Vis Comput (2016) 32:1395–1404
DOI 10.1007/s00371-015-1090-2

ORIGINAL ARTICLE

# Action recognition by learning temporal slowness invariant features

**Lishen Pei**[1] · **Mao Ye**[1] · **Xuezhuan Zhao**[2] · **Yumin Dou**[1] · **Jiao Bao**[1]

Published online: 22 April 2015


**Abstract** Deep learning approaches emphasized on learning spatio-temporal features for action recognition. Different to previous works, we separate the spatio-temporal feature learning unity into the spatial feature learning and the spatial/temporal feature pooling procedures. Using the temporal slowness regularized independent subspace analysis network, we learn invariant spatial features from sampled video cubes. To be robust to the cluttered backgrounds, we incorporate the denoising criterion to our network. The local spatio-temporal features are obtained by pooling features from the spatial and the temporal aspects. The key points are that we learn spatial features from video cubes and pool features from spatial feature sequences. We evaluate the learned local spatio-temporal features on three benchmark action datasets. Extensive experiments demonstrate the effectiveness of the novel feature learning architecture.

**Keywords** Action recognition · Temporal slowness regularization · Spatio-temporal features · Independent subspace analysis · Support vector machine

✉ Mao Ye
cvlab.uestc@gmail.com

[1] School of Computer Science and Engineering, Center for Robotics, Key Laboratory for NeuroInformation of Ministry of Education, University of Electronic Science and Technology of China, Chengdu 611731, People's Republic of China

[2] Chengdu Institute of Computer Application, Chinese Academy of Sciences, Chengdu 610041, People's Republic of China

## 1 Introduction

Action recognition has received much attention in recent years due to its wide applications, such as intelligent video surveillance, human–computer interaction, and smart home. Common approaches in action recognition rely on hand-crafted features, such as STIP [5,17,19], shape-motion features [28], HOG3D [15], human skeleton [11], and so on. Those approaches reported good performance on different action datasets. However, such hand-designed features cease to advance for a long time. At the same time, deep learning gets a great success in speech recognition and object recognition. The deep learning-based approaches [3,10,14,20,23,34] are also proved to be effective in action recognition.

For action recognition, those deep learning-based approaches learn features either in an unsupervised [7] setting, or a semi-supervised [35] setting. They take advantage of large amounts of unlabeled video data. Generally, the features of those deep learning-based approaches are extracted by convolving the learned 3D filters with action videos. For most of the learned spatio-temporal features, its spatial and temporal characters are united together.

Recently, some research works [6,32] that recognize actions from the still images have been done. They proved that spatial filters are effective in action recognition. In [40], with the temporal slowness constraint [4,22], effective spatial features are learned. It is robust to small translations and distortions. Motivated by those research, we try to verify the following two questions. Can the sequence of varying spatial features represent motion features? Is pooling spatial features effective in action recognition?

Experimental evidences [4,40] support that, associating low-level features which are appeared in coherent sequences, high-level visual representations become slow-changing and

tolerant toward non-trivial motion transformations. Temporal slowness analysis is a good approach for invariant feature learning. While performing actions, the clothing of the actors and the backgrounds are different. The denoising criterion [36] makes the learned features robust to costume and backgrounds. In order to recognize actions using spatial features, we propose a novel approach which individually learns the spatial features and pools the temporal features. The robust invariant spatial features are learned by combining the temporal slowness constraint and the denoising criterion.

At first, we densely sample video cubes from the videos, and normalize them using the local contrast normalization [9] method. Then with the regularization of temporal slowness and sparseness, a denoising independent subspace analysis (ISA) network is used to learn invariant and robust spatial features from the normalized video cubes. Thirdly, we adopt several pooling strategies to pool the learned spatial features. Then we use the bag-of-feature approach [37] to organize the pooled local features. Each action video is represented as a histogram of visual words. With the trained support vector machine of each action category, we recognize actions using the one-against-all classification strategy.

The contributions of the proposed approach are multifolds. First, with the regularization of temporal slowness and sparseness, the denoising independent subspace analysis network learns robust invariant spatial features. Second, using the feature pooling strategies, we pool effective spatial features to recognize actions. Third, through extensive experiments, we verify that good spatial features are also effective in action recognition.

The rest of this paper is organized as follows. At first, we introduce the related works in Sect. 2. Then we give a detailed description of the feature learning procedure in Sect. 3. In this section, we introduce the feature learning neural network and the feature pooling strategies. In Sect. 4, after introducing the benchmark datasets, we describe the test pipeline and the details of the experiments, and then the experiment results are reported and analyzed. At last, we conclude this paper in Sect. 5.

## 2 Related works

Recently, most action recognition research works are focused on spatio-temporal features. Many successful hand-designed features are extended from the image features, such as the Histograms of Oriented 3D spatio-temporal Gradients feature (HOG3D) [15], the Harris3D feature [18] and the extended speeded up robust feature (ESURF) [39]. Some other spatio-temporal features are assembled by the image features and the motion features. For example, HOG/HOF descriptor [17,19] combines the histogram of gradient orientations and the histogram of optic flow features. The shape-motion feature [12,28] assembles the shape descriptor and the motion descriptor. Those features are extracted according to the designed algorithms.

With the development of deep learning, some spatio-temporal feature learning approaches are proposed. Most of the current deep architectures for action recognition can be separated into two categories. One category learns spatio-temporal filters from the unlabeled video clips, then it trains traditional classifiers to recognize actions, such as [3,10,20]. The other category semi-supervised trains a classification neural network to recognize actions. Except the classifier layer, other layers of the neural networks are pre-trained using the unsupervised method. The representative work is reported in [34,38].

