capture the underlying structure of HOG tensor, the parameter tensor $\mathcal{W}$ is decomposed according to the Tucker tensor decomposition in Eq. (5) firstly. Therefore, the minimization problem defined in Eq. (11) is rewritten as:

$$\begin{aligned} &\min_{\mathcal{G}, U_1, \ldots, U_M, b} \sum_{i=1}^{n} (\langle \mathcal{X}_i, [\![\mathcal{G}; U_1, U_2, \ldots, U_M]\!] \rangle - y_i + b)^2 \\ &\quad + \lambda \| [\![\mathcal{G}; U_1, U_2, \ldots, U_M]\!] \|_F^2 \end{aligned} \tag{12}$$

Compared with traditional ridge regression, using tensor representation and Tucker tensor decomposition enable our method to select and utilize the discriminative spacial information along each order.

3.2 Optimizing the objective function

We present our solution for obtaining the action recognizer based on tensor descriptor. The objective function in Eq. (12) is not jointly convex for all items of $\mathcal{W}$ after Tucker decomposition. In order to solve the problem, we follow a alternating optimization algorithm, where at each iteration, the convex optimization problem with respect to one item of the parameters is solved, while all the other parameters are kept fixed.

In order to obtain U_k, the tensor $\mathcal{W}$, $\mathcal{G}$ and $\mathcal{X}_i$ should be unfolded along the kth-order, the order-k matricization of $\mathcal{W}$, $\mathcal{G}$ and $\mathcal{X}_i$ are W_k, G_k and X_i^k, respectively. Therefore, the matrix representation along k-order of Eq. (11) can be rewritten as

$$\min_{W_k, b} \sum_{i=1}^{n} \left(Tr\left(W_k {X_i^k}^T \right) - y_i + b \right)^2 + \lambda \| W_k \|_F^2 \tag{13}$$

Substituting W_k in Eq. (13) with Eq. (8), and fixing $U_l|_{l=1, l \neq k}^{M}, G_k$, it becomes:

$$\begin{aligned} &\min_{U_k, b} \sum_{i=1}^{n} \left(Tr\left(U_k G_k \widetilde{U}_k^T {X_i^k}^T \right) - y_i + b \right)^2 \\ &\quad + \lambda Tr\left(U_k G_k \widetilde{U}_k^T \widetilde{U}_k G_k^T U_k^T \right) \end{aligned} \tag{14}$$

We set $\widetilde{X}_i^{kT} = G_k \widetilde{U}_k^T {X_i^k}^T$ and $D_k = G_k \widetilde{U}_k^T \widetilde{U}_k G_k^T$, then Eq. (14) is rewritten as

$$\min_{U_k, b} \sum_{i=1}^{n} (Tr(U_k \widetilde{X}_i^{kT}) - y_i + b)^2 + \lambda Tr(U_k D_k U_k^T) \tag{15}$$

Let us define $Tr(U_k \widetilde{X}_i^{kT}) = [\text{vec}(U_k)^T \; b][\text{vec}(\widetilde{X}_i^k)^T \; 1]^T = \mathbf{v}_k^T \widehat{\mathbf{x}}_i$ and $\widetilde{D}_k = \begin{bmatrix} D_k \otimes I_{d_k \times d_k} & 0 \\ 0 & 1 \end{bmatrix}$. Then, Eq. (15) can be written as

$$\min_{\mathbf{v}_k} \sum_{i=1}^{n} \left(\mathbf{v}_k^T \widehat{\mathbf{x}}_i - y_i \right)^2 + \lambda Tr\left(\mathbf{v}_k^T \widetilde{D}_k \mathbf{v}_k \right) \tag{16}$$

By letting $\widehat{X} = [\widehat{\mathbf{x}}_1, \widehat{\mathbf{x}}_2, \ldots, \widehat{\mathbf{x}}_n]$ and $\mathbf{y} = [y_1, y_2, \ldots, y_n]$, Eq. (16) is rewritten as

$$\min_{\mathbf{v}_k} Tr\left(\mathbf{v}_k^T \left(\widehat{X} \widehat{X}^T + \lambda \widetilde{D}_k \right) \mathbf{v}_k \right) - 2Tr\left(\mathbf{v}_k^T \widehat{X} \mathbf{y}^T \right) + Tr\left(\mathbf{y}^T \mathbf{y} \right) \tag{17}$$

