The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Hierarchical Nonlinear Orthogonal Adaptive-Subspace Self-Organizing Map Based Feature Extraction for Human Action Recognition

Yang Du,[1,2,3] **Chunfeng Yuan,**[1*] **Weiming Hu,**[1] **Hao Yang**[1,2,3]
[1]CAS Center for Excellence in Brain Science and Intelligence Technology, National Laboratory of Pattern Recognition,
Institute of Automation, Chinese Academy of Sciences;
[2]University of Chinese Academy of Sciences
[3]MTdata, Meitu
duyang2014@ia.ac.cn, {cfyuan, wmhu, hao.yang}@nlpr.ia.ac.cn

Abstract

Feature extraction is a critical step in the task of action recognition. Hand-crafted features are often restricted because of their fixed forms and deep learning features are more effective but need large-scale labeled data for training. In this paper, we propose a new hierarchical *Nonlinear Orthogonal Adaptive-Subspace Self-Organizing Map* (NOASSOM) to adaptively and learn effective features from data without supervision. NOASSOM is extended from *Adaptive-Subspace Self-Organizing Map* (ASSOM) which only deals with linear data and is trained with supervision by the labeled data. Firstly, by adding a nonlinear orthogonal map layer, NOASSOM is able to handle the nonlinear input data and it avoids defining the specific form of the nonlinear orthogonal map by a kernel trick. Secondly, we modify loss function of ASSOM such that every input sample is used to train model individually. In this way, NOASSOM effectively learns the statistic patterns from data without supervision. Thirdly, we propose a hierarchical NOASSOM to extract more representative features. Finally, we apply the proposed hierarchical NOASSOM to efficiently describe the appearance and motion information around trajectories for action recognition. Experimental results on widely used datasets show that our method has superior performance than many state-of-the-art hand-crafted features and deep learning features based methods.

Introduction

The effective extraction of features from large-scale is a challenging and highly-focused problem in many data analysis areas, such as text, speech, image and video. Usually, there exist several invariant patterns in most data, and each invariant pattern can be represented by a feature filter. The input data are filtered by a series of different filters to achieve the effective feature. In videos, there are mainly two categories of features for human action recognition:

- Many hand-crafted features are proposed for images and videos, such as SIFT (Lowe 1999), HoF (Laptev et al. 2008), HoG (Dalal and Triggs 2005) and MBH (Dalal, Triggs, and Schmid 2006). For HoG and HoF, their focused patterns are respectively Histogram of oriented Gradient and Histogram of Flow. Namely, these two features use two different histogram filters to represent patterns in the data. Besides, Gabor filter is widely used for feature extraction when the targets are moving or their illumination is varying. These hand-crafted features are limited in that their mathematical forms are fixed a priori. They focus on some specific patterns and are not applicable to various data.
- Deep learning features, like those extracted by convolutional filters, are often learned using supervised methods (BP Algorithm) on large-scale labeled data. For example, the TDD (Wang, Qiao, and Tang 2015) and Two-stream (Simonyan and Zisserman 2014) methods use the labeled frames to train their deep models, and obtain the effective CNN features by convolutional filters. If only unlabeled data are available or there is not enough training data, it is not possible to learn accurate filters.

To overcome the limitations of the above features, we propose a hierarchal *Nonlinear Orthogonal Adaptive-Subspace Self-Organizing Map* (NOASSOM) to learn an effective and adaptive feature from data without supervision. *Adaptive-Subspace Self-Organizing Map* (ASSOM) (Kohonen, Kaski, and Lappalainen 1997) is a neural network that learns feature filters from the labeled data. The data are classified using subspaces. Each subspace corresponds to a pattern found by ASSOM. The NOASSOM is extended from the ASSOM. Compared with ASSOM, the proposed NOASSOM makes changes on mainly three aspects. Firstly, we use the modified polynomial kernel to enhance the representative ability of the ASSOM from linear subspace to nonlinear orthogonal subspace. Secondly, we propose a new loss function and learning method to learn the filters from the unlabeled data. Thirdly, we extend the NOASSOM to a hierarchical architecture to extract the high-level features.

We apply the proposed hierarchical NOASSOM to extract local features of both motion and appearance information for action recognition in videos. Specifically, we use the improved dense trajectories (Wang and Schmid 2013) to extract key regions in videos. Based on the hierarchical NOASSOM, we design a spatial channel to extract the appearance information along these trajectories, and meanwhile a temporal channel to extract the motion information. Then, we use the last layer of the hierarchal NOASSOM to fuse and

*Corresponding Author

describe the spatio-temporal information of two channels. Finally, Fisher Vector (Wang and Schmid 2013) is used to encode local features described by the hierarchal NOASSOM in a video, and is classified by SVM. Experiments demonstrate that our method obtains the-state-of-the-art performance on both large-scale and small-scale datasets.

Related Works

Action recognition occupies an important position in computer vision and feature extraction is a critical step for action recognition. We review the following three kinds of features and their frameworks for action recognition in brief.

STIP/Dense volume/Trajectories+local descriptors: In the early stages, the actions are relatively simple and regular in datasets, such as KTH (Schuldt, Laptev, and Caputo 2004), Weizmann (Blank et al. 2005) and UCF Sports (Rodriguez, Ahmed, and Shah 2008). Relatively less information needs to be extracted. Hand-crafted feature can handle these situations. Space-Time Interest Points (STIP) (Laptev 2005), dense volumes (Le et al. 2011b) and Trajectories (Matikaninen, Hebert, and Sukthankar 2009)(Wang et al. 2013)(Wang and Schmid 2013) are often used to extract key regions in videos. Some popular local features are proposed to represent the 3D volumes extracted around these key regions, such as HOF, HOG, 3D Histogram of Gradient (HOG3D) (Klaser, Marszalek, and Schmid 2008), ISA (Le et al. 2011b), Motion Boundary Histograms (MBH) (Dalal, Triggs, and Schmid 2006) and Extended SURF (ESURF) (Willems, Tuytelaars, and Gool 2008). When larger and challenging datasets emerge, the hand-crafted features are too weak to produce good performance. The trajectory-Pooled Deep-Convolutional Descriptors (TDD) method (Matikaninen, Hebert, and Sukthankar 2009) uses the pre-trained Two-Stream ConvNets to extract the deep learning features along the trajectories in videos and achieves good results on datasets.