In [20], the authors learn the invariant spatial-temporal features via the independent subspace analysis network. In [3] and [34], the convolutional Restricted Boltzmann machine and the convolutional gated Restricted Boltzmann machine [25] are used to learn spatio-temporal features. The 3D convolutional neural network [10,38] is also applied to recognize actions.

Some research works [6,13,32] recognize actions from the still images. Those research declares that robust and effective spatial features contribute to action recognition. In [40], with the regularization of temporal slowness, spatial features are learned from videos. They get a consistent improvement in object classification while using the spatial features to recognize objects.

In [36], the implicit definition of a good representation is modified. They said that "A good representation is one that can be obtained robustly from a corrupted input and that will be useful for recovering the corresponding clean input." In their experiments, they achieve good performance by adding noise to the input data. It is obvious that good features are robust in action recognition.

Motivated by them, based on the denoising independent subspace analysis network and the temporal slowness constraint, we learn a set of robust and effective spatial features. Similar to many recent works [20,40], the network of this paper is extended from the ISA network. Le et al. [20] and Willems et al. [40] directly stack the ISA networks to learn invariant features. Different with them, we revised the ISA network, and constrained it with some different regularizations. Specially, we use it to learn spatial features, and the temporal property of the learned features is achieved by different pooling strategies.

## 3 Learning architecture

In this section, we introduce the feature learning architecture for action recognition. At first, we introduce the spatial feature learning module. It learns spatial features from video

cubes. In [40], the authors learn spatial features from videos and use those features to classify the objects. We use a similar network to learn spatial features from the action videos. Then the spatio-temporal features are extracted by different pooling strategies and those features are used to recognize actions.

### 3.1 Learning spatial features

The ISA network [8] is a two-layer network. Its first layer is a square non-linear network and the second layer is a square-root non-linear network. The weights $W$ of the first layer are learned, and the weights $V$ of the second layer are fixed. The fixed weights $V$ are used to represent the subspace structure of the neurons in the first layer. When training the weight $W$, we add some random noise to the input data. This is beneficial to learn invariant features. The architecture of the denoising network is shown in Fig. 1. In this figure, for the ISA structure, the size of the subspace is 2. Each red neuron looks at 2 blue neurons units. Based on the ISA network, we increase a layer to add noise to the input patterns for the new network.

The input data of this basic learning module are video cubes. While training the feature learning module, we randomly sample video cubes from the action videos. In order to reduce the correlation of the adjacent image pixels, we perform local contrast normalization (LCN) [9] for each video cube. After the processing of the LCN, the size of the normalized video cube is $n_w \times n_w \times T$. The normalized video cube is the input data of the learning module. It is called basic video cube. In this paper, the size of the basic video cube is fixed by setting $n_w = 24$ and $T = 5$.

For a basic video cube $x_s$ which indexed by $s$, we denote its image which is indexed by $t$ as $x_s^t$. By a feed-forward pass of the network in Fig. 1, the image patch $x_s^t$ is represented as $p_s^t$. For the unit $p_{si}^t$ of $p_s^t$, the formulation is shown as follows:

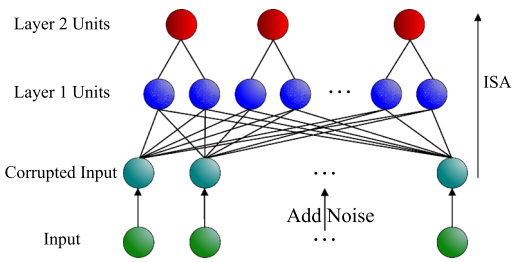



**Fig. 1** The neural network architecture of a denoising ISA network. The bottom bubbles are the input pattern $x^t$. Adding random noise to the input data, we get the corrupted input $\tilde{x}^t$. The *blue bubbles* are the simple units $z^t$, and the *red bubbles* are the pooling units $p^t$. In this figure, each pooling unit is pooled from two adjacent simple units

$$p_{si}^t = \sqrt{\sum_{k=1}^{M} V_{ik} \left( \sum_{j=1}^{N} W_{kj} \tilde{x}_{sj}^t \right)^2}, \tag{1}$$

where $M$ is the neural number of the first layer of the ISA network, $N$ is the pixel number of $x_s^t$, $V$ is the fixed weight that represents the subspace structure of the neurons in the first layer of the ISA network, $W$ is the learned weight of the first layer of the ISA network, and $\tilde{x}_{sj}^t$ is the corrupted value of the input pattern unit $x_{sj}^t$.

In Eq. 1, we use the corrupted data $\tilde{x}_s^t$ to replace the input pattern $x_s^t$. This is similar to Denoising Autoencoder [36]. As described in [36], good representations can be obtained robustly from corrupted inputs and that will be useful for recovering the corresponding clean inputs. Using the denoising criterion, we can learn rather stable and robust features. Action recognition is seriously affected by clothes,clutter background, and so on. The robust and stable features are helpful for action recognition.

As mentioned above, the number of the images of each basic video cube is denoted as $T$. For a basic video cube $x_s$, with the constraint of temporal slowness and sparseness, the loss function of reconstruction is defined as follows

$$\begin{aligned} \mathcal{L}_s(x_s; W) = & \sum_{t=1}^{T} \| x_s^t - W' W \tilde{x}_s^t \|_2^2 \\ & + \lambda \sum_{t=1}^{T-1} \| p_s^t - p_s^{t+1} \|_1 + \gamma \sum_{t=1}^{T} \| p_s^t \|_1, \end{aligned} \tag{2}$$

where $W'$ is the transposed matrix of $W$. The first term is the reconstruction error. It helps in avoiding features degeneracy. The second term is the temporal slowness constraint. It constrains the codes of the images to vary slowly. The third term is sparsity regularization. $\lambda$ and $\gamma$ are the corresponding regularization coefficients.