Thus, the optimization problem for U_k, b in Eq. (14) is formulated as a Ridge Regression problem with respect to $\mathbf{v}_k$. Setting the derivative of Eq. (17) w.r.t. $\mathbf{v}_k$ to 0, we have:

$$\begin{aligned} &2\left(\widehat{X} \widehat{X}^T + \lambda \widetilde{D}_k \right) \mathbf{v}_k - 2\widehat{X} \mathbf{y}^T = 0 \\ &\Rightarrow \mathbf{v}_k = \left(\widehat{X} \widehat{X}^T + \lambda \widetilde{D}_k \right)^{-1} \widehat{X} \mathbf{y}^T \end{aligned} \tag{18}$$

After obtaining the closed form solution of $\{U_1, U_2, \ldots, U_M\}$, because the core tensor $\mathcal{G}$ can be unfolded along arbitrary order, in this paper, we solve for $\mathcal{G}$ along the 1th order, namely G_1. Therefore, we have $\text{vec}(W_1) = U_{\otimes} \text{vec}(G_1)$. The Eq. (13) can be rewritten as:

$$\begin{aligned} &\min_{G_1, b} \sum_{i=1}^{n} \left(\text{vec}(G_1)^T U_{\otimes}^T \text{vec}(X_i^k) - y_i + b \right)^2 \\ &\quad + \lambda \text{vec}(G_1)^T U_{\otimes}^T U_{\otimes} \text{vec}(G_1) \end{aligned} \tag{19}$$

Similarly, we denote $\bar{\mathbf{x}}_i = [(U_{\otimes} \text{vec}(X_i^k))^T \; 1]$, $\mathbf{g}_1 = [(\text{vec}(G_1))^T \; b]$, $D = \begin{bmatrix} U_{\otimes}^T U_{\otimes} & 0 \\ 0 & 1 \end{bmatrix}$ and $\overline{X} = [\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \ldots, \bar{\mathbf{x}}_n]$. Equation (19) can be represented as ridge regression problem about $\mathbf{g}_1$, as follow

$$\min_{\mathbf{g}_1, b} Tr\left(\mathbf{g}_1^T \left(\overline{X} \, \overline{X}^T + \lambda D \right) \mathbf{g}_1 \right) - 2Tr\left(\left(\mathbf{g}_1^T \overline{X} \mathbf{y}^T \right) + Tr\left(\mathbf{y}^T \mathbf{y} \right) \right) \tag{20}$$

Setting the derivative of Eq. (20) w.r.t. $\mathbf{g}_1$ to 0, it becomes:

$$\begin{aligned} &2\left(\overline{X} \, \overline{X}^T + \lambda D \right) \mathbf{g}_1 - 2\overline{X} \mathbf{y}^T = 0 \\ &\Rightarrow \mathbf{g}_1 = \left(\overline{X} \, \overline{X}^T + \lambda D \right)^{-1} \overline{X} \mathbf{y}^T \end{aligned} \tag{21}$$

The optimization of $\{U_1, U_2, \ldots, U_M; \mathcal{G}\}$ is iterated until convergence. The detail iteration process is given in Algorithm 1.

Once the $U_1, U_2, \ldots, U_M, G_1, b$ for c classes are obtained, we can easily get c groups of recognition parameters $\{U_1^r, U_2^r, \ldots, U_M^r, G_1^r, b^r\}|_{r=1}^{c}$. Then we propose Algorithm 2 to recognize the labels of the testing data.

According to the following proof, we can verify that the proposed iterative approach in Algorithm 1 converges and the global solutions of $U_1, U_2, \ldots, U_M; \mathcal{G}$ are obtained.

Proof By fixing $U_l|_{l=1, l \neq k}^{M}, G_k$, the objective function in Eq. (12) is converted to the problem in Eq. (17). It can be seen that Eq. (17) is a convex optimization problem w.r.t. $\mathbf{v}_k$. Therefore, we can obtain the global solutions for $\mathbf{v}_k$ by setting the derivative of Eq. (17) w.r.t. $\mathbf{v}_k$ to zero. Based on the similar theory, we also prove that by fixing $U_l|_{l=1}^{M}$, we obtain the global solutions for $\mathbf{g}_1$. □

