TENSOR OBJECT CLASSIFICATION VIA MULTILINEAR DISCRIMINANT ANALYSIS NETWORK
Rui Zeng, Jiasong Wu, Lotfi Senhadji, Huazhong Shu

To cite this version:

<hal-01380110>

HAL Id: hal-01380110

https://hal-univ-rennes1.archives-ouvertes.fr/hal-01380110

Submitted on 24 Feb 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
TENSOR OBJECT CLASSIFICATION VIA MULTILINEAR DISCRIMINANT ANALYSIS NETWORK

Rui Zeng*, Jiasong Wu†‡*, Lotfi Senhadji†‡ and Huazhong Shu**

* LIST, Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing 210096, China  
† INSERM U 1099, 35000 Rennes, France  
‡ Laboratoire Traitement du Signal et de l’Image, Université de Rennes 1, Rennes, France  
* Centre de Recherche en Information Biomédicale Sino-française (CRIBs)

E-mail: zengrui@seu.edu.cn, jswu@seu.edu.cn, lotfi.senhadji@univ-rennes1.fr, shu.list@seu.edu.cn

ABSTRACT

This paper proposes an multilinear discriminant analysis network (MLDANet) for the recognition of multidimensional objects, known as tensor objects. The MLDANet is a variation of linear discriminant analysis network (LDANet) and principal component analysis network (PCANet), both of which are the recently proposed deep learning algorithms. The MLDANet consists of three parts: 1) The encoder learned by MLDA from tensor data. 2) Features maps obtained from decoder. 3) The use of binary hashing and histogram for feature pooling. A learning algorithm for MLDANet is described. Evaluations on UCF11 database indicate that the proposed MLDANet outperforms the PCANet, LDANet, MPCA+LDA, and MLDA in terms of classification for tensor objects.

Index Terms— Deep learning, MLDANet, PCANet, LDANet, tensor object classification

1. INTRODUCTION

One key ingredient for the success of deep learning in visual content classification is the utilization of convolution architectures [1, 2, 3], which are inspired by the structure of human visual system [4]. A convolution neural network (CNN) [2] consists of multiple trainable stages stacked on the top of each other, following a supervised classifier. Each stage of CNN is organized in two layers: convolution layer and pooling layer.

Recently, Chan et al. [5] proposed a new convolutional architecture, namely, principal component analysis network (PCANet), which uses the most basic operation (PCA) to learn the dictionary in the convolution layer and the pooling layer is composed of the simplest binary hashing and histogram. The PCANet leads to some pleasant and thought-provoking surprises: such a basic network has achieved the state-of-the-art performance in many visual content datasets. Meanwhile, Chan et al. [5] proposed linear discriminant analysis network (LDANet) as a variation of PCANet.

However, PCANet and LDANet are deteriorated when dealing with visual content, which is naturally represented as tensor objects. That is because when using PCANet or LDANet, the multidimensional patches, taken from visual content, are simply converted to vectors to learn the dictionary. It is well known that vector representation of patches breaks the natural structure and correlation in the original visual content. Moreover, it may also, suffers from the so-called curse of dimensionality [6].

Recently, there is growing interest in the tensorial extension of deep learning algorithms. Yu et al. [7] proposed deep tensor neural network, which can be seen as a tensorial extension of deep neural network (DNN), outperforms DNN in large vocabulary speech recognition. Hutchinson et al. [8] presented the tensorial extension of deep stack neural network, which has been successfully used in MNIST handwriting image recognition, phone classification, etc. However, the similar tensorial extension research has not been reported for deep learning algorithms with convolutional architecture.

In this paper, we propose a simple deep learning algorithm for tensor object classification, that is, multilinear discriminant analysis network, which is a tensorial extension of PCANet and LDANet. The simulation on UCF11 database [9] demonstrates that the MLDANet outperforms PCANet and LDANet in terms of classification accuracy for tensor objects.

2. REVIEW OF MLDA

In this section, we briefly review MLDA [10], which is a multilinear extension of LDA. The MLDA obtains discriminative
features through maximizing the Fisher’s discrimination criterion, which is described as follows.