As shown in the second term of Eq. 2, temporal slowness asks the represented features to be slow-changing across time. In action videos, the object in the images of the sampled video cube has some small translations. We depict the images of an video cube in Fig. 2a. Using the temporal

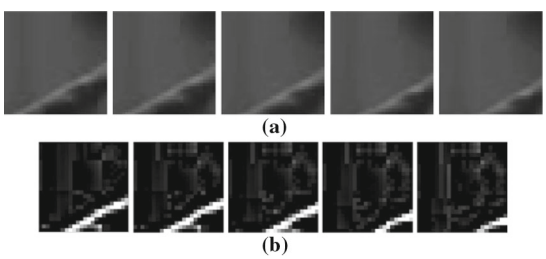



**Fig. 2** **a** Images of a video cube. It is obvious that the object is moving slowly in the video cube. **b** LCN processed images of the video cube. The size of the precessing kernel is $9 \times 9$

slowness constraint to learn features from such input data, the learned features are invariant to local translations. The last term of Eq. 2 is sparsity regularization. This criterion solves the degeneracy introduced by over-completeness. It helps to learn good features.

In order to learn the weights $W$, we solve the following minimization problem:

$$\min_{W} \sum_{s=1}^{S} \mathcal{L}_s(x_s; W), \tag{3}$$

where $S$ is the number of the training basic video cubes. Because the second term and the third term of Eq. 2 are $L_1$-norm regularization, the objective function is non-differentiable. However, this is an $L_1$-regularized least square problem while optimizing over $W$. We solve this optimization problem using the custom-designed solvers [1,21].

### 3.2 Feature visualization

After unsupervised training of the basic video cubes, we visualize the learned spatial features. As the size of the basic video cube is $n_w \times n_w \times T$, the learned spatial feature has a size of $n_w \times n_w$. In Fig. 3, we display three groups of spatial features. Those features are learned using different neural networks or with different constraints.

As comparison, we display the spatial features that learned by the traditional ISA network with sparse regularization in the left column of Fig. 3. This network is also used to learn 3D spatio-temporal features in [20]. As shown, those features are sparse. Based on this learning architecture, we constrain the features using the temporal slowness regularization, and the learned spatial features are shown in the middle column. From this column, we can find that the features have more orientation properties. In the last column, the features are learned using the denoising ISA with the sparse regularization and the temporal slowness constraint. The features of this group are sparse and they also have orientation properties. In order to evaluate the learned features of the three neural networks, we use those features to recognize actions respectively in the experiment section.

### 3.3 Pooling features

As described above, we learn spatial features from the basic video cubes. In order to make the spatial features have the same effectiveness as the spatio-temporal features in action recognition, we pool the spatial features from the temporal and the spatial aspects. In this section, we introduce feature pooling strategies.

At first, let us think about how actions are represented in action videos. In computer vision, action is an image sequence of slowly changed actor postures. Theoretically, sequences of varying spatial features can represent motion features. So, with basic spatial feature units, we can represent spatio-temporal features or human motions.

In order to pool spatio-temporal features, we sample large-scale video cubes which are bigger than basic video cube. The video cubes are sampled using the overlapped dense sampling approach. After LCN processing, we split the large video cube into several basic video cubes. Then the image patches of each basic video cube are encoded by the ISA network. Denote one image of a basic video cube as $x_s^t$. Based on the trained neural network, without the corrupt processing, the image patch $x_s^t$ is encoded as $p_s^t$ using Eq. 1.

#### *3.3.1 Temporal pooling*

With the extracted spatial features, we pool features using the temporal pooling strategy. In Fig. 4, we depict the temporal

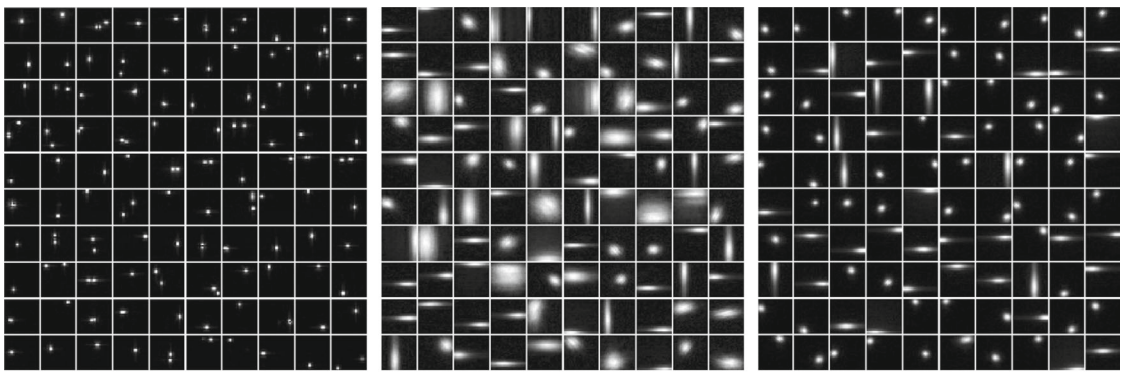



**Fig. 3** Visualization of the learned spatial features (patch size $24 \times 24$). The *left column* is the features which are learned using the traditional ISA network with the sparse constraint. The *middle column* is the learned features which are based on the ISA network with the temporal slowness constraint and the sparse regularization. The *right column* is learned using the denoising ISA network with the constraint of temporal slowness and sparse