Algorithm 1 Tucker Ridge Regression

```
Input: Input tensors $\mathcal{X}_i, i = 1, ..., n$. $\mathcal{X}_i \in IR^{d_1 \times d_2 \times ... \times d_M}$, and labels $Y \in IR^{c \times n}$ where c is the number of
    action classes. The dimensions set $R_1, R_2, ..., R_M$ of the core tensor.
    Regularization parameters $\lambda$.
Output: The parameters $\{U_1^r, U_2^r, ..., U_M^r, G_1^r, b^r\}_{r=1}^{c}$.
for $r = 1$ to $c$ do
    for $i = 1$ to $n$ do
        if $Y(r,i) = c$, $\mathbf{y}(r,i) = 1$; otherwise, $\mathbf{y}(r,i) = -1$.
    end for
    Set $t = 0$ and initialize factor matrices $U_1^r, U_2^r, ..., U_M^r, G_1^r$ randomly;
    repeat
        $t = t + 1$;
        for $k = 1$ to $M$ do
            Update $\mathbf{v}_k$ according to Eq.(18) ;
        end for
        Update $\mathbf{g}_1$ according to Eq.(21) ;
    until Convergence
end for
return parameter set $\{U_1^r, U_2^r, ..., U_M^r, G_1^r, b^r\}_{r=1}^{c}$ ;
```

Algorithm 2 The recognition process

```
Input: Testing tensor $\mathcal{X}_i, i = 1, ..., n$. $\mathcal{X}_i \in IR^{d_1 \times d_2 \times ... \times d_M}$.
    The parameter set $\{U_1^r, U_2^r, ..., U_M^r, G_1^r, b^r\}|_{r=1}^{c}$ for $c$ classes.
Output: Predicted labels $\mathbf{y}$ of the testing data
for $r = 1$ to $c$ do
    Compute the tensor parameter of the $r$th class according to $\mathcal{W}^r = \mathcal{G}^r \times_1 U_1^r \times_2 U_2^r \cdots \times_M U_M^r$
end for
for $i = 1$ to $n$ do
    Compute the predict label of $\mathcal{X}_i$ as
    $y_i = \arg\max_r (< \mathcal{X}_i, \mathcal{W}^r > + b^r)|_{r=1}^{c}$
end for
return Recognized label $\mathbf{y}$;
```

In this paper, the parameter tensor $\mathcal{W}$ is decomposed as $\{U_1, \ldots, U_k, \ldots, U_M, \mathcal{G}\}$. The dimensions of U_k and $\mathcal{G}$ are $d \times R$ and $\prod_{k=1}^{M} R$, respectively. Because R is much smaller than d in practice, the most time-consuming operation is to solve the ridge regression problem associated with $\mathbf{v}_k$ (i.e., vectorized form of U_k). The complexity of our algorithm is roughly $O((d * R)^3)$.

4 Experiments

In the experiment, we compare Tucker Ridge Regression (TuRR) with several tensor algorithms [5], namely Support Tucker Machines (STuM) [22], higher rank Support Tensor Regression (hrSTR), higher rank Tensor Ridge Regression (hrTRR), optimal-rank Support Tensor Regression (orSTR) and optimal-rank Tensor Ridge Regression (orTRR). STuM adopts Tucker decomposition to decompose parameter tensor, hrTRR, hrSTR, orTRR and orSTR employ CP decomposition to realize parameter tensor decomposition. We also include comparison with the Support Vector Regression (SVR) [24], which needs transform the tensor data into vectorization form at first. The average accuracy over all action categories is chosen as the evaluation metric.

All action images are resized to be of size 128 × 128. We use the probability of boundary (Pb) operator [25] to delineate the object boundaries in action images. Subsequently, each image is divided into non-overlapping spaced blocks of size 16 × 16, and a HOG filter with 4 orientations and 4 normalization schemes is applied at each spaced blocks. Thus, each image was represented as a 16 × 16 × 16 tensor of 3-order. Algorithm SVR deals with 4096 × 1 vectorization form of 3-order HOG descriptor while other algorithms utilize 3-order HOG tensor directly.To fairly compare different recognition algorithms, we use a "grid-research" strategy from $\{10^{-6}, 10^{-5}, \ldots, 10^{5}, 10^{6}\}$ to tune all the parameters for all the algorithms, and we report the best results obtained from different parameters.

4.1 Datasets

This section describes the datasets we use and the experimental protocols for these datasets. We evaluate our algorithm on three action image datasets, i.e., Sport Action [26], People playing musical instruments (PPMI) [27] and Still DB [28], see Fig. 2. These datasets are collected from various sources. Thus, we investigate the performance of our approach on diverse datasets with different resolutions, illumination changes, background clutter, irregular motion, etc.

The sport action dataset [26] consists of six human action classes: tennis-forehand, tennis-serve, volleyball-smash, cricket-defensive shot, cricket-bowling, and croquet-shot. The images for the first five classes are downloaded from internet and the sixth class are collected from publicly available dataset. The classes were selected so that they had significant confusion due to scene and pose. We follow the original experimental setup of the authors, i.e., dividing the 180 samples into test set (30 from each class) and 120 samples into training set (20 from each class).