End-to-end network: In this framework, the video is directly used or is decomposed into frames and optical flows as input to the end-to-end networks. Deep learning features are implicitly extracted in networks. The 3D ConvNet (Ji et al. 2013) extends the 2D ConvNet to directly train using videos, but it needs abundant computations and has not achieved outstanding performance. Deep ConvNets (Karpathy et al. 2014) use the fusion of different layers and are trained on large scale dataset such as Sports1-M. Two-Stream ConvNet (Simonyan and Zisserman 2014) using two channels of RGB frames and optical flow is successfully applied to action recognition. Temporal Segment Network (Wang et al. 2016) divides the video into multiple parts, on which Two-Stream ConvNets are used separately. It achieves the better performance than Two-Stream. Another end-to-end network, ST-ResNet (Feichtenhofer, Pinz, and Zisserman 2016) obtains the higher performance than previous methods.

Temporal modeling framework: Many methods model temporal structure (Zhu et al. 2016)(Song et al. 2017) of the low-level visual features which are obtained by deep learning or hand-craft. The temporal modeling methods include recurrent neural network (RNN) (Elman 1990), or its variants such as long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997). For example, a stack of LSTMs upon CNN features is used to learn the high-level temporal structure (Donahue et al. 2015) or to model the dynamics of CNN features (Ng et al. 2015). There are also methods based on hidden Markov model (HMM) (Sun and Nevatia 2013) and linear dynamic systems (LDS) (Bhattacharya et al. 2014). These methods decompose actions into multiple states corresponding to shots of sub-actions. VLAD^3 (Li, Li, and Vasconcelos 2016) represents a video by encompassing long-range level of the hierarchy in videos.

Hand-crafted features are often more effective in small-scale datasets than deep learning features, but they are restricted in more complicated applications because of their fixed forms. Deep learning features are more effective for the large-scale datasets but they need a large amount of data for training. In our paper, the proposed hierarchical NOASSOM can adaptively learn feature filters from unlabeled videos. Subsequently, we use the hierarchical NOASSOM to sufficiently extract the local features in videos.

Nonlinear Orthogonal Adaptive-Subspace Self-Organizing Map

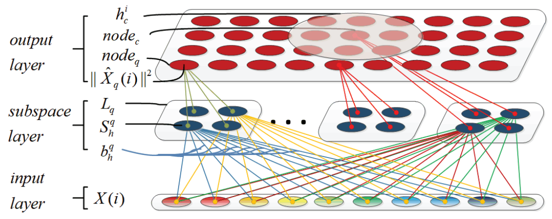
In this section, we briefly introduce the general ASSOM, and then propose our NOASSOM. Next, we present the unsupervised training method of NOASSOM.

The general ASSOM

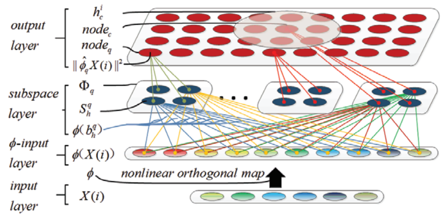
Adaptive-Subspace Self-Organizing Map (ASSOM) is a modular neural network, the modules of which adaptively learn to identify input patterns subject to different transformations. Specifically, ASSOM includes three layers as shown in Figure 1(a). The $subspace$ layer has many subspaces L_q , which learn the different patterns from the $input$ layer $X(i)$. The $output$ layer $||\hat{X}_q(i)||^2$ is composed of the linear orthogonal projections of the $input$ layer on all subspaces of the $subspace$ layer, and it can be used as the feature representation of the $input$.

On one hand, the input data are linearly projected to the subspaces of ASSOM. However, the data are often nonlinear in reality. If the input data are highly nonlinear, the subspaces of ASSOM cannot effectively learn the patterns from data. It is theoretically demonstrated that nonlinear data can be linearly separable if the data are mapped to a space with a high enough dimension. An effective method is to first map the original data to a high dimension to make data linear, and then the data after mapped can be linearly projected to the subspaces in the high dimensional space.

On the other hand, ASSOM needs the labeled data for supervised learning and cannot work on unsupervised situations. It is usually used for the classification of the supervised data. For example, (Peng et al. 1999) uses 10 different classes of digits $0 \sim 9$ to train 10 different ASSOMs with supervision for handwritten digit recognition. In addition, (Liu 2002) uses the features of the same person to train the corresponding ASSOM for face recognition.



(a) The architecture of ASSOM



(b) The architecture of the proposed NOASSOM.

Figure 1: The illustrations of ASSOM and NOASSOM. ASSOM consists of three layers, which are *input*, *subspace* and *output* layer. NOASSOM consists of four layers, which are *input*, ϕ-*input*, *subspace* and *output* layer.

The model of NOASSOM

We propose a NOASSOM to effectively learn the patterns from the nonlinear data by adding a nonlinear orthogonal map layer. Simultaneously, we modify the loss function of ASSOM to learn patterns without supervision.