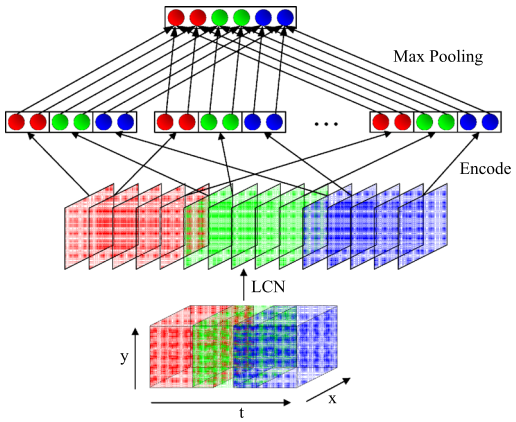



**Fig. 4** Temporal Pooling. At first, we normalize the sampled large video cube using the LCN method, and split it into three basic video cubes. Those basic video cubes are partially overlapped. Then the five image patches of each basic video cube are encoded by the feature learning architecture. At last, the large video cube is represented by the result of max pooling the spatial codes

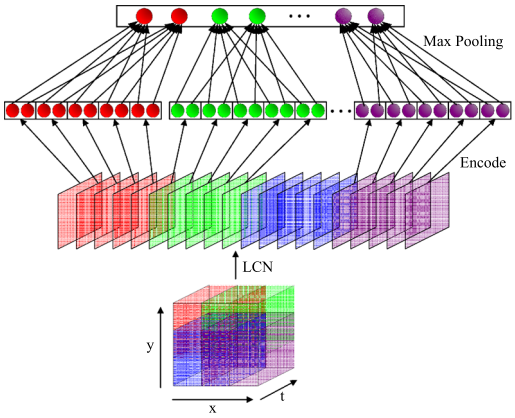



**Fig. 5** Spatial pooling. At first, we normalize the big video cube using the LCN method, and split it into four basic video cubes. Those basic video cubes are partially overlapped. Then we encode each image using the trained neural network. At last, the video cube is represented by max pooling the codes

feature pooling procedure. In this figure, the spatio-temporal features are pooled from $nt$ basic video cubes. For temporal pooling, $nt = 3$. At first, let us determine the size of the originally normalized big action video cube. As depicted, after local contrast normalization, the big video cube is split into three basic video cubes in the temporal axis. Those basic video cubes are overlapped. The overlapped size is half of the temporal length of the basic video cube. For $T = 5$, the two image patches are overlapped. The temporal length of the big video cube is $T + (T-2) \times (nt-1) = 11$. The pixel size of the big action video cube is $n_w \times n_w \times 11$ ($n_w = 24$). Considering that it is the size of the LCN processed action video, and the window size of the normalization kernel is 9, the size of the randomly sampled video cube is $32 \times 32 \times 11$ ($n_w + 8 = 32$).

Taking Fig. 4 as an example, we give a detail description of temporal pooling procedure. At first, we densely sample video cubes with the determined size. After LCN processing, we split each video cube into $nt$ cubes. They are denoted as $x_s$, ($s = 1, 2, \ldots, nt$). With the trained denoising ISA network, we encode image patch $x_s^t$ as $p_s^t$. Representing the codes as feature vectors $p_o^t = (p_1^t, p_2^t, \ldots, p_{nt}^t)$, ($t = 1, \ldots, T$), we pool features using the following formulation:

$$\hat{p} = \max(p_o^1, p_o^2, \ldots, p_o^T) \tag{4}$$

$$= (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{nt}). \tag{5}$$

As shown in Fig. 1, we set the output dimension of the denoising ISA network as $d$. So $\hat{p}_s = (\hat{p}_{s1}, \hat{p}_{s2}, \ldots, \hat{p}_{sd})$, $s = (1, 2, \ldots, nt)$. $\hat{p}$ is the output of the temporal pooling architecture. At last, each sampled video cube is represented as a feature vector $\hat{p}$ as described in Fig. 4. As shown in Fig. 4, the dimension of the output local features is $nt \times d$.

### 3.3.2 Spatial pooling

Similar to the temporal pooling strategy, the spatial pooling strategy also needs to sample large-scale video cubes. Different to temporal pooling, the sampled video cubes have a larger spatial size. In Fig. 5, after LCN processing, the large video cube is split into $ns \times ns$ basic input video cubes in the spatial plane. For spatial pooling, we set $ns = 2$. Those basic video cubes have an overlap in half of the spatial size of the basic video cube. So the size of the normalized large video cube is $(24 + (ns-1) \times 12) \times (24 + (ns-1) \times 12) \times T$. Because the window size of the normalization kernel is 9. The size of the randomly sampled video cube is $44 \times 44 \times T$

As shown in Fig. 5, after LCN processing, the sampled video cube is split into 4 basic video cubes. They are denoted as $x_s$,($s = 1, 2, \ldots, ns \times ns$). Then the image patch $x_s^t$ of $x_s$ is encoded as $p_s^t$. For each basic video cube, we pool the codes of the image patches using the following formulation:

$$\hat{p}_s = \max(p_s^1, p_s^2, \ldots, p_s^T). \tag{6}$$

Then we concatenate the pooled features $\hat{p}_s$ of the basic video cubes to represent the sampled video cube.

$$\hat{p} = (\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_{ns \times ns}), \tag{7}$$

The output feature vector of Fig. 5 is $\hat{p}$. The dimension of the output local features is $ns \times ns \times d$.

### 3.3.3 Spatial and temporal pooling

In order to extract effective features to recognize actions, we combine the spatial and the temporal pooling strategies to extract features. At first, correspondingly, with the number

of the split basic video cubes $ns \times ns \times nt$, the size of the randomly sampled video cube is determined as $(24 + (ns - 1) \times 12 + 8) \times (24 + (ns - 1) \times 12 + 8) \times T + (T - 2) \times (nt - 1)$. Then we densely sample video cubes with the determined cube size. After feature extraction, the dimension of the pooled feature is $ns \times ns \times nt \times d$. For sampled big video cubes that the value of $ns \times ns \times nt$ is very big, the dimension of the extracted feature is very high. We reduce the dimension of the features using the PCA approach. In literature [33], those sampling strategies have the best action recognition performance, while the dimension of the feature set is 864. For the pooled features whose dimension is higher than 864, we reduce the dimension of the feature to 864 using the PCA dimensionality reduction algorithm.