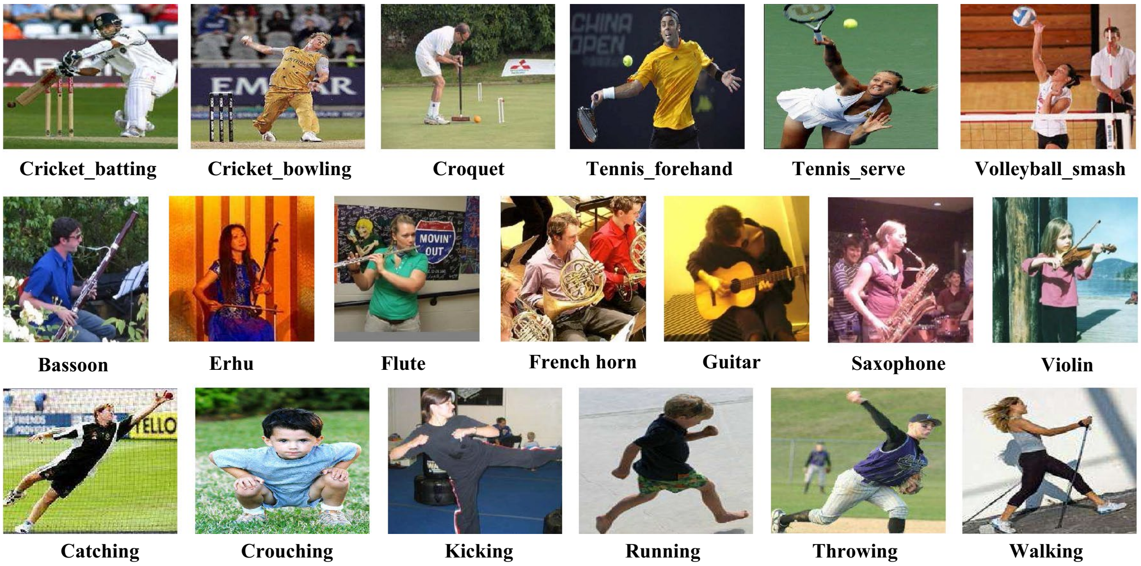


Fig. 2 Sample images from the three action recognition datasets used in our experiments. From *top* to *bottom*: sport action, PPMI and still DB

Table 1 Recognition results of different datasets

Datasets	SVR (%)	hrSTR (%)	hrTRR (%)	orSTR (%)	orTRR (%)	STuM (%)	TuRR (%)
Sport action	66.67	71.67	68.33	67.50	66.67	74.17	**77.50**
PPMI	64.57	66.71	68.29	65.43	66.29	69.14	**71.57**
Still DB	48.15	50.26	50.79	49.74	49.21	51.85	**55.56**

