JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 35, 851-870 (2019)
DOI: 10.6688/JISE.201907_35(4).0009

# Trajectory-Based 3D Convolutional Descriptors for Human Action Recognition*

SHEERAZ ARIF[1], JING WANG[1,+], FIDA HUSSAIN[2] AND ZESONG FEI[1]
[1]*Department of Information and Communication Engineering*
*School of Information and Electronics*
*Beijing Institute of Technology*
*Beijing, 100081 P.R. China*
*E-mail: {Sheeraz.arif; wangjing+; feizesong}@bit.edu.cn*
[2]*School of Electrical and Information Engineering*
*Jiangsu University*
*Zhenjiang, 212013 P.R.China*
*E-mail: fidahussain@ujs.edu.cn*

This article presents a new method for video representation, called trajectory based 3D convolutional descriptor (TCD), which incorporates the advantages of both deep learned features and hand-crafted features. We utilize deep architectures to learn discriminative convolutional feature maps, and conduct trajectory constrained pooling to aggregate these convolutional features into effective descriptors. Firstly, valid trajectories are generated by tracking the interest points within co-motion super-pixels. Secondly, we utilize the 3D ConvNet (C3D) to capture both motion and appearance information in the form of convolutional feature maps. Finally, feature maps are transformed by using two normalization methods, namely channel normalization and spatiotemporal normalization. Trajectory constrained sampling and pooling are used to aggregate deep learned features into descriptors. The proposed (TCD) contains high discriminative capacity compared with hand-crafted features and is able to boost the recognition performance. Experimental results on benchmark datasets demonstrate that our pipeline obtains superior performance over conventional algorithms in terms of both efficiency and accuracy.

***Keywords:*** human action recognition (HAR), deep learning, trajectory feature, hybrid featured, super-pixel

## 1. INTRODUCTION

Over the last decade, Human Action Recognition (HAR) has been one of the enabling component behind video understanding and Human-Computer Interaction (HCI) with huge pool of potential applications. The applications include: video indexing, video content analysis, video surveillance, video entertainment, Ambient-Assisted Living (AAL), abnormal behaviors detection, and intelligent driving. Despite receiving much attention from the research community and achieving effective results, HAR remains as a challenging task when focusing on realistic datasets collected from web videos, TV shows and movies. There are so many intra-class variations caused by partial occlusion, the presence of background clutter, camera motion, various motion styles and viewpoint changes. Meanwhile, videos with low resolution and high dimensionality may restrict recognition problem. To address the aforementioned issues, there is immense need of effective visual representations.

Received January 31, 2018; revised May 21 & July 27, 2018; accepted July 26, 2018.
Communicated by Chu-Song Chen.
+ Corresponding author.
* This research work was supported by Multimedia Tech Lab funded by Beijing Institute of Technology, China.

We can break down the pipeline of HAR into three main steps: feature extraction, encoding and classification. While for the classification part, the existing techniques are more mature, but for feature extraction and encoding there is still a significant room for improvement. Currently, there are two main directions of video feature extraction: deep features and hand-crafted features. Deep features are learned throughout a trainable deep neural network, providing high discriminative power on the upper layers of the network and having the ability to model high level of information (objects and actions) in a much better way. Deep feature-based techniques represent a breakthrough in research and obtaining remarkable results while hand-crafted features are manually designed and usually contain low-level information such as appearance and edge characteristics. Calculation of local features in these approaches can be usually decomposed into two phrases: detector, which aims to discover the salient and informative regions for action understanding, and descriptor, whose goal is to describe the visual patterns of extracted regions. Representations of hand-crafted local features are considered very effective to deal with large variations of motion speed, background clutter, video noise and illumination changes. However, these approaches may lack discriminative and semantic capacity for action recognition. Handcrafted feature-based techniques require expert designed feature detectors, descriptors, and vocabulary building methods for feature extraction and representation. This feature engineering process is labor-intensive and requires expertise of the subject matter.

With the current high availability of pre-trained off-the-shelf neural network and to overcome the limitations of hand-crafted techniques, current research is directed to deep learning-based approaches. These representations are characterized by their high level of sparsity and discriminative power and can be applied to several domains such as speech recognition, image classification and object recognition. Deep-learning-based methods automatically learn high level of semantic information from training data without applying any heuristic rules. However, deep-learning-based models lack consideration of temporal characteristics of video data and require huge amount of data for training, while most available datasets are relatively small. Meanwhile, most of the current deep-learning-based action recognition models largely ignore the intrinsic difference between spatial and temporal domain, and just treat temporal dimension as feature channels.

Motivated by above analysis and inspired by the method of Wang *et al.* [1] which combined the trajectories with deep convolution features for action recognition. In this work, we intelligently incorporate video temporal characteristics into deep architectures by using strategy of trajectory-constrained sampling and pooling, and propose a new descriptor called trajectory-based 3D convolutional descriptor (TCD). Firstly, we use the combination of motion boundary detector and super-pixel to detect salient motion regions then valid points are tracked within salient motion super-pixels. After that, trajectories are extracted by using KLT and SIFT tracker on the salient motion points within super-pixels. Secondly, we use the spatio-temporal stream for 3D ConvNet (C3D) [3] to model temporal information and capture multi-scale feature maps. Finally, we apply trajectory-based sampling and pooling over extracted feature maps to obtain our descriptors (TCDs). We encode the generated descriptors by vector of locally aggregated descriptors (VLAD) [33] encoding scheme. Our main contributions in this article are three-fold:

(1) We efficiently integrate the reduced but valid trajectories with deep learning features into a descriptor, while maintaining the low computational complexity.

(2) We experimentally show that combination of both features provides reliable representation with intrinsic characteristics and obtains superior and robust performance, as evidenced by comparison with the other state-of-the-art approaches.
(3) Our automatically learned descriptors (TCDs) are complimentary to available hand-crafted features (HOF, HOG, and MBH) and contain high discriminative capacity. Their fusion is able to further boost the recognition accuracy.

The remainder of this article is organized as follows: Section 2 reviews the related works. In section 3, we present the extraction of both hand-crafted and deep features pipeline. Section 4 explains the designing process of trajectory-based 3D convolutional descriptor (TCD). We demonstrate the experimental evaluation in section 5. Finally, conclusion is drawn in section 6.