## 4 Experiments

In this section, we evaluate the extracted local features on three datasets. For the three variants of the ISA network, their features are compared for action recognition. Comparison of our algorithm and the state-of-the-art action recognition approaches is also reported. After introducing three public action datasets, we briefly describe the test pipeline. In order to make a comparison with most of the algorithms on the same basis, we use the identical test pipeline with [37] to recognize actions. Then in the following subsection, we describe the details of the experiments. At last, we report the control experiment results of tuning the dimension of the learned spatial feature and the size of the spatial/temporal pooling operation. The comparison experiments are also displayed and analyzed.

### 4.1 Datasets

Our experiments are carried out on three benchmark action datasets. The actions of the three datasets are different to each other. One is a scene behavior set that is collected from the Hollywood films. It is Hollywood2 action dataset [24]. The second is the KTH action dataset [31]. It is a daily action dataset which is captured in the experimental constraint environment. The third is the UCF sport action dataset [30] which is collected from various broadcast sports channels.

#### *4.1.1 Hollywood2 action dataset*

The Hollywood2 action dataset contains 12 human actions. They are "answer phone," "drive car," "eat," "fight person," "get out car," "hand shake," "hug person," "kiss," "run," "sit down," "sit up," and "stand up." The action clips are extracted from 69 movies. Each action class contains approximately 150 samples. In our experiments, we use the training set to train the feature learning module. The training set contains 823 action clips, and the testing set includes 884 action sequences. The training and the testing action clips are extracted from different movies. Due to the limitation of the computer memory, we down-sample the original Hollywood2 action sequence to its half spatial resolution.

#### *4.1.2 KTH action dataset*

The KTH actions dataset contains six categories of human actions. They are "waking," "jogging," "running," "boxing," "hand waving," and "hand clapping" which are performed by 25 subjects in four different scenarios (outdoors, outdoors with scale variation, outdoors with different clothes, and indoors). This database includes 2391 video sequences. All of the action sequences were taken over homogeneous backgrounds. We use the original experimental setup of [31] to perform experiments. The KTH actions dataset is split into the training set (16 subjects) and the testing set (9 subjects).

#### *4.1.3 UCF sport action dataset*

The UCF Sports action dataset includes ten sport actions. The actions are "diving," "kicking," "lifting," "riding," "running," "walking," "skateboarding," "swing-1," "swing-2," and "swing-3." The sport videos are captured in cluttered background with a wide range of viewpoints. This dataset contains 150 video samples. In our experiments, we use the split experiment setup. We split the dataset into 103 training examples and 47 testing examples as did in [16,29]. According to [29], this separation minimizes the background correlation for the training set and the testing set. In order to increase the number of the training examples, we extend the training set by adding a horizontally flipped version of each training example video. Due to the limitation of the computer memory, we down-sample the original UCF sports action sequence to its half spatial resolution.

### 4.2 Test pipeline

To evaluate the performance of the learned local features, following [37], we use the bag-of-feature based pipeline to recognize actions. This makes a fair comparison with most state-of-the-art algorithms. At first, we extract features from the action sequences, and represent each sequence as a bag of local features. Then we use the bag-of-feature representation to train non-linear support vector machines [2] with $\chi^2$-kernel [19]. For the multi-class action recognition, we adopt the one-against-rest strategy to recognize actions.

For the bag-of-feature representation of the action videos, we densely sample video cubes using the computed cube size. Each video cube is encoded as a feature vector using the proposed network architecture. For the KTH action dataset, its actions are captured from homogeneous backgrounds. In

order to improve the action recognition performance, we use the norm-thresholding method [20] to choose the densely sampled video cubes. Then the feature vocabulary is constructed by clustering the encoded feature vectors using the k-means clustering algorithms. Each cluster group is taken as a visual word. In our experiments, the number of the visual words $V$ is set as 3000. With the constructed vocabulary, we assign each feature vector to its closest visual word using the Euclidean distance metric. Based on this, each action video is represented as a frequency histogram over the visual words. This is a usual practice for local features.

In our experiments, for the Hollywood2 action dataset, the experiment performance is evaluated by computing the mean average precision (mean AP) over all action classes which is suggested by [24]. For the UCF sport and the KTH action datasets, we compute their average action recognition accuracy to evaluate our approach. The reason of using two evaluating measures is that different measures are reported on these datasets in previous literatures.

### 4.3 Details of the experiments

Three experiments are performed to choose the feature learning architecture, to tune the spatial feature dimension and the spatial/temporal pooling size. At the same time, the performance of the three variants of the ISA network is compared. Comparisons with the state-of-the-art action recognition algorithms are also reported. Extensive experiments confirm the efficiency of the proposed algorithm. All of the experiments are carried out on the Hollywood2, the KTH and the UCF sport action datasets. In this subsection, we introduce the detailed information of each experiment. The experiment results and the analyses are shown in the following subsection.

While learning spatial features based on the original ISA network, we learned three groups of spatial features. They are learned from the original ISA network, the temporal slowness regularized ISA network and the temporal slowness regularized denoising ISA network. Obviously, the learned features are different to each other. To choose the most efficient feature learning approach, we compare the three groups of features on action recognition with the same basis. The output dimension $d$ of the three networks is set as 100. For this experiment, while sampling video cubes, we set $ns = 1$ and $nt = 3$. So, the normalized size of the randomly sampled video cube is fixed as $24 \times 24 \times 11$. Each sampled video cube is split into three overlapped basic video cubes. After training those three neural networks, we extract features from video cubes. The dimension of the obtained local feature is $1 \times 1 \times 3 \times 100 = 300$.