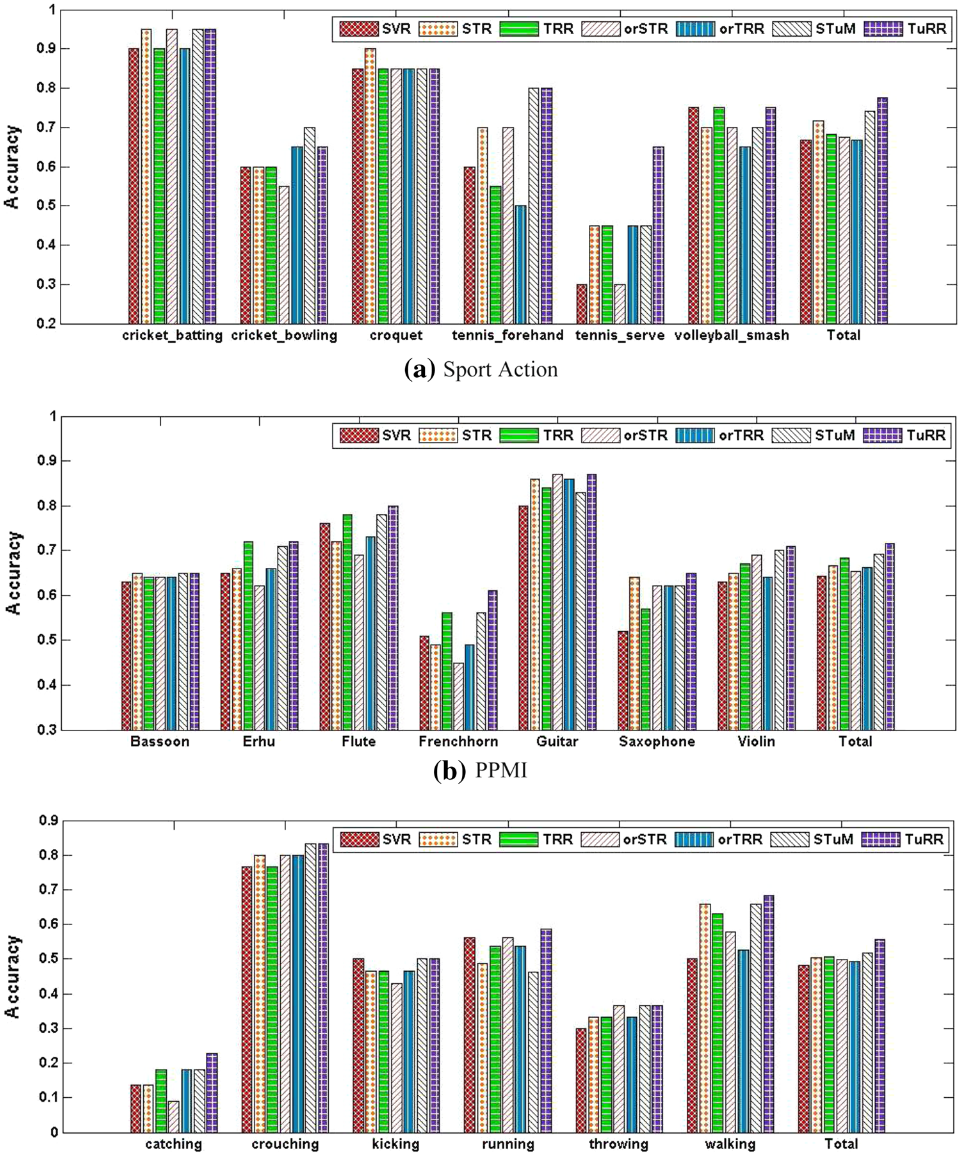
The best results are highlighted in bold

The PPMI dataset [27] contains 7 activities of humans playing instruments; playing bassoon, playing erhu, playing flute, playing French horn, playing guitar, playing saxophone, and playing violin. All the images are downloaded from internet. So images in PPMI are highly diverse and cluttered. Distinguishing PPMI images of the same instrument strongly depends on the spatial information in the images, such as the spatial relations between the object and the human. This property can be captured by HOG tensor descriptor. We follow the original setup, for each class, 100 normalized PPMI images are randomly selected for training and the remaining 100 images for testing.

The still DB [28] has been collected from various sources like Google Image Search, Flickr, BBC Motion database, etc. This dataset consists of 467 images and includes 6 different actions. These are running, walking, catching, throwing, crouching and kicking. This image collection involves a huge amount of diversity by means of viewpoints, shooting conditions, cluttered backgrounds and resolution. For Still DB dataset, we cannot obtain the split of training/test samples according to the original setup. So for every class, we randomly select 60 % images for training and 40 % images for testing. We repeat the splitting for 10 times and report the average results.

4.2 Experimental results and discussion

In Table 1, we report the comparisons of different algorithms on three datasets. From the Table 1, we observe that: (a) The tensor-based algorithms outperform the vector-based algorithm (SVR), indicating that tensor representation can better utilize the spatial structure information for action recognition, In contrast, vector-based algorithm simply concatenates the elements of HOG descriptor into a vector, and thus the spatial information of HOG descriptor may be lost. (b) STuM gains the top performance among all datasets, which indicates that tensor learning can do benefit much from the usage of Tucker tensor decomposition. (c) TuRR consistently gains the best performances among all the comparing algorithms. It indicates that tensor Tucker decomposition and tensor Ridge Regression learning both contribute to the performance. The TuRR gains around performance improvement 4.5, 3.5, and 7.2 % over these algorithms for each dataset, respectively. (d) The overall average performance is lower on the Still DB set than that on the other two datasets. The reasons are: on one hand, the number of the testing images on Still DB set is larger than that on Sport Action set. Meanwhile, the number of the training images on Still DB set is smaller than that on