## 2. RELATED WORKS

**Hand-Crafted Based Representations** Early research efforts mainly rely on hand-crafted local features and have become effective representations. Most of these approaches used detectors to define informative regions, which are robust to video noise and background clutter. In [5], Harris3D detector has been proposed to effectively extract the salient regions. Hessian detector [6] is used for blob detection in images. In [2], combination of the motion boundary detector with super-pixels is presented to detect informative motion regions. LTP [7], 3-D SIFT [8] and cuboid [9] have shown effectiveness and robustness against noise and partial occlusion. These aforementioned approaches commonly focus on extracting texture and edge characteristics defined by interest points. However, these approaches blend together different types of motions related to human action within the 3D space time block, thus resulting in a loss of discriminative power. Meanwhile many trajectory-based feature extraction methods have been introduced to facilitate motion information in effective ways. SIFT matching [10], KLT-based tracker [11] and dense trajectory features (DTF) [12] make it possible to separate different types of motion information from background information but these methods do not effectively blend the different types of motion related to a human action. Many hand-crafted local descriptors such as HOG-3D [13], Extended SURF [6], histogram of oriented gradient (HOG) [14], histogram of optical flow (HOF) [15], trajectory shapes (TS) [16] and motion boundary histogram (MBH) [17] have shown remarkable performance. These approaches extract the 3D volume around the interest points. However, unable to capture the local contents and classify the complex actions. Improved DT (iDT) [18], which is considered as state-of-the-art method makes use of sample interest points and optical flow to extract dense trajectories and represents each trajectory using different descriptors such as (HOG), (HOF), (MBH) and (TS). IDT method uses a human detector to suppress camera motion by estimating homography and is able to effectively represent the complex motion of human action. However, various issues such as presence of irrelevant and redundant trajectories and computational complexity still need to be addressed in a satisfactory way.

**Mid and High-Level Video Representations** To overcome these aforementioned limitations, several mid-high-level video representations scheme have been developed such

as Motionlets [19], DynamicPoselets [20], Actons [21], Actions Bank [22] and Motion Atoms and Phrases [23]. These approaches utilized some heuristic mining methods to extract discriminative visual points from feature units. But these methods are not fully able to address the complex issues related to human action recognition.

**Deep Learning-Based Representations** Due to the remarkable success of deep-CNNs, recent research is directed to deep-neural networks for action recognition. CNNs are widely used to learn spatial and temporal features and have been successfully applied to several domains such as speech recognition, object recognition, image classification, and human action recognition. Most of the early methods are based on convolution neural networks (CNNs) to learn deep video representations. Ji *et al.* [24] extended 2D ConvNet to video domain for action recognition on relatively small datasets, and recently Karpathy *et al.* [25] tested ConvNets with deep structures on a large dataset, called Sports-1M. Two-stream ConNet designed by Simonyan *et al.* [26] is probably the most successful architecture at present. It is composed of two neural networks, namely spatial nets and temporal nets. Spatial nets mainly capture the discriminative appearance features for action understanding, while temporal nets aim to learn the effective motion features. There have been a number of other prominent attempts such as Convolution RBMs [27], 3D ConNets [24], Deep ConNets [25] and learning spatio-temporal with 3D ConNets (C3D) [3] have achieved competitive recognition results. In these types of approaches, all features engineering processes are not very labor-intensive and the semantic representation is learned automatically from raw video data. We can characterize the deep features by their high discriminative capacity. Despite prominent contributions of these deep-learning-based approaches, there are some limitations as well. CNN-based networks only capture temporal dynamics and ignore the intrinsic difference between spatial and temporal domain. Another problem associated with these models is that they highly relied on large training datasets, while most of current available datasets are relatively small.

To address these issues, we propose a new method by incorporating the spatial and temporal characteristic of both hand-crafted and deep-learning features. The proposed descriptor can be applied to any small-size dataset for action recognition.

## 3. EXTRACTION OF HAND-CRAFTED AND DEEP FEATURES

In this section, we explain the methods for extracting hand-crafted and deep features. Section 3.1 details the extraction of hand-crafted features and relevant trajectories by tracking the interest points. In section 3.2, we present the process of extracting deep-learning features by using spatio-temporal stream of 3D ConNet [3]. Fig. 1 illustrates the complete overview of our framework.

### 3.1 Extraction of Hand-Crafted Features (Relevant Trajectories)

Despite the promising level of performance achieved by existing trajectory-based methods, a number of weaknesses still need to be addressed, such as computational complexity, presence of irrelevant changes and lack of robustness to camera motion. IDT [18] can be considered as the most effective trajectory-based representation and can efficient-

ly address aforementioned issues. However, IDT still suffers from some weaknesses, such as presence of substantial amount of redundant trajectories. In this section, we present an effective way to extract compact but valid and relevant trajectories at minimal computing cost, which can efficiently satisfy the current HAR requirements.

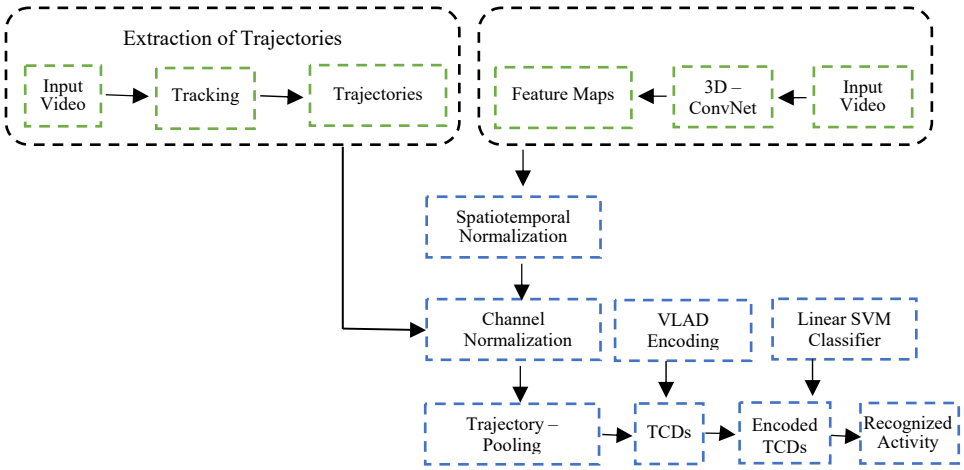



Fig. 1. Detailed description of the proposed framework.

Under the constrained environment, the changes of consecutive frames that are caused by motioning objects are only in local regions. Therefore, we select salient motion boundaries then define the super-pixels which contain salient information as change regions. Recently, motion boundary detector provides the most appealing way to suppress the camera constant motion and extraction of motion features by utilizing optical flow gradients. Motivated by the [2], we introduce a method in which salient motion features are detected by using motion boundary detector and then adaptive threshold method is used to define motion boundaries. The super-pixels that contain salient key points are defined as relative motion regions. Finally, SIFT matching [10] and KLT-based tracker [11] are used to generate the trajectories within super-pixels. The process of extracting motion super-pixel and trajectories by using our proposed method is shown in Fig. 2.

(a) Image frame (b) Optical flow (c) Motion boundaries (d) Motion super-pixels (e) Valid trajectories

Fig. 2. Process of salient motion super pixels and trajectories extraction.