An $N$th tensor object is denoted as $X \in \mathcal{R}^{I_1 \times I_2 \times \cdots \times I_N}$. It is addressed by $N$ indices $i_n$, $n = 1, 2, \ldots, N$, and each $i_n$ addresses the $n$-mode of $X$. The $n$-mode tensor product of $X$ by a matrix $U \in \mathcal{R}^{I_n \times I}$ is defined as:

\[
(X \times_n U)(i_1, \ldots, i_{n-1}, i_n, i_{n+1}, \ldots, i_N) = \sum_{i_n} X(i_1, \ldots, i_N) U(i_n).
\]

(1)

The projection from tensor $X \in \mathcal{R}^{I_1 \times I_2 \times \cdots \times I_N}$ to a scalar $y$ can be described as follows:

\[
y = X \times (U^{(1)})^T \times_2 (U^{(2)})^T \cdots \times_N (U^{(N)})^T
\]

where $\{u^{(n)}_n\}^N_{n=1}$ is a set of unit projection vectors. This tensor to scalar projection is called elementary multilinear projection (EMP), which consists of one projection vector in each mode. An EMP of a tensor $X \in \mathcal{R}^{I_1 \times I_2 \times I_3}$ is illustrated in Fig. 1.

The tensor-to-vector projection (TVP) from a tensor $X \in \mathcal{R}^{I_1 \times I_2 \times \cdots \times I_N}$ to a vector $Y$ can be described as follows:

\[
Y = \begin{bmatrix} X \\ \vdots \\ X \\ \vdots \\ X \\ \end{bmatrix} \times \begin{bmatrix} u^{(1)}_1 \\ \vdots \\ u^{(N)}_N \\ \end{bmatrix}^T
\]

(2)

where $\{u^{(n)}_n\}^N_{n=1}$ is a set of unit projection vectors. This tensor to scalar projection is called elementary multilinear projection (EMP), which consists of one projection vector in each mode. An EMP of a tensor $X \in \mathcal{R}^{I_1 \times I_2 \times I_3}$ is illustrated in Fig. 1.

The tensor-to-vector projection (TVP) from a tensor $X \in \mathcal{R}^{I_1 \times I_2 \times \cdots \times I_N}$ to a vector $y \in \mathcal{R}^P$ is to find a vector set $\{u^{(n)}_n\}^P_{n=1}$, which are able to do $P$ times EMP. The process can be described as follows:

\[
y = X \times (U^{(1)})^T \times_2 (U^{(2)})^T \cdots \times_N (U^{(N)})^T
\]

where $\{u^{(n)}_n\}^N_{n=1}$ is a set of unit projection vectors. This tensor to scalar projection is called elementary multilinear projection (EMP), which consists of one projection vector in each mode. An EMP of a tensor $X \in \mathcal{R}^{I_1 \times I_2 \times I_3}$ is illustrated in Fig. 1.

3. THE ARCHITECTURE OF MLDANET

Fig. 3 shows the architecture of MLDANet for third-order tensor objects classification. It contains two convolutional layers and one pooling layer. The filter bank in each convolutional layer is learned independently. We use binary hashing and histogram as pooling operation for the features extracted from the first two convolutional layers.

3.1. The first stage of MLDANet

For the given $M$ third-order tensor objects $\{X_m\}^M_{m=1} \in \mathcal{R}^{I_1 \times I_2 \times I_3}$, which contains $C$ classes, we take tensor patches around the 1-mode and 2-mode by taking all 3-mode elements of $m$th tensor object, i.e., the tensor patch size is $k_1 \times k_2 \times I_3$, we collect all (overlapping) $I_1 \times I_2$ tensor patches from $X_m$. The tensor patches have the same class with $X_m$. We put these tensor patches into a set $t_m = \{i_{m,q} \in \mathcal{R}^{k_1 \times k_2 \times I_3} \}_{1 \leq q \leq l_m}$. By repeating the above process for every tensor object, we can get all tensor patches $t = \{t_m\}^M_{m=1}$ for learning filter bank in the first stage.