(a) Sport Action

(b) PPMI

(c) Still DB

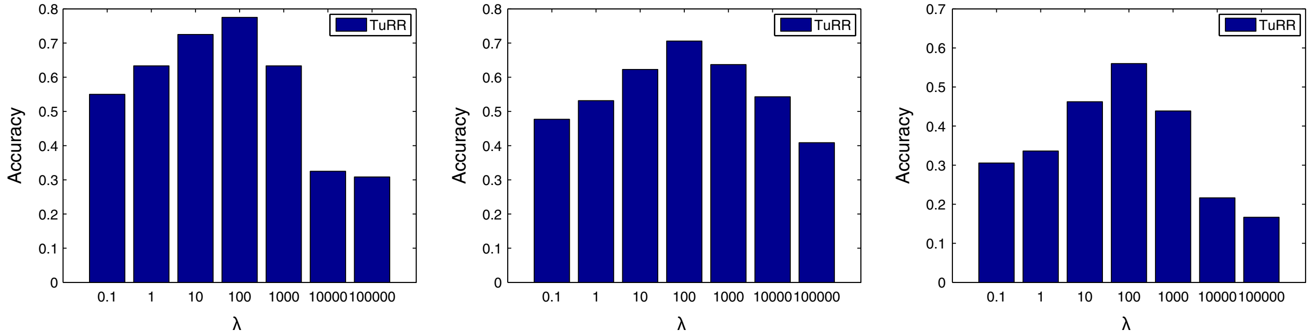
Fig. 3 Recognition results of different algorithms with our algorithms. This figure shows the accuracy corresponding to each class of the three action recognition datasets used in our experiments. ‘Total’ is the average accuracy of all classes

PPMI set. On the other hand, Still DB set is more challenging due to many similar action classes. For example, the actions of the humans “running” and “walking”, “catching” and “throwing” shown in Fig. 2 are similar in shape appearance. Therefore, it is difficult to recognize these actions in still images.

To further investigate the effectiveness of proposed TuRR, we additionally report the accuracy of each class in Fig. 3 when the HOG descriptor is represented by vector and tensor. From the Fig. 3, we observe that the performance of TuRR is more stable and it always gains good performance for different classes. It is also worth mentioning that the overall average performance is noticeably low on certain category of actions in each dataset, e.g., “tennis-serve”, “Frenchhorn” and “catching” action classes. The main reason is that these human poses are easily confused with “tennis-forehand”, “Saxophone” and “throwing”, respectively. But the proposed TuRR obtains the better result for “tennis-serve”, “Frenchhorn” and “catching” classes. This indicates that TuRR can effectively select the discriminative spatial information from tensor representation and enhance the performance of action recognition.

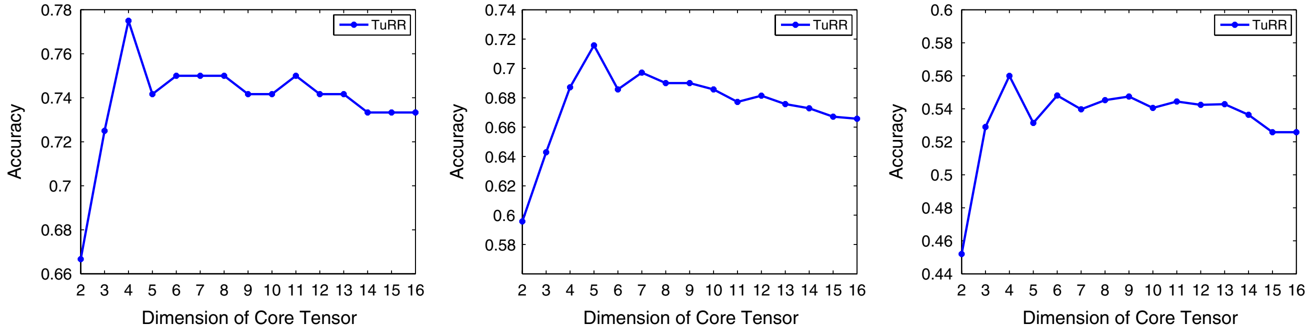
4.3 Parameter sensitivity and convergence

We tune the parameter λ of TuRR for each dataset. The parameter tuning results are shown in Fig. 4. The best performance on three datasets was obtained when $\{\lambda = 100\}$. According to [20], the dimension of core tensor is equal to or less than initial tensor. In the following, we fix the value