The gradients of the variations of optical flow can be computed at point X and its surrounding grid points. It is very useful to decompose the gradient of optical flow into x and y directions, and a histogram can be obtained in both x and y directions. The computed optical flow is shown in Fig. 2 (b) and optical flow gradients in x and y direction can be represented by Eq. (1).

$$V_x = \frac{\partial v(x,y)}{\partial x},\ V_y = \frac{\partial u(x,y)}{\partial y}. \tag{1}$$

Where $V_x$ and $V_y$ are the horizontal and vertical flow gradient maps, each partial derivative captures the change of optical flow (*i.e.*, the motion boundary) with $v$ and $u$ as its correspondents. The changes of two consecutive frames $F_t$ and $F_{t+1}$ are always in local regions and we can compute the salient motion boundaries information by using Eq. (2), as shown in Fig. 2 (c).

$$A_{FT} = \tfrac{1}{2}(|V_x| + |V_y|) \tag{2}$$

Where $A_{FT}$ is the quantity that measures the motion boundary information at any patch $P_{t,j}$ of the frame. Eq. (2) measures the normalized sum of the change of the motion vectors. Now, we can define the super-pixels as change regions, which contain the key information, as illustrated in Fig. 2 (d). We first pre-define the changes number $q = HWa$ between the consecutive boundary images, which is proportional to the image resolution $WH$. Empirically, $a$ is set to $0.002 \le a \le 0.003$. Secondly, the pixel values distribution is computed by uniform quantization level $Bin$. After quantizing, the bin indexes which satisfy $q$ are ranked by the number within each interval. Having selected the last index $Ind_{max}$ from the ordered indexes, the adaptive threshold $\beta$ can be calculated by Eq. (3).

$$\beta = V_{min} + Ind_{max}\,(V_{max} - V_{min})/Bin, \tag{3}$$

where $V_{\max}$ and $V_{\min}$ are the maximum and minimum values in boundary image.

After this, we apply the KLT and SIFT tracker on salient motion super-pixels to track multiple points at original spatial scales. KLT and SIFT served as complementary sources to generate trajectories. The trajectories generated by our method are shown in Fig. 2 (e). In summary, given a video $V$, we obtain a set of trajectories as:

$$K(V) = \{K_1, K_2, K_3, \ldots, K_T\}, \tag{4}$$

where $T$ is the number of trajectories and $K_T$ denotes the $t$th trajectory in the original spatial scale:

$$K_T = \{(x_1^t, y_1^t, z_1^t), (x_2^t, y_2^t, z_2^t), \ldots, (x_p^t, y_p^t, z_p^t)\}. \tag{5}$$

Where $(x_p^t, y_p^t, z_p^t)$ is the pixel position of the $p$th point in trajectory KT. These trajectories will be used to extract our TCD by using trajectory-constrained sampling and pooling process.

### 3.2 Deep Features Extraction

This section represents the pipeline for deep features extraction. In principle, any kind of ConvNet architecture can be adopted for TCD extraction. In our implementation, we choose the 3D ConvNet (C3D) [3] for the spatio-temporal stream due to their remarkable performance on prominent public datasets. This network can be trained on large-scale video dataset and modeled the appearance and motion information simultaneously. The empirical and systematic study indicates that homogeneous setting with convolution kernels of 3×3×3 is the best option for C3D net. With these kernels settings C3D net can be used as deep as possible subject to the machine memory limit and computation affordability.

#### 3.2.1 C3D network architecture

C3D network has the ability to learn visual patterns directly from pixels without any pre-processing step. The architecture of C3D comprises of trainable filters and local pool operations, which is very useful to find hidden patterns in a video frame and captures all tiny changes in terms of spatial and temporal information.

**Table 1. The convolutional and pooling layers of the C3D architecture.**

| **Layers** | Conv1a | Conv2a | Conv3a | Conv3b | Conv4a | Conv4b | Conv5a | Conv5b |
|---|---|---|---|---|---|---|---|---|
| Size | 3×3×3 | 3×3×3 | 3×3×3 | 3×3×3 | 3×3×3 | 3×3×3 | 3×3×3 | 3×3×3 |
| Stride | 1×1×1 | 1×1×1 | 1×1×1 | 1×1×1 | 1×1×1 | 1×1×1 | 1×1×1 | 1×1×1 |
| Channel | 64 | 128 | 256 | 256 | 512 | 512 | 512 | 512 |
| Ratio | 1 | 1/2 | 1/4 | 1/4 | 1/8 | 1/8 | 1/16 | 1/1 |
| **Layers** | Pool1 | Pool2 | Pool3 | Pool4 | Pool5 | Fc6 | Fc7 | Softmax Layer |
| Size | 1×2×2 | 2×2×2 | 2×2×2 | 2×2×2 | 2×2×2 | – | – | |
| Stride | 1×2×2 | 2×2×2 | 2×2×2 | 2×2×2 | 2×2×2 | – | – | |
| Channel | 64 | 128 | 256 | 512 | 512 | 4096 | 4096 | |
| Ratio | 1/2 | 1/4 | 1/8 | 1/16 | 1/32 | – | – | |

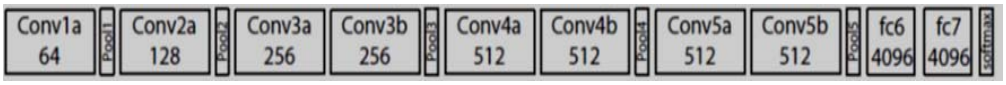



Fig. 3. Complete network architecture of C3D.

The architecture of C3D network is given in Fig. 3 and (Table 1) illustrates the different parameters setting of each convolutional and pooling layer. We refer 3D Convolution and pooling kernel size as $d \times k \times k$, where $d$ is kernel temporal depth and $k$ is kernel spatial size. The network has 5 convolution layers and 5 pooling layers (each convolution layer is immediately followed by a pooling layer), 2 fully connected layers and softmax loss layer. The number of channels (filters) for 5 convolution layers from 1 to 5 are 64, 128, 256, 512, and 512 respectively. Ratio represents the spatial map size ratio. In both spatial and temporal dimensions, all convolutional layers have 3×3×3 convolution filters with stride 1×1×1. All pooling layers from *pool*2 to *pool*5 (except for the first layer) have 2×2×2 pooling kernels with stride 2×2×2 which means the size of the output signal is reduced by a factor of 8 compared with input signal. The first pooling layer *i.e.* *pool*1 layer has kernel size 1×2×2 with the intention of not to merge temporal signal and

to preserve the temporal information in the early phases. The output of each convolutional layer is a kind of volume in the form of feature maps. All pooling layers lead to the same number of feature maps as convolution layers but with reduced spatial resolution and also these pooling layers introduce the scale-invariant features. The two fully connected layers have 2048 outputs and finally softmax layer is used to predict action labels.