The architecture of NOASSOM is illustrated in Figure 1(b). It consists of four layers, *input*, ϕ-*input*, *subspace* and *output* layer. Compared with ASSOM, NOASSOM adds a nonlinear orthogonal map layer, denoted as ϕ-*input*, which is mapped from the input layer by ϕ. Assume that the input stimulus is denoted as $\{X(i)\}_{i=1}^{N}$, where the P-dimensional vector $X(i)$ is the i-th input sample and N is the number of the samples. There are Q nodes which are denoted as $\{node_q\}_{q=1}^{Q}$ in the output layer. Each output $node_q$ in the output layer corresponds to a mapped subspace Φ_q in the subspace layer. Φ_q consists of H basis vectors $\{\phi(b_h^q)\}_{h=1}^{H}$, which are mapped from $\{b_h^q\}_{h=1}^{H}$ by ϕ. Specifically, $\{b_h^q\}_{h=1}^{H}$ are the orthonormal basis vectors in the original subspace L_q, and have the same dimension with $X(i)$ satisfying that,

$$< b_{h_1}^q, b_{h_2}^q >= \begin{cases} 0, & h_1 \neq h_2, \\ 1, & otherwise. \end{cases} \tag{1}$$

$\{\phi(b_h^q)\}_{h=1}^{H}$ respectively correspond to the subspace nodes $\{S_h^q\}_{h=1}^{H}$ in the mapped subspace Φ_q and are represented as the connections between the ϕ-*input* layer and the subspace nodes $\{S_h^q\}_{h=1}^{H}$ in the subspace layer.

In order to preserve the orthogonality after the basis vectors $\{b_h^q\}_{h=1}^{H}$ are mapped, we set $\phi(b_h^q)$ to satisfy that,

$$< \phi(b_{h_1}^q), \phi(b_{h_2}^q) >= 0, h_1 \neq h_2. \tag{2}$$

In this way, we achieve the nonlinear orthogonal map of $X(i)$ and b_h^q by ϕ. Then, we call the mapped subspace $\Phi_q = \{\phi(b_h^q)\}_{h=1}^{H}$ the nonlinear orthogonal subspace.

Subsequently, we derive the loss function E_ϕ of NOASSOM by computing the nonlinear orthogonal projection in the output layer for each input sample based on the assumption in Equation 2. The specific implement of ϕ is detailed in the next section. Let $O_i(S_h^q)$ denote the nonlinear orthogonal projection of the input $X(i)$ on the basis vector b_h^q, and it is achieved by the orthogonal projection of $\phi(X(i))$ on $\phi(b_h^q)$,

$$O_i(S_h^q) = \frac{\phi^T(b_h^q)}{||\phi^T(b_h^q)||}\phi(X(i))\frac{\phi(b_h^q)}{||\phi(b_h^q)||}. \tag{3}$$

Let $\hat{\phi}_q(X(i))$ denote the nonlinear orthogonal projection of $X(i)$ on the mapped subspace Φ_q. Due to the orthogonality of $\{\phi(b_h^q)\}_{h=1}^{H}$ by Equation 2, $\hat{\phi}_q(X(i))$ can be calculated by the easy sum of $O_i(S_h^q)$ with respect to all the mapped basis vectors $\{\phi(b_h^q)\}_{h=1}^{H}$ of the subspace Φ_q, as follows,

$$\hat{\phi}_q(X(i)) = \sum_{h=1}^{H} O_i(S_h^q). \tag{4}$$

$||\hat{\phi}_q(X(i))||^2$ is used as the output of $X(i)$ on $node_q$ and it represents the response value of $X(i)$ on the mapped subspace Φ_q. Simultaneously, the orthogonal projection error vector of $\phi(X(i))$ on subspace Φ_q is denoted as follows,

$$\tilde{\phi}_q(X(i)) = \phi(X(i)) - \hat{\phi}_q(X(i)). \tag{5}$$

Among all the nodes in the output layer, we define a winner node as the one with the minimal projection error $||\tilde{\phi}_q(X(i))||^2$. Mathematically, for each input $X(i)$, the index $c(i)$ of its winner node $node_{c(i)}$ is formulated as,

$$c(i) = \underset{q}{argmin}\{||\tilde{\phi}_q(X(i))||^2\}. \tag{6}$$

After obtained the projection errors of all samples on all output nodes, the loss function is defined as minimizing the weighted sum of these projection errors, as follows,

$$E_\phi = \sum_{i=1}^{N}\sum_{q=1}^{Q} h_{c(i)}^q ||\tilde{\phi}_q(X(i))||^2, \tag{7}$$

where $h_{c(i)}^q$ is the weight and it is a decreasing function of the distance between the winner $node_{c(i)}$ and $node_q$ in the NOASSOM array. We often choose

$$h_{c(i)}^q = \exp(-\frac{||c(i) - q||^2}{2\sigma^2}). \tag{8}$$

Discussion: The loss function of ASSOM has the following form (Kohonen, Kaski, and Lappalainen 1997),

$$E = \sum_{q=1}^{Q} h_c^q \sum_{i=1}^{N} ||\tilde{X}(i)||^2, \tag{9}$$

where $||\tilde{X}(i)||^2$ is the orthogonal projection error of $X(i)$ on the subspace L_q in ASSOM. Each output $node_q$ corresponds to a pattern, and each input $X(i)$ is assigned to a winner $node_c$. The $node_c$ acts as a label for a pattern class. In Equation 9, the loss of ASSOM is calculated by $\sum_{i=1}^{N} ||\tilde{X}(i)||^2$. The samples $\{X(i)\}_{i=1}^{N}$ must be of the same label to train the class-specific ASSOM with supervision, because they share one winner $node_c$ and h_c^q in Equation 9. If $\{X(i)\}_{i=1}^{N}$ are with different labels, they lie in different patterns and have different winner $node_c$. So we must use clusters of samples with the same label $node_c$ for supervised training of ASSOM. By experiments, a group of the unlabeled data can not converge this ASSOM.

For NOASSOM, $\{X(i)\}_{i=1}^{N}$ individually contributes to loss in Equation 7. We don't need to know each pattern of $X(i)$, which has its own winner $node_{c(i)}$ and weight $h_{c(i)}^q$. Then $h_{c(i)}^q$ is used to obtain the weighted sum of these errors.