To measure the influence of the parameters on action recognition, we perform extensive experiments to tune the parameters. While learning features, the number of the basic features $d$ can be set arbitrarily to validate whether it affects the action recognition performance. We also carry out some experiments to verify the influence of the number of the basic features $d$. Then we choose the best value of $d$ and fix it. While performing experiments to choose the appropriate value for $d$, we set the normalized video cube as $24 \times 24 \times 11$. For spatial pooling and temporal pooling, the tunable parameters are the temporal length $nt$ and the spatial size $ns$. To pool features from the spatial and temporal aspects, each randomly sampled larger video cube is split into $ns \times ns \times nt$ overlapped basic video cubes. We perform a large set of experiments with different values of $ns \times ns \times nt$.

At last, based on the above experiments, we use the optimized parameters to recognize actions. The experiment results are compared with the state-of-the-art algorithms on three benchmark action datasets. The compared hand-crafted features are HOG3D [15], HOG/HOF [19], ESURF [39], and so forth. As local feature, those features are widely applied to recognize actions in various algorithms. Most of them get good performance. Besides, we compare our algorithm with some deep learning approaches, such as the space–time deep belief network method [3], the convolutional learning algorithm [34], and the 3D convolutional neural network approach [10]. These methods have good action recognition performance and the algorithm accuracies on some of the three datasets are reported. The comparison experiment results confirm the effectiveness of our algorithm.

### 4.4 Results and analyses

In this section, at first, we successively present the results of the experiments on three action datasets. Then we analyze them respectively. In Tables 1 and 2, we compare the action recognition performance of the ISA, the denoising ISA, and the temporal slowness regularized denoising ISA networks.

**Table 1** Comparison of the mean AP on the Hollywood2 dataset with different feature learning structures

| Mean AP | Hoolywood2 (%) |
|---|---|
| ISA | 36.2 |
| Denoising ISA | 38.2 |
| Denoising ISA + slowness | 39.4 |

**Table 2** Comparison of the average accuracy on the KTH dataset and the UCF sports action data set with different feature learning structures

| Average Accuracy | KTH (%) | UCF (%) |
|---|---|---|
| ISA | 88.0 | 79.0 |
| Denoising ISA | 88.2 | 79.3 |
| Denoising ISA + slowness | 88.6 | 81.6 |

**Table 3** Mean AP of action recognition with different values of $d$ on the Hoolywood2 action dataset. $d$ is the number of the learned basic features

| Mean AP | Hoolywood2 (%) |
|---|---|
| $d = 50$ | 38.2 |
| $d = 100$ | 39.4 |
| $d = 150$ | 39.7 |

**Table 4** Action recognition accuracy with different values of $d$ on the KTH and UCF sports action datasets. $d$ is the number of the learned basic features

| Accuracy | KTH (%) | UCF (%) |
|---|---|---|
| $d = 50$ | 88.0 | 79.0 |
| $d = 100$ | 88.6 | 81.6 |
| $d = 150$ | 88.2 | 81.4 |

**Table 5** Mean AP of action recognition with different values of $ns \times ns \times nt$ for the hollywood2 action dataset. $ns \times ns \times nt$ is the number of the basic video cubes for each random sampled video cube

| Mean AP | Hoolywood2 (%) |
|---|---|
| ST pooling $1 \times 1 \times 3$ | 39.4 |
| ST pooling $2 \times 2 \times 1$ | 42.9 |
| ST pooling $2 \times 2 \times 3$ | 43.9 |
| ST pooling $3 \times 3 \times 4$ | 41.3 |

The learned features of the three architectures are shown in Fig. 3. From those two tables, we find that the features learned by the temporal slowness regularized denoising ISA network have the best results. The features that are learned by the denoising ISA network have better action recognition performance than the features of the ISA network. The reasons are that the denoising constraint enforces the learned features invariant to noises, and the temporal slowness constraint makes the features more robust.

In Tables 3 and 4, we report the experiment results with different values of $d$ on the Hollywood2, the KTH and the UCF sports action datasets. From Table 3, we can find that we get good action recognition performance while $d = 100$. Although the experiment has better performance while $d = 150$, the dimension of the final feature is very high and the mean AP only increased a little. From Table 4, we find that the experiment get better action recognition performance while $d = 100$. With the fixed value 100 of $d$, we perform the following experiments. Based on the spatial and temporal pooling strategy, we split the randomly sampled video cubes into several basic video cubes. The experiment results with different number of the basic video cubes are shown in Tables 5 and 6. From those tables, we conclude that the experiments get worse performance, while the number of the basic video cubes of each sampled video cube is too little or too big. The reason is that the number of the basic video cubes of each sampled video cube determines the size of the sampled cube. The size of the randomly sampled video cube is very important for action recognition.