### 3.2.2 Extraction process of feature maps

For the extraction of feature maps, we utilize C3D network [3] which can yield a feature hierarchy with increasing complexity. The special trainable filters of C3D network capture the input data and automatically extract the spatiotemporal features for classification. The degree of abstraction of features is enhanced as the number of layers increases. The result of a convolutional layer for each channel can be viewed as a spatiotemporal block *i.e.*, feature map. Through visualizing the feature maps, it can be found that the bottom two convolutional layers *i.e.* Conv1a and Conv2a learn the underlying features such as edges and colors. The next two layers *i.e.* Conv3a and Conv3b extract high texture features. Conv4a and Conv4b learn more discriminative features such as face of the object. The last two layers *i.e.* Conv5a and Conv5b have larger receptive fields and obtain the most invariant and discriminative features (complex visual elements). This is the reason, we only consider the output feature maps of layer Conv5b for our model because these feature maps have high level of visualization and projection.

Similar to [3], we use a sampling step size of one frame to iterate over the frames of the video for creating the input clips. As the last layer of this network *i.e.*, pool5 has the size of feature maps of only 4×4. In our pipeline, we make a modification in the network. We only consider the output feature maps of layer conv5b and will remove the layers from pool5 to fc7. The layer conv5b contains two feature maps (Spatial and temporal feature maps), each of them 7×7×512. These feature maps will be used for extracting TCD in the next subsection. We can represent these two feature maps as:

$C_s$ denotes the feature map of spatial scale and $C_t$ is the feature map of temporal scale of $M$ layers.

$$C_s = \{c_1^s, c_2^s, c_3^s, ..., c_m^s\} \text{ and}$$
$$C_t = \{c_1^t, c_2^t, c_3^t, ..., c_m^t\}$$

So we can represent the set of all convolutional maps of video stream $V$ as:

$$C_v = \{C_s, C_t\}. \tag{6}$$

We denote the size of the feature map by $C_v \in \mathbb{R}^{H \times W \times D \times N}$, where $H$ and $W$ are the height and width in spatial dimension, $D$ is the depth in temporal dimension, and $C$ is the number of channels.

## 4. TRAJECTORY BASED 3D CONVOLUTIONAL DESCRIPTOR (TCD)

TCD is a kind of local trajectory-aligned descriptor computed in a 3D volume around the trajectory. The size of the volume is $N \times N$ pixels, where $N$ is the receptive

field size and $P$ is the trajectory length. This section describes the method for extracting TCDs from convolutional feature maps $C_v$ and a set of valid trajectories $K(V)$ for a given video $V$. Our extracted feature maps capture the appearance and motion information of 3D volume by using C3D net. However, to enhance the robustness of TCDs and to transform the feature maps in more scalable and generalized form, we design two normalization method, namely spatiotemporal and channel normalization. These normalization methods provide the following useful aspects to boost the recognition accuracy.

(1) To ensure the high visualization of the feature maps and their projection to the image space.
(2) To reduce the influence of the illumination changes and to keep the discriminative information intact in the feature maps.
(3) To suppress the activation burstiness caused by some neurons in the C3D net.
(4) To make sure that the feature value of each pixel range in the same interval, and let each pixel make the equal contribution in the final representation.

*Spatiotemporal normalization* is conducted by dividing the feature map values by the maximum value of the spatiotemporal block for each channel. Given a feature map $C_v \in \mathbb{R}^{H\times W\times N}$, we normalize the feature value as follows:

$$\tilde{C}_{st}(h, w, n) = C(h, w, n)/\max_{h,w} C(h, w, n), \tag{7}$$

where $\max_{h,w} C(h, w, n)$ is the maximum value of $n$th feature map over the whole video spatiotemporal extent. This normalization method ensures that each convolutional feature channel ranges in the same interval, and thus contributes equally to final TCD recognition performance.

*Channel normalization* is conducted by dividing the feature map values by the maximum value in the spatiotemporal position across different channels. Given a feature map $C_v \in \mathbb{R}^{H\times W\times N}$, we can conduct channel normalization as follows:

$$\tilde{C}_{ch}(h, w, n) = C(h, w, n)/\max_{n} C(h, w, n), \tag{8}$$

where $\max_n C(h, w, n)$ is the maximum value of different feature channels. This normalization method ensures the equal contribution of feature maps in the final representation.

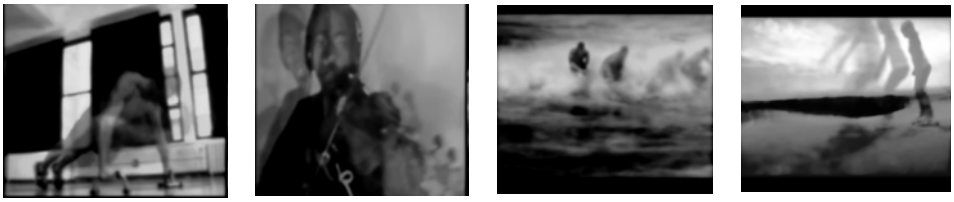
After the step of feature normalization, the values of the points on feature maps are aligned into a same interval. In experiment, these two normalization methods are used separately and then results $\tilde{C}_{st}(h, w, n)$ and $\tilde{C}_{ch}(h, w, n)$ are fused to further enhance the performance. Fig. 4 shows some real examples of generated feature maps after normalization methods. We can observe that the generated feature maps have the high visualization with minimum illumination effects.

Now, we will extract TCD based on valid trajectories $K_T$ and normalized convolutional feature maps $\tilde{C}$ by performing trajectory pooling. In C3D net, spatial and temporal padding are implemented on the convolutional layer to make its inputs and outputs have the same size. The effect of the padding is that it creates the mapping between the points in videos and those on feature maps. So, the points with coordinate $(h, w, d)$ in video clip

$V$ corresponds to that with coordinate ($r\times h$, $r\times w$) on obtained representation of feature maps. Where $r$ is the spatial map size ratio calculated in advanced, as listed in (Table 1). In this way, points of the trajectories $K_T$ are mapped directly on obtained normalized feature map $\tilde{C}$ when conducting trajectory pooling by using following Eq. (9).

$$D(K_T, \tilde{C}) = \max_p \tilde{C}(\overline{(r \times h_p^k)}, \overline{(r \times w_p^k)}, n), \tag{9}$$

where, $r$ is the spatial feature map size ratio with respective to input size and $\overline{(r \times h_p^k)}$, $\overline{(r \times w_p^k)}$ is mapped from the corresponding $p$th point $(h_p^k, w_p^k, d_p^k)$ of original video in trajectory $K_T$, $(\bar{.})$ is the rounding operation. So, $D(K_T, \tilde{C}) \in \mathbb{R}^{N\times T}$ is our designed trajectory based 3D convolutional descriptor (TCD), where $T$ is the number of trajectories and $N$ is the number of channels.



Pushups Playing violen Surfing Skat boarding

Fig. 4. Motion map after normalization methods, illustrating the discriminative information with high visualization and minimum effect of illumination changes.