The unsupervised training of NOASSOM

The parameters needed to learn in NOASSOM include the orthonormal basis vectors b_h^q and the nonlinear orthogonal map ϕ. In order to avoid directly computing ϕ, we employ a nonlinear kernel $\kappa(u, v) =< \phi(u), \phi(v) >$ to replace ϕ in the final loss function E_ϕ in Equation 7.

Specifically, $\hat{\phi}_q(X(i))$ in Equation 4 is reformulated in the term of κ, as follows,

$$\hat{\phi}_q(X(i)) = \sum_{h=1}^{H} \frac{\kappa(b_h^q, X(i))}{\kappa(b_h^q, b_h^q)} \phi(b_h^q). \tag{10}$$

Then, the output $||\hat{\phi}_q(X(i))||^2$ and the projection error $||\tilde{\phi}_q(X(i))||^2$ are reformulated in the term of κ, as follows,

$$\begin{aligned} ||\hat{\phi}_q(X(i))||^2 &=< \hat{\phi}_q(X(i)), \hat{\phi}_q(X(i)) > \\ &= \sum_{h=1}^{H} \frac{\kappa^2(b_h^q, X(i))}{\kappa(b_h^q, b_h^q)} \\ ||\tilde{\phi}_q(X(i))||^2 &=< \tilde{\phi}_q(X(i)), \tilde{\phi}_q(X(i)) > \\ &= \kappa(X(i), X(i)) - \sum_{h=1}^{H} \frac{\kappa^2(b_h^q, X(i))}{\kappa(b_h^q, b_h^q)}. \end{aligned} \tag{11}$$

The normalization can be then incorporated into the loss function Equation 7 by using relative projection errors $||\tilde{\phi}_q(X(i))||^2/||\phi(X(i))||^2$. By Equation 11, the final loss function E_ϕ is recalculated as follows,

$$\begin{aligned} E_\phi &= \sum_{i=1}^{N} \sum_{q=1}^{Q} h_{c(i)}^q \frac{||\tilde{\phi}_q(X(i))||^2}{||\phi(X(i))||^2} \\ &= \sum_{i=1}^{N} \sum_{q=1}^{Q} h_{c(i)}^q [1 - \sum_{h=1}^{H} \frac{\kappa^2(b_h^q, X(i))}{\kappa(X(i), X(i))\kappa(b_h^q, b_h^q)}] \end{aligned} \tag{12}$$

The kernel needs to be artificially selected and the modified polynomial kernel is the most effective kernel to satisfy the orthogonality assumption in Equation 2, we adopt a modified polynomial kernel function,

$$\kappa(u, v) =< \phi(u), \phi(v) >= \sum_{j=1}^{L} w_j (u^T v)^j \tag{13}$$

L is the highest order of the polynomial, and w_j is the weight of the product term in kernel. We use the gradient descent method to automatically learn the w_j and b_h^q, as follows,

$$\begin{aligned} b_h^q(t+1) &= b_h^q(t) - 0.5\lambda(t)\frac{\partial E_\phi}{\partial b_h^q} \\ w_j(t+1) &= w_j(t) - 0.5\lambda(t)\frac{\partial E_\phi}{\partial w_j} \end{aligned} \tag{14}$$

where $\lambda(t)$ is the learning rate, which is a decreasing function over the time. The specific form will be detailed in the experiment section. To accelerate the convergency, the learning rate is set a monotonically increasing function of $||\tilde{\phi}_q(X(i))||^2$ or decreasing function of $||\hat{\phi}_q(X(i))||^2$. So we divide $\lambda(t)$ by $||\hat{\phi}_q(X(i))||/||\phi(X(i))||$. The final learning rules of E_ϕ with respect to the basis vector b_h^q and w_j are calculated as follows,

$$\begin{gathered} b_h^q(t+1) = b_h^q(t) + \\ \sum_{i=1}^{N} \sum_{j=1}^{L} \frac{\lambda(t) h_{c(i)}^q j w_j [{b_h^q}^T X(i)]^{j-1} \kappa(b_h^q, X(i)) X(i)}{||\hat{\phi}_q(X(i))|| ||\phi(X(i))|| \kappa(b_h^q, b_h^q)}, \\ w_j(t+1) = w_j(t) + \\ \sum_{i=1}^{N} \sum_{q=1}^{Q} \sum_{h=1}^{H} \frac{\lambda(t) h_{c(i)}^q ||\phi(X(i))||}{2[\kappa(X(i), X(i))\kappa(b_h^q, b_h^q)]^2 ||\hat{\phi}_q(X(i))||} \\ [2\kappa(b_h^q, X(i))[{b_h^q}^T X(i)]^j \kappa(X(i), X(i))\kappa(b_h^q, b_h^q) - \\ \kappa^2(b_h^q, X(i))[[X(i)^T X(i)]^j \kappa(b_h^q, b_h^q) + \kappa(X(i), X(i))] \end{gathered} \tag{15}$$

After $b_h^q(t+1)$ is updated in each iteration t, we orthogonalize the basis vectors $\{b_h^q\}_{h=1}^{H}$ again by Gram-Schmidt Orthogonalization as in ASSOM. This can make $\{\Phi(b_h^q)\}_{h=1}^{H}$ orthogonal because of our kernel in Equation 13.

Activation function for feature extraction: when the learning process of NOASSOM is finished, it can be used for feature extraction. We use the activation of the output layer of NOASSOM as the final feature of the input layer. Given any input X, its activated output with respect to the output $node_q$ is defined as follows,

$$\mathcal{O}_q = \mathcal{F}(||\hat{\phi}_q(X)||^2) \tag{16}$$

where we employ the bipolar sigmoid function as the activation function, $\mathcal{F} = \frac{1-e^{-\alpha x}}{1+e^{-\alpha x}}$. Finally, $[\mathcal{O}_1, \mathcal{O}_2, ..., \mathcal{O}_Q]$ is used as the feature of the input X by our NOASSOM.