Let $L_1$ be the number of filters in the first stage. We apply MLDA to $t$ to learn the $L_1$ vector sets $\{u^{(1)}_p, u^{(2)}_p, u^{(3)}_p\}^P_{p=1}$. Tensor objects are defined as follows, respectively:

\[
S_{B_p} = \sum_{c=1}^{C} N_c \langle y_{c_p} - \bar{y}_p \rangle^2, \quad S_{W_p} = \sum_{m=1}^{M} \langle y_{m_p} - \bar{y}_{m_p} \rangle^2,
\]

where $\bar{y}_p = (1/M) \sum_{m=1}^{M} y_{m_p}$, $\bar{y}_{c_p} = (1/N_c) \sum_{m \in c} y_{m_p}$. $C$ is the number of classes, $N_c$ is the number of tensor objects in class $c$, and $c_m$ is the class label for the $m$th tensor object. Thus, the Fishers Discriminant criterion for the $p$th scalar tensor objects is $F_p = S_{B_p} / S_{W_p}$. The objective of MLDA is to determine a set of $P$ EMPs $\{u^{(n)}_p\}^P_{n=1}$ satisfying the following conditions:

\[
\{u^{(n)}_p\}^P_{n=1}, \quad n = 1, \ldots, N = \arg \max F_p
\]

(5)
Each tensor patch is converted into $L_1$ scalars by using \( \{u_i^{(1)}, u_i^{(2)}, u_i^{(3)}\}_{i=1}^{L_1} \). Thus, the \( l \)th feature map of tensor object \( X_m \) in the first stage is defined as:

\[
F^{ml} = \text{mat}(t_{m,q} X_{m,l}^{(n)} R_{ml}^{(n)}) \in \mathcal{R}_{I_1 \times I_2},
\]

where \( \text{mat}(v) \) is a function that maps \( v \in \mathcal{R}_{I_1 \times I_2} \) on a matrix \( F \in \mathcal{R}_{I_1 \times I_2} \).

For each tensor object, we can obtain \( L_1 \) feature maps of size \( I_1 \times I_2 \). We denote these feature maps of \( m \)th tensor object in the first stage with \( \{F^{ml} \in \mathcal{R}_{I_1 \times I_2}, l = 1, \ldots, L_1\} \). The feature maps of each tensor object capture the main variation of original data.

### 3.2. The second stage of MLDANet

Through the first stage, the tensor object is already mapped into low-dimensional tensor feature, the dimension of the 3-mode is much lower than that of the 1-mode and 2-mode. That is to say, the redundancy of 3-mode has been greatly reduced. Therefore, for the simplicity of computation and the convenience of building network, we use the conventional LDA in the second stage to learn the filter bank. The number of filters in the second stage is \( L_2 \).

Around each pixel, we take a \( k_1 \times k_2 \) patch, and collect all (overlapping) patches of all the feature maps \( F^{ml} \), i.e., \( \{r_{ml,q}^{(h)}\}_{q=1}^{L_1 L_2} \in \mathcal{R}_{k_1 \times k_2} \) where each \( r_{ml,q}^{(h)} \) denotes the \( q \)th vectorized patch in the \( h \)th feature map of \( m \)th tensor object. We then subtract the patch mean from each patch, and construct the matrix \( R_{ml} = [r_{ml,1}, r_{ml,2}, \ldots, r_{ml,L_1 L_2}] \) for them, where \( R_{ml} \) belongs to the same class with the \( m \)th tensor object. Let \( S_c \) be the set of matrix \( R_{ml} \) in class \( c \). We then compute the class mean \( \Gamma_c \) and the mean of class \( \Gamma \) as follows:

\[
\Gamma_c = \sum_{m \in S_c} R_{ml} / |S_c|, \quad \Gamma = \sum_{m \neq m_l} R_{ml} / |ML_1|.
\]