## 5. EXPERIMENTATION AND RESULTS

In this section, the proposed approach is experimentally evaluated on well-known benchmark human action datasets: KTH [28], UCF101 [29], HDMB51 [30] and UCF Sports [31]. The description of these datasets, experimental setup and comparative analysis are presented in subsequent sections.

### 5.1 Datasets and Their Validation Schemes

The **KTH dataset** [28] is a well-known public dataset containing 599 video clips of 6 human action classes, including running, jogging, walking, boxing, clapping and waving. The sequences were recorded with a fixed camera at the rate of 25 fps in different setups such as illuminations, appearance and scale variations. Most of the sequences are single view with static and homogeneous background. We follow the original experimental setup *i.e* distributing the samples into training set of (2, 3, 5, 6, 7, 8, 9, 10 and 22) (9 subjects) and the remaining 16 subjects into training set. Average accuracy is reported as the performance measure over all classes.

The **UCF101** dataset [29] is a widely adopted benchmark for human action recognition and the extension of UCF50. It comprises of 101 action classes and at least 100 video clips associated with each action class. There are 13,320 video clips in total. Most of the video clips are realistic, clean and user-uploaded videos with cluttered background and camera motion. We adopt validation scheme of the THUMOS13 challenge [32] and

follow the three testing/training splits for performance evaluation by reporting average recognition accuracy.

The **HDMB51** dataset [30] is a large collection of realistic videos ranging from YouTube and Google videos to digitized videos collected from various sources. In total, there are 6,766 manually annotated video sequences of 51 different action categories and each category contains at least 100 video clips. This dataset is very challenging and complex as it contains videos with more interclass variations and complicated background. For experimental setting, we follow the original evaluation guidelines [41] using three different testing/training splits. Each split with each action class has 30 clips for testing and 70 clips for training. We report the average recognition accuracy over these three splits.

The **UCF sports** dataset [31] is also a very popular dataset and characterized by higher action complexity. This dataset encompasses 150 videos from 10 action classes, including diving, golf swing, kicking, lifting, horse riding, running, skate-boarding, swing-bench and walking. These videos were recorded in real sports environment taken from broadcast television channel such as ESPN and BBC. We adopt the evaluation scheme proposed in [43, 44] that splits datasets into 47 testing samples and 103 training samples. This kind of distribution minimizes the strong correlation of background cues between training and testing sets. The average accuracy is used to measure the final performance.

### 5.2 Experimental Setup and Parameters Tuning

This section explains the implementation details for the validation scheme of benchmark datasets and training of (C3D) network. The validation scheme for KTH and UCF sports are given in the section 5.1. As UCF101 is larger than HMDB51 dataset so we use it to train (C3D) network initially, and transfer this learned model for TCD extraction on the HMDB51 dataset. We choose the training dataset of UCF101 split1 for learning (C3D) network to extract the local deep features. Caffe toolbox is used to implement our model while OpenCV is used for trajectory extraction. We fine tune the model parameters on the UCF101 dataset, where the learning rate is set as $10^{-2}$, decreased to $10^{-3}$ after 14K iterations, and training stopped at 20K iterations.

We crop each testing frame from center and four corners with size of 224×224 and the mirror of these crops, so there are 10 crops for each testing frame. In the temporal multi-scale strategy, we sample 32 frames with equal interval. We use VLAD (Vector of Locally Aggregated Descriptors) [33] as encoding method to capture high statistical information. Unlike iFV (improved fisher vector) [34], VLAD is simpler and based on first order statistics. In many recent works, VLAD encoding method outperforms the iFV, when using deep features. We create the codebooks from 500K random selected features extracted from a subset of videos. We set the size of the codebook to 256 visual words, which is the standard adopted size. For the classification part, we use linear SVM with the parameter C = 100 in all experiments.

### 5.3 Experiments and Comparative Analysis

In this section, we test our proposed method in the context of action recognition. We give experimental results and comparative analysis with state of the art approaches.

**Experiment 1:** Evaluation of effectiveness and efficiency of our extracted trajectories

We compare our relative motion point trajectory method with three other state of the art trajectory-based methods: SIFT [10], KLT [11] and dense cuboid [9] on UCF Sports dataset. We list the performance of each individual descriptor (HOF, HOG and MBH) and their overall possible combinations in terms of average accuracy for all trajectory methods in (Table 2). For SIFT, KLT and dense cuboid approaches, we adopt Leave-One-Out experimental setup *i.e.*, they are using more than 90% of the data for training and the rest for testing. For our approach, we split dataset into 47 testing samples and 103 training samples. It can be observed that results obtained by our trajectory method are much superior to existing state-of-the-art existing trajectory-based methods. In general, the combination of all the descriptors improves the performance significantly on our proposed trajectory-based method. Results are 5.4% better than the best counterpart.

Furthermore, we also investigate the efficiency of the proposed approach, we analyzed the trajectory rejection rate and the speed of operation *i.e.*, the number of frames per seconds (fps). We measured the operation time from loading of video to obtaining all the descriptors. We used 101 videos from the action class pushing of HMDB51, having an average frame size of 335×240 pixels. All results are obtained on single Geforce GTX Titan 3.6GHZ with 6 GB RAM and GTX 980 graphics card, not using any parallel processing. We compare our introduced approach with other existing state-of-the-art methods such as improved dense trajectories (IDT) [18], ordered trajectories (OT) [38] and trajectories rejection (TR) [39] methods.

According to (Table 3), the proposed method only selects 99.90 valid and irredundant trajectories per frame. On the other hand, (IDT) and (OT) select 128.72 and 110.0 trajectories per frame respectively and (TR) produces minimum number of trajectories *i.e.*, 90.41 per frame. Additionally, we can observe that the processing speed of (IDT) is 7.82±0.003 fps and (OD) obtains 8.05±0.003 fps. In contrast, the processing speed of proposed approach is about 8.4±0.004 fps which is nearly 17% of increase in speed as compared to (IDT) and 11% as compared to (OD). The primary reason is that our approach require minimum computing time for optical flow computation. Furthermore, since the proposed approach does not make use of homography estimation and only extracts descriptors from the selected trajectories, the proposed approach is able to attain a lower computational complexity.

**Table 2. Performance comparisons of our method (extracted trajectories) with state of the art trajectories methods. We report average accuracy over all classes for the UCF sports dataset.**

| Descriptors | Trajectories (Tested on UCF sports Dataset) | | | |
|---|---|---|---|---|
| | SIFT | KLT | Dense-cuboids | Our (Trajectories) |
| TS | 55.7 | 72.8 | – | 71.8 |
| HOF | 74.2 | 80.2 | 80.2 | 75.7 |
| HOG | 69.9 | 72.7 | 77.8 | 87.3 |
| HOF+HOG | 70.8 | 75.4 | 79.1 | 83.1 |
| MBH | 72.1 | 78.4 | 83.2 | 86.7 |
| HOF+MBH | 76.9 | 80.3 | 83.9 | 87.0 |
| Combined | 77.9 | 82.1 | 85.5 | **90.9** |

Overall, our method achieves the second best results after (TR), which obtains processing speed of 9.84±0.004 by extracting only 90.41 trajectories per frame. (TR) follow dynamic frame skipping scheme by computing absolute difference between two consecutive frames in order to measure the significance change. However, our method outperform the (TR) in terms of recognition accuracy as shown in (Table 7) on two well-known datasets. The possible reason is that, we do not follow any frame skipping scheme as in (TR) in order to prevent the chance of losing any valid information from skipped frames. In summary, our method proved very effective to minimize the computation complexity of different subsequent operations without any significant loss of accuracy.