Hierarchical NOASSOM Descriptor for Action Recognition

Inspired by the fact that the proposed NOASSOM can be used to adaptively extract feature from data without supervision, we use it to construct a hierarchical NOASSOM to describe the local regions for action recognition.

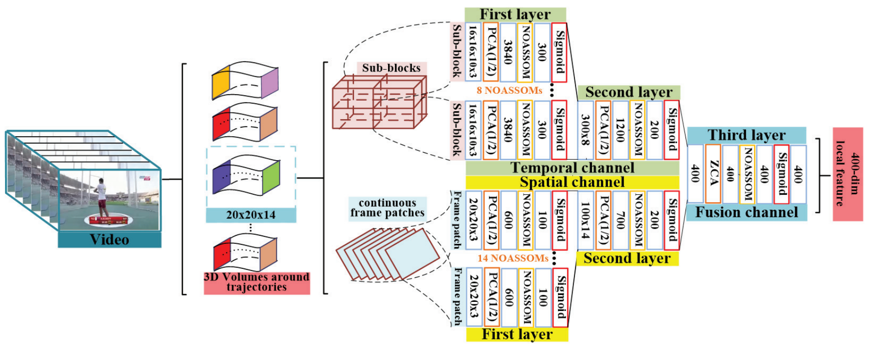


Figure 2: The hierarchical NOASSOM descriptor for the 3D volume around the improved dense trajectoy. The size of the 3D Volume is 20×20(spatial)×14(temporal). The size of the sub-block is 16×16 (spatial)×10 (temporal)×3 (RGB). The size of the frame patch is 20×20 (spatial)×3 (RGB). The dimensions of the output of each NOASSOM in the first layer on the spatial and temporal channels are respectively 100 and 300. The numbers of NOASSOMs in the first layer on the spatial and temporal channels are respectively 14 and 8. The dimensions of the output of the NOASSOM in the second layer on two channels are both 200. The dimension of the output of the NOASSOM in the third layer is 400.

At first, we extract improved dense trajectories from all videos by using the method (Wang and Schmid 2013) on its original spatial scale, as done in (Wang, Qiao, and Tang 2015). Improved dense trajectories are widely used to extract key regions in the video and boost the recognition performance of dense trajectories (Wang et al. 2013) by taking camera motion into account. Around each trajectory, a 3D volume is extracted. Subsequently, we present a hierarchical NOASSOM model to produce a 400-dim local feature for the 3D volume by the fusion of the spatial and temporal channels. The proposed hierarchical NOASSOM model is composed of three layers. The former two layers include PCA, NOASSOM and activation operation and the last layer replaces the PCA with ZCA, as shown in Figure 2. We use PCA to halve the input dimension and use ZCA to whiten the input data for better learning NOASSOM.

Specifically, in the first layer, we respectively extract the low-level features from the spatial and temporal channels. In the temporal channel, we divide the 3D volume into sub-blocks as input of the NOASSOM. In the spatial channel, we use the continuous frame patches along the trajectory as input of the NOASSOM. In the second layer, for each of the two channels we use a NOASSOM to fuse the extracted low-level features from all NOASSOMs of this channel in the first layer. In each channel, the concatenation of all NOASSOM outputs in the first layer is used as input of NOASSOM in the second layer. In the last layer, we use a NOASSOM in the fusion channel to fuse two channels. Namely, the concatenation of the activated outputs of two channels in the second layer is used as input of NOASSOM in the third layer. The input is whitened by ZCA at first. The activated output of the NOASSOM is a 400-dim vector used to describe the 3D volumes around these trajectories.

The training process of the proposed hierarchical NOASSOM is executed layer by layer. Firstly, we randomly sample the 3D volumes around the improved trajectories from the train datasets. Then, we randomly sample sub-blocks and frame patches from the 3D volumes to respectively train the NOASSOM of the temporal and spatial channels in the first layer. Subsequently, we use the activated outputs of the 3D volumes on the two channels in the first layer to respectively train the NOASSOM of two channels in the second layer. Finally, we use the activated output of the 3D volumes through the spatial and temporal channels to train NOASSOM of the third layer on the fusion channel.

Finally, several 400-dim local features, extracted by the hierarchical NOASSOM from videos, are used to train GMMs. Based on GMMs, local features in a video are encoded as a Fisher Vector which has been verified very effective for action recognition in previous works (Wang and Schmid 2013)(Wang, Qiao, and Tang 2015). We use the linear SVM as the classifier to classify the videos for action recognition.

Experiments

We evaluated our method on the KTH (Schuldt, Laptev, and Caputo 2004), UCF-101 (Soomro, Zamir, and Shah 2012) and HMDB-51 (Kuehne et al. 2011) action recognition benchmarks. The first one is a public traditional dataset and the other two are public large and challenging datasets. We followed the original evaluation scheme by using three different training and testing splits on UCF-101 and HMDB-51. The mean classification accuracy over these three splits is reported for our method. When tested on HMDB-51, hierarchal NOASSOM is pre-trained without supervision on UCF-101 to better initialize the network parameters and then fine-tuned on HMDB-51 train splits. Without pre-training on UCF-101, the result is less accurate, and a higher performance is obtained if the model is trained on large datasets without supervision. When evaluated on UCF-101, hierarchal NOASSOM is pre-trained without supervision on train splits of UCF-101 and HMDB-51.

For each NOASSOM, we set the basis vector number H of each subspace to 2 considering the training time and performance. We set the learning rate $\lambda(t) = \frac{0.1T}{T+99t}$ with $N = 200$, $T = 10000$ (T is the max iteration) and $L = 5$.

Table 1: Evaluations of hierarchal NOASSOM on the HMDB51 dataset and UCF-101 dataset.