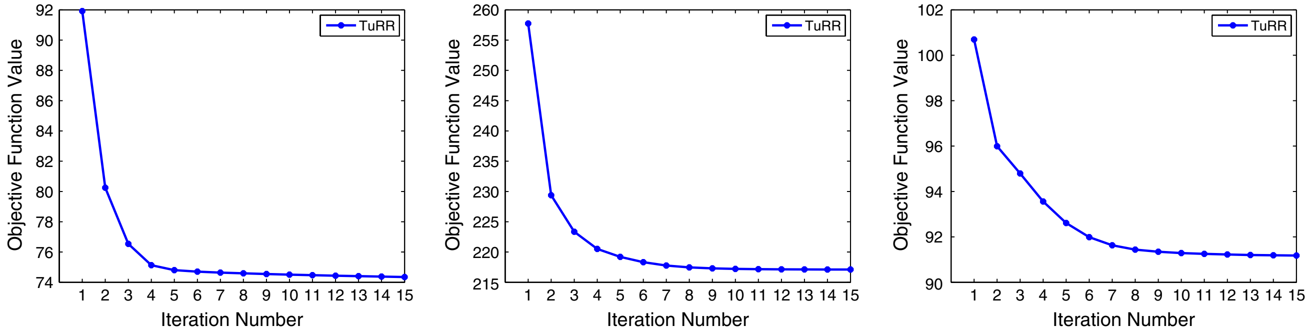
(a) Sport Action **(b)** PPMI **(c)** Still DB

Fig. 4 Parameter sensitivity



(a) Sport Action **(b)** PPMI **(c)** Still DB

Fig. 5 Dimension of core tensor R versus recognition accuracy for all the datasets



(a) Sport Action **(b)** PPMI **(c)** Still DB

Fig. 6 *Convergence curves* of the objective function value in Eq. (11) using Algorithm 1. The figure show that the objective function value monotomically decreases until converged by applying the proposed algorithm

of λ to be the values with which the best performance is obtained and tune the dimension of core tensor R from 2 to 16.

In Fig. 5, we plot the recognition accuracy against dimension of core tensor for all the datasets. In this paper, the HOG tensor descriptors are $16 \times 16 \times 16$ for action images. The dimension of initial tensor d is 16. It is clear that the performances are almost weakened when $R > 12$. The best recognition accuracy rate is achieved when $R < 6$. It indicates that the choice of a smaller dimension core tensor leads to dimension reduction tailored to the recognition problem and if the R is properly chosen, and the most significant spatial information will be retained.

Moreover, we study the convergence of the proposed TuRR in Algorithm 1. Figure 6 shows the convergence curves of our TuRR algorithm according to the objective function value in Eq. (11) on all the datasets. The figure shows that the objective function value monotonically decreases until converged by applying the proposed algorithm. It can be seen that our algorithm converges within a few iterations. For example, it takes no more than 10 iterations for Sport Action, and PPMI and no more than 15 iterations for Still DB.

5 Conclusions

We have presented a action recognition algorithm based on 3-order HOG tensor representation. With this tensor representation, the natural structure of the HOG descriptor is preserved, and spatial cues can be adopted to describe the action. The proposed algorithm is constructed by extended the vector-based Ridge Regression to the tensor representation. We propose Tucker tensor decomposition to decompose the tensor parameter during learning. This tensor decomposition can obtain more accurate decomposition result than CP decomposition. Meanwhile, we can select the discriminative spatial information from tensor representation and use it to enhance the performance of action recognition. Compared to the related methods, our proposed algorithm is proven to be more robust. Experiments have demonstrated the effectiveness of the proposed approach. In real word, massive amount of action image datum has been emerging on the web for publishing and sharing. However, most images are unlabeled or weak-labeled. To achieve an effective and efficient action recognition for real-word images, the proposed method can be extended to the semi-supervised algorithm, which leverages both labeled and unlabeled data. Moreover, in order to make full use of the spatial information, many tensor decomposition methods and high-order features have been proposed. We can use other tensor decomposition methods and high-order features to design new algorithms, which are applied to many tasks of multimedia analysis, i.e., video tracking, event analysis, age estimation and so on.

Acknowledgments This work was partly supported by the National Program on the Key Basic Research Project (under Grant 2013CB329301), NSFC (under Grant 61202166, 61472276), the Major Project of National Social Science Fund (under Grant 14ZDB153) and Doctoral Fund of Ministry of Education of China (under Grant 20120032120042).