**Table 3. Efficiency of human action recognition (%) for our method for extracted trajectories and comparison with other trajectory based methods.**

| Trajectory Method | Total frames | # of extracted Trajectories | Processing time (fps) |
|---|---|---|---|
| IDT [18] | 29,258 | 128.72/frame | 7.82±0.003 |
| OT [38] | 29,258 | 110.0/frame | 8.05±0.003 |
| TR [39] | 29,258 | 90.41/frame | **9.84±0.004** |
| Our method | 29,258 | 99.90/frame | 8.48±0.004 |

**Experiment 2:** Exploration Experiments

In this section, TCD is used to explore the influence of different settings in steps of the proposed pipeline. To specify the PCA dimension of TCD, we first explore different dimensions reduced by PCA on the HMDB51 dataset. We employ our proposed TCD with conv5b descriptor and spatiotemporal normalization to investigate the impact of different PCA dimensions and the results are summarized in (Table 4). We vary the dimension from 32 to 128 and results show that dimension 64 gets the best performance among them, and higher dimension may cause performance degradation. Thus, TCD is reduced to 64 dimension and then fed into our encoding scheme *i.e.* (VLAD) in the whole experiments.

We also conduct experiments to analyze the effectiveness of normalization methods by using conv5b descriptor from 3D convolutional spatio-temporal net [3] on the HMDB51 dataset. In (Table 4), we describe the average accuracy of different normalization methods. *ST.Norm* and *Cha.Norm* stands for spatiotemporal normalization and channel normalization respectively and *No.Norm* stands for the original representation without normalization. Combination of them is 5% better than *No.Norm,* which demonstrates the good effects of normalization methods. Therefore, in the remainder of this section, we will use the combined representation obtained from these two normalization methods for our TCD.

**Table 4. Exploration of different setting for PCA (dimension) and normalization method for designing TCDs.**

| Different settings | No. Norm | Cha. Norm. | ST. Norm. | Combined. Norm. | PCA 32 dim | PCA 64 dim | PCA 128 dim |
|---|---|---|---|---|---|---|---|
| Accuracy | 0.49 | 0.51 | 0.52 | 0.54 | 0.48 | 0.52 | 0.50 |

**Experiment 3:** Class-wise accuracy for action recognition

In this section, we compute the class-wise accuracy for action recognition to evaluate the performance of our method. Fig. 5 shows the category-wise accuracy of HMDB51 dataset on test data. The horizontal-axis represents categories and the vertical-axis shows the percentage accuracy of corresponding category. The HMDB51 dataset contains a variety of actions related to human body movement, facial actions and human interaction for body movements. There are total 51different classes, each containing more than one hundred clips. For our experiment, we consider all 51 action classes. From results, it can be seen that recognition results of most of the categories are greater than 90%; some of them reach 100%; and only 4 categories have accuracies less than 50%. The variation in accuracy of most of the classes is between 80% to 100%. The proposed method improved the recognition rate on HMDB51 dataset from 68.7% to 77%.

We further investigate the recognition accuracy of our method by constructing confusion matrix of two datasets *i.e.* KTH and UCF Sports datasets. The confusion matrixes for TCD on KTH and UCF Sports datasets are shown in (Table 5) and (Table 6) respectively. The confusion matrix indicating the accuracy of each action and correspondence between the target classes along x-axis and output classes along y-axis. We consider 6 action categories from KTH dataset and 10 action categories from UCF Sports dataset to conduct our experiment. On the KTH dataset, our model performs well on *Boxing*, *HandClapping* and *Walking* categories. According to the confusion matrix, intensity of the true score is high (diagonal) for each category and our method achieves 94% average accuracy for all six classes. It is interesting to note that in KTH dataset, some of categories with similar actions are easily confused with each other and giving the minimal false prediction. For example *Handwaving* and *Handclapping*, both are interfering with each other and misclassified. Similarly, some leg-related actions such as *Jogging* and *Running* are interfering with each other and giving low recognition scores. The possible explanation is that *Jogging* has similar trajectories with *Running.* The confusion matrix of UCF Sports is also well diagonalized and our method obtains 92% average recognition accuracy for all 10 action categories. However, some action categories such as *Golfswing*, *Kicking* and *Running* are misclassifying each other with some minimal false prediction and giving low recognition scores. The underlying reasons are the similarity of the features and representations among different actions, and the number of training samples are too small, despite TCD still performs well on most categories.


Fig. 5. Class wise accuracy of HMDB51 dataset for action recognition.

**Table 5. Confusion matrix of KTH dataset with action categories.**

| Categories | Boxing | Hand Clapping | Hand Waving | Jogging | Running | Walking |
|---|---|---|---|---|---|---|
| Boxing | 1.000 | 0 | 0 | 0 | 0 | 0 |
| Hand Clapping | 0 | 0.95 | 0 | 0 | 0 | 0.01 |
| Hand Waving | 0 | 0.03 | 0.94 | 0 | 0 | 0.01 |
| Jogging | 0 | 0 | 0 | 0.91 | 0.02 | 0.01 |
| Running | 0 | 0 | 0 | 0.11 | 0.90 | 0.01 |
| Walking | 0 | 0 | | 0.01 | 0 | 0.96 |
| **Average Accuracy** | | | | | | **0.94** |

**Table 6. Confusion matrix of UCF sports dataset with action categories.**

| Categories | Diving | Golf-Swing | Kicking | Lifting | Riding Horse | Running | Skate boarding | Swing Bench | Swing-Side | Walking |
|---|---|---|---|---|---|---|---|---|---|---|
| Diving | 1.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Golf-Swing | 0 | 0.90 | 0.07 | 0 | 0 | 0 | 0 | 0.10 | 0 | 0 |
| Kicking | 0 | 0.17 | 0.91 | 0 | 0 | 0 | 0 | 0.01 | 0 | 0 |
| Lifting | 0 | 0 | 0 | 0.95 | 0 | 0 | 0 | 0 | 0 | 0 |
| Riding Horse | 0 | 0 | 0 | 0 | 0.93 | 0 | 0 | 0 | 0 | 0.06 |
| Running | 0 | 0.02 | 0.17 | 0 | 0.09 | 0.89 | 0 | 0 | 0 | 0.09 |
| Skateboarding | 0 | 0 | 0 | 0 | 0 | 0 | 0.93 | 0 | 0 | 0 |
| Swing Bench | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.94 | 0 | 0 |
| Swing Side | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.93 | 0 |
| Walking | 0 | 0.10 | 0 | 0.09 | 0 | 0 | 0 | 0.07 | 0.05 | 0.90 |
| **Average Accuracy** | | | | | | | | | | **0.92** |