Training setting	HMDB-51	UCF-101
Spatio-temporal & ASSOM	62.9%	87.4%
Spatial channel & NOASSOM	58.5%	83.6%
Temporal channel & NOASSOM	63.4%	88.2%
Spatio-temporal & NOASSOM	66.4%	92.1%
NOASSOM+iDT	69.3%	93.8%

We set the activation function as $\mathcal{F} = \frac{1-e^{-0.3x}}{1+e^{-0.3x}}$. For the Fisher vector, we reduced the dimension of local features by half, namely $D = 200$. Next, GMM with $K(K = 256)$ was trained. Then, each video was denoted as a $2KD$-dimensional vector. Finally, we used a linear SVM to classify these actions with $C = 100$.

Evaluation of NOASSOM

We evaluated our method with different pipelines as shown in Table 1. Specifically, we evaluated the performance between ASSOM and NOASSOM, the effectiveness of spatial channel, temporal channel and fusion channel.

We compared the performance between ASSOM and NOASSOM on UCF-101 and HMDB-51 datasets. Since randomly sampled 3D volumes around the trajectories have no labels, the original supervised training method of ASSOM cannot work. So we removed the nonlinear orthogonal map ϕ from our NOASSOM to construct a new ASSOM method, the loss function of which turns to $E = \sum_{q=1}^{Q} \sum_{i=1}^{N} h_{c(i)}^{q} \frac{||\tilde{X}(i)||^2}{||X(i)||^2}$. We used gradient decent to minimize this loss function to learn b_h^q for training ASSOM without supervision. In this experiment the activated output of the fusion channel was used as the local feature. This method is denoted as "Spatio-temporal & ASSOM" in Table 1, and it achieves a relative good performance (UCF-101 87.4%, HMDB-51 62.9%) by our modified loss function and training method. Moreover, there are improvements by 3.5% and 4.7% on HMDB-51 and UCF-101 datasets respectively when we use NOASSOM, denoted as "Spatio-temporal & NOASSOM" in Table 1. Results demonstrate the effectiveness of our training method and nonlinear orthogonal map.

We evaluated the performance of spatial channel and temporal channel by respectively using the activated output of spatial channel and temporal channel in the second layer. The temporal channel outperforms the spatial channel by 4.9% and 4.6% on HMDB-51 and UCF-101 respectively. Results indicate that motion information has a higher discrimination than appearance information for action recognition. A similar result is obtained in Two-stream method (Simonyan and Zisserman 2014). In spatial and temporal channels, we respectively use NOASSOM to learn the patterns of two different types of data, images and video blocks. The results prove the generality and effectiveness of NOASSOM.

We used a three-layer NOASSOM to naturedly fuse these two channels of features by concatenating the activated output of spatial and temporal channels in the second layer as the input of the third layer. We obtained a high improvement compared with using the spatial and temporal channels separately. This demonstrates that our hierarchical NOASSOM effectively captures the motion and appearance information. Finally, we further fused the iDT features (Wang and Schmid 2013) by the fusion channel, namely additionally concatenating the iDT features as the input of the third layer. By the concatenated input, NOASSOM of the third layer was trained and it improved 2.9% and 1.7% performance on HMDB-51 and UCF-101 respectively. It can be seen that our hierarchical NOASSOM is able to extract and fuse multiple kinds of information for obtaining better representative features.

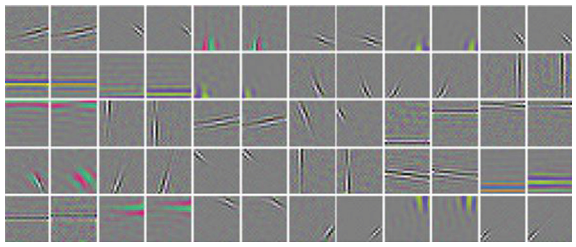


Figure 3: Learned appearance filters of NOASSOM on UCF-101. Each filter is the same as the size of the frame patch along the trajectory. The filters are represented by RGB values. The size of each filter is 20x20.

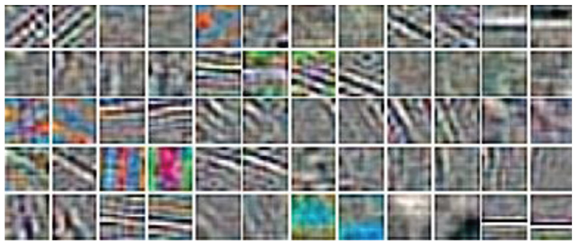


Figure 4: Learned motion filters of NOASSOM at time $t = 1$ in the 3D filters on UCF-101. Each 3D filter is the same as the size of the sub-block in the 3D volume around the trajectory. The size of the 3D filter is 16x16x10.

Visualization of NOASSOM

We visualized the learned basis vectors $\{b_h^q\}_{h=1}^2$ of the subspaces in the first layer on the spatial channel in Figure 3. In each row, two filters are grouped as the basis vectors of a subspace because we set $H = 2$. It can be seen that some filters are similar with the vertical or horizontal edge filters, which mainly extract the contour information. Another filters are more sensitive to the color and texture information. By these filters, appearance information along the trajectories are effectively extracted.

Similarly, we also visualized the learned basis vectors $\{b_h^q\}_{h=1}^2$ in the first layer on the temporal channel in Figure 4. Some of these filters vary in frequency and orientation, which are similar to Gabor filters. They are more sensitive to movement information. By these filters, motion information along the trajectories can be effectively extracted. Compared with the learnt filters on the spatial channel, the filters

Table 2: Mean accuracy of the state-of-the-art methods and our results on HMDB-51 and UCF-101. We respectively show the results of the hierarchal NOASSOM and our best results combing the NOASSOM features with iDT features.