**Table 6** Action recognition accuracies with different values of $ns \times ns \times nt$ for the KTH and UCF sports action dataset. $ns \times ns \times nt$ is the number of the basic video cubes for each random sampled video cube

| Accuracy | KTH (%) | UCF (%) |
|---|---|---|
| ST pooling $1 \times 1 \times 3$ | 88.6 | 81.6 |
| ST pooling $2 \times 2 \times 1$ | 89.0 | 82.6 |
| ST pooling $2 \times 2 \times 3$ | 90.0 | 85.6 |
| ST pooling $3 \times 3 \times 4$ | 88.7 | 82.9 |

**Table 7** Mean Ap comparison on the Hollywood2 action dataset

| Algorithm | Mean Ap (%) |
|---|---|
| Action context [24] | 35.5 |
| HOG + KM + SVM [34] | 39.4 |
| Hessian + HOG3D [37] | 41.3 |
| Dense HOG [37] | 39.4 |
| Hessian + ESURF [37] | 38.2 |
| Our method | 43.9 |

**Table 8** Average accuracy comparison on the KTH action dataset

| Algorithm | Accuracy (%) |
|---|---|
| pLSA [26] | 83.3 |
| HOG/HOF [37] | 86.1 |
| HOG3D [37] | 85.3 |
| ST-DBN [3] | 86.6 |
| GRBM [34] | 90.0 |
| 3DCNN [10] | 90.2 |
| ACTION STATE [27] | 88.8 |
| Our method | 90.0 |

**Table 9** Average accuracy comparison on the UCF sports action dataset

| Algorithm | Accuracy (%) |
|---|---|
| HOG3D [37] | 82.9 |
| HOF [37] | 82.6 |
| HOG/HOF [37] | 79.3 |
| ACTION STATE [27] | 85.4 |
| Our method | 85.6 |

At last, a comparison of our method and the state-of-the-art action recognition algorithms for the Hollywood2, the KTH, and the UCF sports action datasets are reported in Tables 7, 8, and 9. With the best parameters, we report the comparison experiments on three benchmark action datasets. From Tables 7, 8, and 9, we can discover that our algorithm outperforms most of the recently published approaches.

## 5 Conclusion

We propose a local spatio-temporal feature learning approach. This approach separates the spatial feature learning course and the spatio-temporal feature pooling procedure. We use the denoising independent subspace analysis network with the temporal slowness constraint to learn robust spatial features. The spatio-temporal features are formed by processing the spatial features using the spatial/temporal pooling strategies. The learned features are invariant to small translation and robust to background clutters.

In order to validate the effectiveness of the feature learning architecture, we performed extensive experiments. Three groups of spatial features are learned using different methods, and we compared their action recognition performance. The parameter values of the feature learning architecture are turned to recognize actions. Extensive experiments on the Hollywood2 action dataset, the KTH action dataset, and the UCF sports action dataset confirm the effectiveness of the proposed algorithm.

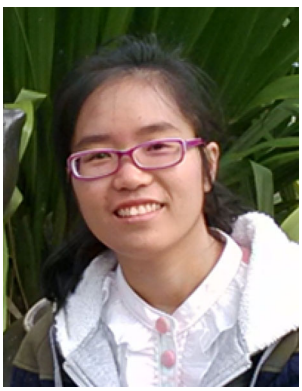
Our work verified the two questions which is proposed in the introduction section. The sequence of varying spatial features can represent motion features. Robust spatial features are effective in action recognition. This paper demonstrates that, with spatial and temporal pooling, robust spatial features are valid in action recognition. In our experiments, we find that the performance of the spatial features greatly depends on the pooling strategies. In our future work, we will devote our efforts to weighted pooling strategies.

**Acknowledgments** This work was supported in part by the National Natural Science Foundation of China (61375038).

## References

1. Andrew, G., Gao, J.: Scalable training of $l_1$-regularized log-linear models. In: International conference on Machine Learning, pp. 33–34 (2007)
2. Chang, C., Lin, C.: Libsvm : a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27 (2011)
3. Chen, B., Ting, J.A., Marlin, B., de Freitas, N.: Deep learning of invariant spatio-temporal features from video. In: Workshop of Neural Information Processing Systems (2010)
4. Cox, D., Meier, P., Oertelt, N., Dicarlo, J.: 'Breaking' position-invariant object recognition. Nat. Neurosci. **8**(9), 1145–1147 (2005)
5. Dawn, D.D., Shaikh, S.H.: A comprehensive survey of human action recognition with spatio-temporal interest point (STIP) detector. Vis. Comput. (2015). doi:10.1007/s00371-015-1066-2
6. Delaitre, V., Laptev, I., Sivic, J.: Recognizing human actions in still images: a study of bag-of-features and part-based representations. In: British Machine Vision Conference (2010)
7. Larochelle, H., Erhan, D., Courville, A., Bergstra, Bengio, B.: An empirical evaluation of deep architectures on problems with may factors of variation. In: IEEE International Conference on Machine Learning, New York, ACM, pp. 473–480 (2007)
8. Hyvarinen, A., Hurri, J., Hoyer, P.: Natural Image Statistics. Springer, Heidelberg (2009)
9. Jarrett, K., Kavukcuoglu, K., Ranzato, M., LeCun, Y.: What is the best multi-stage architecture for object recognition? In: IEEE International Conference on Computer Vision (2009)
10. Ji, S., Xu, W., Yang, M., Yu, K.: 3D convolutional neural networks for human action recognition. In: IEEE International Conference on Machine Learning, pp. 3212–3220 (2012)
11. Jiang, X., Zhong, F., Peng, Q., Qin, X.: Online robust action recognition based on a hierarchical model. Vis. Comput. **30**, 1021–1033 (2014)
12. Jiang, Z., Lin, Z., Davis, L.S.: Recognizing human actions by learning and matching shape-motion prototype trees. IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 533–547 (2012)
13. Karlinsky, L., Dinerstein, M., Ullman, S.: Using body-anchored priors for identifying actions in single images. In: IEEE Conference on Neural Information Processing Systems (2010)
14. Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., Fei-Fei, L.: Large-scale video classification with convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (2014)
15. Kläser, A., Marszalek, M., Schmid, C.: A spatio-temporal descriptor based on 3D gradients. In: British Machine Vision Conference (2008)
16. Lan, T., Wang, Y., Mori, G.: Discriminative figure-centric models for joint action localization and recognition. In: IEEE International Conference on Computer Vision (2011)
17. Laptev, I.: On space-time interest points. IEEE Int. J. Comput. Vis. **64**, 107–123 (2005)
18. Laptev, I., Lindeberg, T.: Space-time interest points. In: IEEE International Conference on Computer Vision (2003)
19. Laptev, I., Marszalek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
20. Le, Q.V., Zou, W.Y., Yeung, S.Y., Ng, A.Y.: Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3361–3368 (2011)
21. Lee, H., Battle, A., Raina, R., Ng, A.Y.: Efficient sparse coding algorithms. Neural Inf. Process. Syst. **19**, 801–808 (2006)
22. Li, N., Dicarlo, J.J.: Unsupervised natural experience rapidly alters invariant object representation. Science **321**, 1502–1507 (2008)
23. Liang, X., Lin, L., Cao, L.: Learning latent spatio-temporal compositional model for human action recognition. In: Proceedings of the 21st ACM International Conference on Multimedia, pp. 263–272 (2013)
24. Marszalek, M., Laptev, I., Schmid, C.: Actions in context. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2929–2936 (2009)
25. Memisevic, R., Hinton, G.: Unsupervised learning of image transformations. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8 (2008)
26. Niebles, J., Wang, H., Fei-Fei, L.: Unsupervised learning of human action categories using spatial-temporal words. Int. J. Comput. Vis. **79**(3), 299–318 (2008)
27. Pei, L., Ye, M., Xu, P., Li, T.: Fast multi-class action recognition by querying inverted index tables. Multimed. Tools Appl. (2014). doi:10.1007/s11042-014-2207-8
28. Pei, L., Ye, M., Xu, P., Zhao, X., Li, T.: Multi-class action recognition based on inverted index of action states. In: IEEE International Conference on Image Processing (2013)
29. Raptis, M., Kokkinos, I., Soatto, S.: Discovering discriminative action parts from mid-level video representations. In: IEEE Conference on Computer Vision and Pattern Recognition (2012)
30. Rodriguez, M., Ahmed, J., Shah, M.: Action mach: a spatio-temporal maximum average correlation height filter for action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 3361–3366 (2008)