**Experiment 4:** Comparison to the state of the art methods

To evaluate the effectiveness and performance of proposed TCD, we compare it to a variety of existing state-of-the-art methods on UCF101 and HMDB51 datasets. Since videos in both datasets are relatively long, we adopt a spatial-temporal multi-scale testing strategy for these two datasets. The comparison results are presented in (Table 7). We divide these baseline methods into different groups according to the type of feature being used, such as hand-crafted features, deep-learned features, and hybrid features. Among the hand-crafted feature-based techniques, iDT-FV [37] performs well and have competitive results but our approach outperforms the iDT-FV by fair margin on both datasets. Compared with deep learning models such as Hierarchal Rank pooling [44], Chained Multi-stream [45], the proposed method achieves better results than [44, 45] on UCF101 and HDMB51 respectively. As our approach is based on hybrid model, we also compare our method with existing state-of-the-art hybrid model based techniques such as TDD [1] and MTC3D [47]. Both of these models follow iDT [18] for trajectories extraction and adopt higher-order encoding scheme *i.e.* Fisher Vector (FV) to encode the handcrafted features. According to the result, our proposed method outperforms these two methods by fair margin on both datasets. The underlying reason is that our introduced method for

trajectories extraction is very much capable of reducing irrelevant background and camera motion trajectories without skipping of any frame of video. We analyze the performance of our TCDs with both encoding scheme *i.e.* improved Fisher Vector (iFV) and VLAD and achieve best results with VLAD encoding scheme. On the whole, the combination of (C3D) with trajectory pooling strategy provides the best results, which obtains the accuracy of 70.0 % on HMDB51 and 92.3% on UCF101 dataset and witness the effectiveness of trajectory-based 3D convolutional descriptor (TCD) for the action recognition tasks.

**Table 7. Comparison of our (TCDs) method to the state-of-the-arts approaches.**

| Features | State of the art approach | Year | UCF101 | HMDB51 |
|---|---|---|---|---|
| Hand-Crafted | iDT [18] | 2013 | 84.70 | 57.21 |
| | iVLAD [35] | 2014 | 84.16 | 56.36 |
| | M-PCCA-MVSV [36] | 2014 | 83.50 | 55.90 |
| | iDT-FV [37] | 2014 | 87.90 | 61.10 |
| | Ordered Trajectories [38] | 2015 | 72.80 | 47.30 |
| | Trajectory Rejection [39] | 2017 | 85.74 | 58.91 |
| Deep-CNNs | Two-stream ConvNets [26] | 2014 | 88.0 | 59.40 |
| | Factorized ConvNets [40] | 2015 | 88.10 | 59.10 |
| | Temporal Pyramid CNNs [41] | 2015 | 89.10 | 63.10 |
| | Dynamic Images [42] | 2016 | 89.10 | 65.20 |
| | Two Stream 3D-Nets [43] | 2016 | 90.20 | – |
| | Hierarchal Rank pooling [44] | 2016 | 91.41 | 66.90 |
| | Chained Multi-stream [45] | 2017 | 91.10 | 69.90 |
| | Hidden Two-stream CNNs [46] | 2017 | 90.30 | 58.90 |
| Hybrid | TDD+iDT-FV [1] | 2015 | 91.50 | 65.90 |
| | MTC3D+iDT+FV [47] | 2017 | 90.4 | 65.0 |
| | Our TCD-iFV | – | 91.40 | 66.70 |
| | Our TCD-VLAD | – | **92.3** | **70.0** |

## 6. CONCLUSION

In this paper, we introduce an effective action recognition framework in the form of trajectory based 3D convolutional descriptor, which efficiently integrates the merits of both hand-crafted and deep features. First, relative point motion trajectories are extracted from relative motion regions within the super-pixels. 3D convolutional architecture for spatio-temporal stream is utilized to learn discriminative feature maps. Then we aggregate these convolutional features into TCDs by adopting trajectory constrained sampling and pooling techniques. In addition, we introduce some normalization methods which further boost the recognition accuracy. The achieved results over the benchmark datasets demonstrate the robustness and superiority of our video representation.

## REFERENCES

1. L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proceeding of IEEE International Conference on Com-*

*puter Vision and Pattern Recognition*, 2015, pp. 4305-4314.
2. X. Cheng, N. Li, and T. Zhou, "Online tracking via super-pixel and sparse representation," *Journal of Electronics and Information Technology*, Vol. 36, 2014, pp. 2393-2399.
3. D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d Convolutional networks," in *Proceedings of IEEE International Conference on Computer Vision*, 2015, pp. 4489-4497.
4. Z. Xu, Y. Yang, and A. G. Hauptmann, "A discriminative CNN video representation for event detection," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1799-1807.
5. I. Leptev, "on space-time interest points," *International Journal of Computer Vision*, Vol. 64, 2005, pp. 107-123.
6. G. Willems, T. Tuytelaars, and L. J. V. Gool, "An efficient dense and scale – variant spatio-temporal interest point detector," in *Proceedings of European Conference on Computer Vision*, 2008, pp. 650-663.
7. L. Yeffet and L. Wolf, "Local trinary patterns for human action recognition," in *Proceedings of IEEE International Conference on Computer Vision*, 2009, pp. 492-497.
8. P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional SIFT descriptor and its application to action recognition," in *Proceedings of International Conference on Multimedia*, 2007, pp. 357-360.
9. P. Dollr, V. Rabaud, G. Cottrell, and S. Belongie, "Behavior recognition via sparse spatio-temporal features," in *Proceedings of IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*, 2005, pp. 65-72.
10. J. Sun, X. Wu, S. Yan, L. F. Cheong, T. S. Chua, and J. Li, "Hierarchical spatio-temporal context modeling for action recognition," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2009, pp. 2004-2011.
11. P. Matikanen, M. Hebert, and R. Sukthankar, "Trajectons: Action recognition through the motion analysis of tracked features," in *Proceedings of IEEE International Conference on Computer Vision Workshop*, 2009, pp. 514-521.
12. H. Wang, A. Klser, C. Schmid, and C. L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, Vol. 103, 2013, pp. 60-79.
13. A. Klaser, M. Marszlek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proceedings of the 19th British Machine Vision Conference*, 2008, pp. 1-10.
14. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2005, pp. 886-893.
15. N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proceedings of European Conference on Computer Vision*, 2006. pp. 428-441.
16. N. Sundaram, N. Brox, and K. Keutzer, "Dense point trajectories by GPU-accelerated large displacement optical flow," in *Proceedings of European Conference on Computer Vision*, 2010, pp. 438-451.