HMDB-51		UCF-101	
STIP+BoVW (Soomro, Zamir, and Shah 2011)	23.0%	STIP+BoVW (Soomro, Zamir, and Shah 2011)	43.9%
Motionlets (Wang, Qiao, and Tang 2013)	42.1%	Deep Net(Karpathy et al. 2014)	63.3%
DT+BoVW (Wang et al. 2013)	46.6%	DT+VLAD (Cai et al. 2014)	79.9%
iDT+FV (Wang and Schmid 2013)	57.2%	iDT+FV (Wang and Schmid 2013)	85.9%
FstCN (Sun et al. 2015)	59.1%	iDT+HSV (Peng et al. 2014)	87.9%
Two Stream (Simonyan and Zisserman 2014)	59.4%	FstCN (Sun et al. 2015)	87.9%
iDT+HSV (Peng et al. 2014)	61.1%	Two Stream (Simonyan and Zisserman 2014)	88.0%
TDD+iDT (Wang, Qiao, and Tang 2015)	65.9%	TDD+iDT (Wang, Qiao, and Tang 2015)	91.5%
RNN+FV (Lev et al. 2016)	54.3%	RNN+FV (Lev et al. 2016)	88.0%
LTC (Varol, Laptev, and Schmid 2016)	64.8%	VLAD^3+iDT(FV) (Li, Li, and Vasconcelos 2016)	92.2%
Adascan+iDT (Kar et al. 2017)	61.0%	Adascan+iDT (Kar et al. 2017)	91.3%
TSN (Wang et al. 2016)	69.4%	TSN (Wang et al. 2016)	94.2%
ActionVLAD+iDT (Girdhar et al. 2017)	69.8%	ActionVLAD+iDT (Girdhar et al. 2017)	93.6%
DT+Hybrid architectures (de Souza et al. 2016)	70.4%	DT+Hybrid architectures (de Souza et al. 2016)	92.5%
ST-ResNet+iDT (Feichtenhofer, Pinz, and Zisserman 2016)	70.3%	ST-ResNet+iDT	94.6%
NOASSOM	**66.4%**	**NOASSOM**	**92.1%**
NOASSOM+iDT	**69.3%**	**NOASSOM+iDT**	**93.8%**

Table 3: Evaluation of hierarchal NOASSOM on KTH.

Algorithm	Accuracy
3D CNN(Ji et al. 2010)	90.2%
ISA(Le. et al. 2011a)	93.9%
DT+HOG/HOF/MBH(Wang et al. 2013)	94.2%
DT+MBH(Wang et al. 2013)	95.0%
DT+NOASSOM+BoVW	**97.9%**
NOASSOM+FV	**98.2%**

of temporal channel are more complex and irregular, which indicates that motion patterns are more complicated.

Visualizations of two channels prove that NOASSOM adaptively and effectively learn different filters to extract complementary patterns from the visual data. So the combination of spatial channel and temporal channel can be more effective, which are also proven in Table 1.

Comparison to the state of the art

In Table 2, we compared our methods with recent state-of-the-art methods on UCF-101 and HMDB-51. From the comparison results, we observe the following points. I) Our method outperforms all hand-crafted features based methods, such as (HoG, HoF, MBH, HSV) + iDT. The iDT+HSV method (Peng et al. 2014) uses different fusions of hand-crafted features to obtain the best performance among the hand-crafted features based methods, and we respectively outperforms the iDT+HSV method by 8.2% and 5.9% on HMDB-51 and UCF-101. II) ST-Resnet based on the Resnet and TSN based on multiple Two-Stream ConvNets using the end-to-end networks achieve higher performance than other methods. However, it is noted that our method also outperforms some end-to-end network based methods such as FstCN, Adascan+iDT, Two-stream. III) Our methods outperform the methods which use deep learning features as local features. Specifically, our method outperforms the TDD method which uses CNN local feature+iDT by 3.4% on HMDB-51, and outperforms it by 2.3% on UCF-101. Another deep learning local feature based method RNN+FV also has lower performance than our method.

We also tested our method on the small-scale dataset KTH in Table 3 by using the same pipeline (dense trajectory + BovW) and the pipeline in this paper. The results show that our method also has a superior performance on the small-scale dataset compared with the hand-crafted feature based methods and 3D CNN based method. In speed, we use fewer layers (3 layers) of NOASSOM compared with CNNs to extract local features and it has fewer parameters to learn. So the training time is faster than other deep CNNs based state-of-the-art methods.

Conclusions

We have proposed a new feature extraction method based on hierarchical NOASSOM for action recognition. The hierarchical NOASSOM has adaptively learnt effective feature filters of different hierarchies without supervision from images and video blocks. We combined the iDT features with our hierarchical NOASSOM and achieved the state-of-the-art results on the UCF-101 , HMDB-51 and KTH datasets. Since the proposed NOASSOM can be used as a generalized feature descriptor, we expect to apply it in more areas and investigate NOASSOM more deeply in the future work.

Acknowledgments

In this paper, the work is partly supported by the National Key Research & Development Plan of China (No. 2017YFB1002801), the 973 basic research program of China (Grant No. 2014CB349303), the Natural Science Foundation of China (Grant No. 61472420, 61472063, U1636218, 61472421) and the Strategic Priority Research Program of the CAS(Grant No. XDB02070003), and the CAS External cooperation key project.

References

Bhattacharya, S.; Kalayeh, M. M.; Sukthankar, R.; and Shah, M. 2014. Recognition of complex events: Exploiting temporal dynamics between underlying concepts. In *CVPR* 2243–2250.

Blank, M.; Gorelick, L.; Shechtman, E.; Irani, M.; and Basri, R. 2005. Actions as space-time shapes. In *ICCV*.

Cai, Z.; Wang, L.; Peng, X.; and Qiao, Y. 2014. Multi-view super vector for action recognition. In *CVPR*.

Dalal, N., and Triggs, B. 2005. Histogram of oriented gradients for human detection. In *CVPR* 2(886-893).

Dalal, N.; Triggs, B.; and Schmid, C. 2006. Human detection using oriented histograms of flowand appearance. In *ECCV*.

de Souza, C. R.; Gaidon, A.; Vig, E.; and Lopez, A. M. 2016. Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition. In *ECCV*.