So, the within-class scatter matrix and the between-class scatter matrix are defined respectively as follows:

\[
S_W = \sum_{c=1}^{C} (\sum_{m \in S_c} (R_{ml} - \Gamma_c)(R_{ml} - \Gamma_c)^T) / |S_c|,
\]

\[
S_B = \sum_{c=1}^{C} (\Gamma_c - \Gamma)(\Gamma_c - \Gamma)^T / C.
\]

We then get \( w^* \in \mathcal{R}_{k_1 k_2 L_2} \) by maximizing the Fisher’s discriminant criterion as follows:

\[
w^* = \arg \max_w \frac{\text{Tr}(w^T S_B w)}{\text{Tr}(w^T S_W w)}, \quad \text{s.t. } w^T w = I_{L_2}.
\]

By using mat function, each column of \( w^* \) is converted into matrix \( \{v_h \in \mathcal{R}_{k_1 k_2 L_2}, h = 1, \ldots, L_2\} \). These matrices are treated as the filter bank in the second stage. Let the \( h \)th output of the \( l \)th feature map for the \( m \)th tensor object in the second stage be

\[
G^{mlh} = R_{ml} \ast v_h, h = 1, \ldots, L_2, l = 1, \ldots, L_1.
\]

where \( \ast \) denotes 2D convolution [2], and the boundary of \( R_{ml} \) is zero-padded before convolving with \( v_h \) so as to make \( G^{mlh} \) have the same size as \( R_{ml} \). The number of output feature maps of the second stage is \( L_1 L_2 \). One or more additional stages can be built if a deep architecture is found to be beneficial.

### 3.3. The second stage of MLDANet

First, we binarize each feature map by using Heaviside step function \( H(\cdot) \), whose value is one for positive entries and zero otherwise. The binarized feature maps are denoted by \( G^{mlh} \in \{0, 1\}_{L_1 L_2} \). Owing to every feature map capture different variations by \( v_h \), \( G^{mlh} \) should be weighted to convert into a single integer-valued feature map:

\[
W^{ml} = \sum_{h=1}^{L_2} 2^{h-1} G^{mlh}.
\]

Note that each entry of feature map \( W^{ml} \) are integers in the range \([0, 2^{L_2-1}]\).

Next, we partition \( W^{ml} \) into \( B \) blocks, and then compute the histogram (with \( 2^{L_2} \) bins) of the decimal values in each block. All the \( B \) histograms are concatenated into one vector as the \( l \)th feature vector \( Bhist(W^{ml}) \) of tensor object \( X_m \). The final feature of input tensor object \( X_m \) is then defined as the set of feature vector, i.e.,

\[
f_m = [Bhist(W^{m1}), Bhist(W^{m2}), \ldots, Bhist(W^{mL_1})].
\]

Note that the local block can be either overlapping or non-overlapping depending on applications [5].

### 4. EXPERIMENTAL RESULTS

We evaluate the performance of MLDANet on UCF11 dataset for tensor object classification. UCF11 is a sport action video dataset which contains 11 action categories: basketball shooting, biking, diving, golf swinging, horseback riding, soccer juggling, swinging, tennis swing, trampoline jumping, volleyball spiking, and walking with a dog. All videos in UCF11 are manually collected from YouTube and their sizes are all \( 240 \times 320 \) pixels. For each category, the videos are grouped into 25 groups with more than 4 action clips in it. The video clips in the same group have common scenario. This video dataset is very challenging in classification due to large variations in camera motion, object appearance and pose, object scale, viewpoint, cluttered background, illumination, etc.

In this experiment, we only choose the first ten groups in each category. The total number of experimental videos is

1973
In this paper, we have proposed and implemented a novel deep learning architecture, that is, MLDANet, which takes full advantage of the structure information in tensor objects by convolutional architecture. MLDANet is composed of two convolutional layers, which use MLDA and LDA to learn filter banks respectively and one pooling layer. We have evaluated the performance of the MLDANet on UCF11 and show that our model performs well in tensor object classification. This work provides the inspiration for other convolutional deep architectures in tensor object classification. As future works, we will focus on the tensoral extension of CNN.
6. REFERENCES