References

1. Wang, M., Hua, X.-S.: Active learning in multimedia annotation and retrieval: a survey. ACM Trans Intell Syst Technol **2**(2), 10 (2011)
2. Wu, F., Yuan, Y., Rui, Y., Yan, S., Zhuang, Y.: Annotating web images using nova: non-convex group sparsity. In: Proceedings of the 20th ACM international conference on Multimedia, pp. 509–518. ACM, (2012)
3. Wang, M., Gao, Y., Lu, K., Rui, Y.: View-based discriminative probabilistic modeling for 3d object retrieval and recognition. IEEE Trans Image Process **22**(4), 1395–1407 (2013)
4. Wu, F., Tan, X., Yang, Y., Tao, D., Tang, S., Zhuang, Y.: Supervised nonnegative tensor factorization with maximum-margin constraint. In: 27th AAAI Conference on Artificial Intelligence, AAAI, pp. 962–968, (2013)
5. Guo, W., Kotsia, I., Patras, I.: Tensor learning for regression. IEEE Trans Image Process **21**(2), 816–827 (2012)
6. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, CVPR, vol. 1, pp. 886–893. IEEE, (2005)
7. Fischer, S., Šroubek, F., Perrinet, L., Redondo, R., Cristóbal, G.: Self-invertible 2d log-gabor wavelets. Int J Comput Vis **75**(2), 231–246 (2007)
8. Hatun, K., Duygulu, P.: Pose sentences: a new representation for action recognition using sequence of pose words. In: 19th International Conference on Pattern Recognition. ICPR, pp. 1–4. IEEE, (2008)
9. Ikizler-Cinbis, N., Gokberk Cinbis, R., Sclaroff, S.: Learning actions from the web. In: IEEE 12th International Conference on Computer Vision, pp. 995–1002. IEEE, (2009)
10. Vo, T., Tran, D., Ma, W., Nguyen, K.: Improved hog descriptors in image classification with cp decomposition. In: Neural Information Processing, pp. 384–391. Springer, (2013)
11. Shakhnarovich, G., Indyk, P., Darrell, T.: Nearest-neighbor methods in learning and vision: theory and practice. (2006)
12. Cortes, C., Vapnik, V.: Support vector machine. In: Machine learning, vol. 20, pp. 273–297. Springer, (1995)
13. Hoerl, A.E., Kennard, R.W.: Ridge regression: biased estimation for nonorthogonal problems. Technometrics **12**(1), 55–67 (1970)
14. Vasilescu M.A.O., Terzopoulos, D.: Multilinear subspace analysis of image ensembles. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. II-93. IEEE, (2003)
15. Pang, Y., Li, X., Yuan, Y.: Robust tensor analysis with l1-norm. IEEE. Trans. Circuits. Syst. Video. Technol. **20**(2):172–178 (2010)
16. Cai, D., He, X., Han, J.: Subspace learning based on tensor analysis. Department of Computer Science Technology Report No. 2572, University of Illinois at Urbana-Champaign (UIUCDCS-R-2005-2572), (2005)
17. Yan, S., Xu, D., Yang, Q., Zhang, L., Tang, X., Zhang, H.-J.: Multilinear discriminant analysis for face recognition. IEEE Trans Image Process **16**(1), 212–220 (2007)
18. Zhigang, M., Yi, Y., Feiping, N., Nicu, S.: Thinking of images as what they are: compound matrix regression for image classification. In: Proceedings of Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI), (2013)
19. Tao, D., Li, X, Hu, W., Maybank, S., Wu, X.: Supervised tensor learning. In: Fifth IEEE International Conference on Data Mining (ICDM), pp. 450–457. IEEE, (2005)
20. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM Rev **51**(3), 455–500 (2009)
21. Tichavsky, P., Phan, A.H., Koldovsky, Z.: Cramér-rao-induced bounds for candecomp/parafac tensor decomposition. IEEE Trans Signal Process **61**(8), 1986–1997 (2013)
22. Kotsia, I., Patras, I.: Support tucker machines. In: Computer Vision and Pattern Recognition (CVPR), pp. 633–640. IEEE, (2011)
23. Fung, G., Mangasarian, O.L.: Multicategory proximal support vector machine classifiers. Mach Learn **59**(1–2), 77–97 (2005)

24. Basak, D., Pal, S., Patranabis, D.C.: Support vector regression. Neural Inform Process Lett Rev **11**(10), 203–224 (2007)
25. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans Pattern Anal Mach Intell **26**(5), 530–549 (2004)
26. Gupta, A., Kembhavi, A., Davis, L.S.: Observing human-object interactions: using spatial and functional compatibility for recognition. IEEE Trans Pattern Anal Mach Intell **31**(10), 1775–1789 (2009)
27. Yao, B., Fei-Fei, L.: Grouplet: a structured image representation for recognizing human and object interactions. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9–16. IEEE, (2010)
28. Ikizler, N., Cinbis, R.G., Pehlivan, S., Duygulu, P.: Recognizing actions from still images. In: 19th International Conference on Pattern Recognition. ICPR 2008, pp. 1–4. IEEE, (2008)