31. Schüldt, C., Laptev, I., Caputo, B.: Recognizing human actions: a local svm approach. In: IEEE International Conference on Pattern Recognition, pp. 32–36 (2004)
32. Sharma, G., Jurie, F., Schmid, C.: Expanded parts model for human attribute and action recognition in still images. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)
33. Shi, F., Petriu, E., Laganière, R.: Sampling strategies for real-time action recognition. In: IEEE Conference on Computer Vision and Pattern Recognition (2013)
34. Taylor, G.W., Fergus, R., LeCun, Y., Bregler, C.: Convolutional learning of spatio-temporal features. In: European Conference on Computer Vision, pp. 140–153 (2010)
35. Nair, V., Hinton, G.: 3D object recognition with deep belief nets. In: Neural Information Processing Systems, pp. 1339–1347 (2009)
36. Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., Manzagol, P.A.: Stacked denoising autoencoders: learning useful representations in a deep network with a local denoising criterion. J. Mach. Learn. Res. **11**, 3371–3408 (2010)
37. Wang, H., Ullah, M.M., Kläser, A., Laptev, L., Schmid, C.: Evaluation of local spatio-temporal features for action recognition. In: British Machine Vision Conference (2010)
38. Wang, K., Wang, X., Lin, L., Wang, M., Zuo, W.: 3D human activity recognition with reconfigurable convolutional neural networks. In: Proceedings of the ACM International Conference on Multimedia (2014)
39. Willems, G., Tuytelaars, T., Gool, L.: An efficient dense and scale-invariant spatio-temporal interest point detector. In: European Conference on Computer Vision (2008)
40. Zou, W.Y., Zhu, S., Ng, A.Y., Yu, K.: Deep learning of invariant features via simulated fixations in video. In: IEEE Conference on Neural Information Processing Systems, pp. 3212–3220 (2012)

**Lishen Pei** received the B.S. degree in computer science and technology from Anyang Teachers College, Anyang, China, in 2010. She has been taking the successive master-doctor program since September 2010. She is currently a Ph.D. candidate in University of Electronic Science and Technology of China, Chengdu, China. Her current research interests include action detection and action recognition in computer vision.

**Mao Ye** received the B.S. degree in mathematics from Sichuan Normal University, Chengdu, China, in 1995, the M.S. degree in mathematics from University of Electronic Science and Technology of China, Chengdu, China, in 1998, and the Ph.D. degree in mathematics from Chinese University of Hong Kong, in 2002. He is currently a professor and Director of CVLab. His current research interests include machine learning and computer vision. In these areas, he has published over 70 papers in leading international journals or conference proceedings.

**Xuezhuan Zhao** received the B.S. degree in computer science and technology from Anyang Teachers College, Anyang, China, in 2010. He is in Chengdu computer application research institute of the Chinese academy of sciences for the Ph.D. degree. His current research interests include object detection, object tracking, abnormal behavior detection and analysis in computer vision.

**Yumin Dou** received his master's degree in Computer Science from Chengdu University of Technology in 2010, He is now a Ph.D. candidate of the School of Computer Engineering and Technology of University of Electronic Science and Technology University of China. His research interests include pattern recognition and computer vision.

**Jiao Bao** received the B.S. degree in computer science and technology from Nanyang Teachers College, Nanyang, China, in 2008, the M.S. degree in mathematics from University of Electronic Science and Technology of China, Chengdu, China, in 2011, She is currently a Ph.D. candidate in University of Electronic Science and Technology of China, Chengdu, China. Her current research interests include face recognition and pretreatment process of personal face image.