Donahue, J.; Hendricks, L. A.; Guadarrama, S.; Rohrbach, M.; Venugopalan, S.; Darrell, T.; and Saenko, K. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR* 2625–2634.

Elman, J. L. 1990. Finding structure in time. In *Cognitive science* 14(2):179–211.

Feichtenhofer, C.; Pinz, A.; and Zisserman, A. 2016. Convolutional two-stream network fusion for video action recognition. In *CVPR*.

Girdhar, R.; Ramanan, D.; Gupta, A.; Sivic, J.; and Russell, B. 2017. Actionvlad: Learning spatio-temporal aggregation for action classification. In *CVPR*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. In *Neural computation* 9(8):1735–1780.

Ji, S.; Xu, W.; Yang, M.; and Yu, K. 2010. 3d convolutional neural networks for human action recognition. In *ICML*.

Ji, S.; Xu, W.; Yang, M.; and Yu, K. 2013. 3d convolutional neural networks for human action recognition. In *TPAMI* 35(1).

Kar, A.; Rai, N.; Sikka, K.; and Sharma, G. 2017. Adascan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In *CVPR*.

Karpathy, A.; Toderici, G.; Shetty, S.; Leung, T.; Sukthankar, R.; and Fei-Fei, L. 2014. Large-scale video classification with convolutional neural networks. In *CVPR*.

Klaser, A.; Marszalek, M.; and Schmid, C. 2008. A spatio-temporal descriptor based on 3d-gradients. In *BMVC*.

Kohonen, T.; Kaski, S.; and Lappalainen, H. 1997. Self-organized formation of various invariant-feature filters in the adaptive-subspace som. In *Neural Computation* 9(6):1321–1344.

Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; and Serre, T. 2011. Hmdb: a large video database for human motion recognition. In *ICCV*.

Laptev, I.; Marszalek, M.; Schmid, C.; and Rozenfeld, B. 2008. Learning realistic human actions from movies. In *CVPR*.

Laptev, I. 2005. On space-time interest points. In *IJCV* 64(2-3).

Le., Q. V.; Zou, W. Y.; Yeung, S. Y.; and Ng, A. Y. 2011a. 3d convolutional neural networks for human action recognition. In *ICML*.

Le, Q. V.; Zou, W. Y.; Yeung, S. Y.; and Ng, A. Y. 2011b. Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In *CVPR* 3361–3368.

Lev, G.; Sadeh, G.; Klein, B.; and Wolf, L. 2016. Rnn fisher vectors for action recognition and image annotation. In *ECCV*.

Li, Y.; Li, W.; and Vasconcelos, N. 2016. Vlad3: Encoding dynamics of deep features for action recognition. In *CVPR*.

Liu, Z. Q. 2002. Adaptive subspace self-organizing map and its application in face recognition. In *International Journal of Image and Graphics*.

Lowe, D. 1999. Object recognition from local scale-invariant features. In *ICCV*.

Matikaninen, P.; Hebert, M.; and Sukthankar, R. 2009. Trajectons: Action recognition through themotion analysis of tracked features. In *ICCV*.

Ng, J. Y.-H.; Hausknecht, M.; Vijayanarasimhan, S.; Vinyals, O.; Monga, R.; and Toderici, G. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR* 4694–4702.

Peng, B.; Zou, M.; Yan, H.; and Jabri, M. A. 1999. Handwritten digit recognition by adaptive-subspace self-organizing map. In *IEEE Trans. on Neural Networks*.

Peng, X.; Wang, L.; Wang, X.; and Qiao, Y. 2014. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. In *CoRR*.

Rodriguez, M.; Ahmed, J.; and Shah, M. 2008. Action mach: A spatiotemporal maximum average correlation height lter for action recognition. In *CVPR*.

Schuldt, C.; Laptev, I.; and Caputo, B. 2004. Recognizing human actions:a local svm approach. In *CVPR*.

Simonyan, K., and Zisserman, A. 2014. Two-stream convolutional networks for action recognition in videos. In *NIPS*.

Song, S.; Lan, C.; Xing, J.; Zeng, W.; and Liu, J. 2017. An end-to-end spatio-temporal attention model for human action recognition from skeleton data. In *AAAI*.

Soomro, K.; Zamir, A. R.; and Shah, M. 2011. Hmdb: A large video database for human motion recognition. In *UCF101: A dataset of 101 human actions classes from videos in the wild*.

Soomro, K.; Zamir, A. R.; and Shah, M. 2012. Ucf101: A dataset of 101 human actions classes from videos in the wild. In *CRCVTR-12-01*.

Sun, C., and Nevatia, R. 2013. Active: Activity concept transitions in video event classification. In *ICCV* 913–920.

Sun, L.; Jia, K.; Yeung, D.-Y.; and Shi, B. E. 2015. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*.

Varol, G.; Laptev, I.; and Schmid, C. 2016. Long-term temporal convolutions for action recognition. In *CoRRarXiv*.

Wang, H., and Schmid, C. 2013. Action recognition with improved trajectories. In *ICCV*.

Wang, H.; Klaser, A.; Schmid, C.; and Liu, C.-L. 2013. Dense trajectories and motion boundary descriptors for action recognition. In *IJCV* 103(1).

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; D; Lin, H.; Tang, X. O.; and Gool, L. V. 2016. 3d convolutional neural networks for human action recognition. In *ECCV*.

Wang, L.; Qiao, Y.; and Tang, X. 2013. Motionlets: Mid-level 3d parts for human motion recognition. In *CVPR*.

Wang, L.; Qiao, Y.; and Tang, X. 2015. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*.

Willems, G.; Tuytelaars, T.; and Gool, L. J. V. 2008. An efficient dense and scale-invariant spatio-temporal interest point detector. In *ECCV*.

Zhu, W.; Lan, C.; Xing, J.; and et al. 2016. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In *AAAI*.