17. H. Wang, A. Klaser, and C. Schmid, "Dense trajectories and motion boundary descriptors for action recognition," *International Journal of Computer Vision*, Vol. 103, 2013, pp. 60-79.
18. H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proceedings of IEEE International Conference Computer Vision*, 2013, pp. 3551-3558.
19. L. Wang, Y. Qiao, and X. Tang, "Motionlets: Mid-level 3D parts for human motion recognition," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2674-2681.
20. L. Wang and Y. Qiao, "Video action detection with relational dynamic-poselets," in *Proceedings of European Conference on Computer Vision*, 2014, pp. 565-580.
21. J. Zhu, B. Wang, and X. Yang, "Action recognition with actions," in *Proceedings of IEEE International Conference on Computer Vision*, 2013, pp. 3559-3566.
22. S. Sadanand and J. J. Corso, "Action bank: A high-level representation of activity in video," in *Proceeding of IEEE International Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1234-1241.
23. L. Wang, Y. Qiao, and X. Tang, "Mining motion atoms and phrases for complex action recognition," in *Proceedings of IEEE International Conference on Computer Vision*, 2013, pp. 2680-2687.
24. S. Ji and W. Xu, "3D convolutional neural networks for human action recognition," *Transactions on Pattern Analysis and Machine Intelligence*, 2013, pp. 221-231.
25. A. Karpathy, G. Toderici, S. Shetty, T. Leubg, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725-1732.
26. K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proceedings of International Conference on Neural Information Processing Systems*, 2014, pp. 568-576.
27. G. W. Taylor and R. Fergus, "Convolutional learning of spatio-temporal features," in *Proceedings of European Conference on Computer Vis*ion, 2010, pp. 140-153.
28. C. Schuldt, I. Laptev, and B. Caputo, "Recognizing human actions: a local SVM approach," in *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2004, pp. 32-36.
29. K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," *Computing Research Repository*, 2012, pp. 1-7.
30. H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: a large video database for human motion recognition," in *Proceedings of IEEE International Conference on Computer Vision*, 2011, pp. 2556-2563.
31. M. Rodriguez, J. Ahmed, and M. Shah, "Spatio-temporal maximum average correlation height templates in action recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2008, pp. 1-8.
32. Y. G. Jiang, J. Liiu, A. R. Zamir, I. Laptev, M. Piccardi, and M. Shah, "THUMOS challenge: Action recognition with a large number of classes," *Computer Vision and Pattern Recognition*, Cornell Unversity, arXiv:1604.06182, 2013.
33. H. Jégou, and F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid, "Aggregating local image descriptors into compact codes," *Transactions on Pattern Analysis and Machine Intelligence*, Vol. 34, 2012, pp. 1704-1716.

34. F. Perronnin, J. Sanchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proceedings of European Conference on Computer Vision*, 2010, pp. 143-156.
35. J. Wu, Y. Zhang, and W. Lin, "Towards good practices for action video encoding," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 2571-2578.
36. Z. Cai, L. Wang, X. Peng, and Y. Qiao, "Multi-view super vector for action recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 596-603.
37. X. Peng, L. Wang, X. Wang, and Y. Qiao, "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice," *Computer Vision and Image Understanding*, Vol. 150, 2016, pp. 109-125.
38. O. R. Murthy and R. Goecke, "Ordered trajectories for human action recognition with large number of classes," *Image and Vision Computing*, Vol. 42, 2015, pp. 22-34.
39. J. J. Seo, H. I. Kim, W. De. Neve, and Y. M. Ro, "Effective and efficient human action recognition using dynamic frame skipping and trajectory rejection," *Image and Vision Computing*, Vol. 58, 2016, pp. 76-85.
40. L. Sun, K. Jia, D. Y. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temmporal convolutional networks," in *Proceedings of European Conference on Computer Vision*, 2015, pp. 1-9.
41. P. Wang, Y. Cao, C. Shen, L. Liu, and H. T. Shen, "Temporal pyramid pooling based convolutional neural networks for action recognition," *Transactions on Circuits and Systems for Video Technology*, Vol. 27, 2016, pp. 2613-2622.
42. H. Bilen, B. Fernando, E. Gavved, A. Vedaldi, and S. Gould, "Dynamic image networks for action recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3034-3042.
43. A. Diba, A. M. Pazandeh, and L. V. Gool, "Efficient two-stream motion and appearance 3D CNN for video classification," in *Proceedings of European Conference on Computer Vision*, 2016, pp. 1-4.
44. B. Fernando, P. Anderson., M. Hutter, and S. Gould, "Discriminative hierarchical rank pooling for activity recognition," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 1924-1932.
45. M. Zolfaghari, G. L. Oliveira, and T. Brox, "Chained multi-stream networks exploiting pose, motion, and appearance for action classification and detection," in *Proceedings of IEEE International Conference on Computer Vision*, 2017, pp. 2904-2913.
46. Y. Zhu, Z. Lan, S. Newsam, and A. G. Hauptmann, "Hidden two-stream convolutional networks for action recognition," in *ArXiv*, 2017.
47. X. Lu, H. Yao, and S. Zhao, "Action recognition with multi-scale trajectory-pooled 3D convolutional descriptors," *Multimedia Tools and Applications*, Vol. 78, 2017, pp. 1-17.

**Sheeraz Arif** received the M.S degree in Telecommunications Engineering and Computer Networks from London South Bank University, UK, in 2006. He is currently pursuing the Ph.D. degree from the School of Information and Electronics Engineering, Beijing Institute of Technology China. His research interests include machine learning, human action recognition, computer vision and video analysis.

**Jing Wang (王晶)** received the Ph.D. degree in Electronic Engineering in 2007 from Beijing Institute of Technology (BIT), China. She is now an Associate Professor in School of Information and Electronics Engineering, (BIT), China and currently associated with the Research Institute of Communication Technology (RICT) in Beijing Institute of Technology. Her research interests include speech and audio signal processing, multimedia quality assessment and mobile communication.

**Fida Hussain** received the B.E degree from D.U.E.T. Pakistan, in 2009 and the M.E. degree from Hamdard University Pakistan, in 2011 and Ph.D. degree from School Electrical and Information Engineering, Jiangsu University, China in 2018. His research interests include smart grids, power system automation machine learning.

**Zesong Fei (费泽松)** received the Ph.D. degree in Electronic Engineering in 2004 from Beijing Institute of Technology (BIT), China. He is now a Professor in School of Information and Electronics Engineering, Beijing Institute of Technology China. He is currently associated with the Research Institute of Communication Technology (RICT) in Beijing Institute of Technology. His research interests include wireless communication and multimedia signal processing.