Multimed Tools Appl (2019) 78:507–523
https://doi.org/10.1007/s11042-017-5251-3

Action recognition with multi-scale trajectory-pooled 3D convolutional descriptors

Xiusheng Lu[1] · Hongxun Yao[1] · Sicheng Zhao[1] · Xiaoshuai Sun[1] · Shengping Zhang[1]

Received: 1 June 2017 / Revised: 7 September 2017 / Accepted: 20 September 2017 /
Published online: 5 October 2017

Abstract Hand-crafted and learning-based features are two main types of video representations in the field of video understanding. How to integrate their merits to design good descriptors has been the research hotspot recently. Motivated by TDD (Wang et al. 2015), we combine trajectory pooling method and 3D ConvNets (Tran et al. 2015) and put forward a novel multi-scale trajectory-pooled 3D convolutional descriptor (MTC3D) for action recognition in this paper. Specifically, we calculate multi-scale dense trajectories from the input video and perform trajectory pooling on feature maps of 3D CNN. The proposed descriptor has two advantages: 3D CNN has the ability to extract high-level semantic information from videos and multi-scale trajectory pooling method utilizes the temporal information of videos subtly. The experiments on the datasets of HMDB51 and UCF101 demonstrate that the proposed descriptor achieves state-of-the-art results.

Keywords Trajectory pooling · 3D ConvNets · Action recognition

1 Introduction

With the explosive growth in the amount of videos on the Internet, action recognition [1, 15, 27] has attracted increasing attention in recent years, which has potential applications in different fields such as abnormal event detection [3], human-computer interaction [28], video retrieval [33, 45] and robot perception [6]. Numerous researchers dedicate to this area to deal with the challenges like occlusions, low resolution, background clutter and camera motions.

✉ Hongxun Yao
h.yao@hit.edu.cn

Xiusheng Lu
xiusheng.lu.cs@gmail.com

[1] School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China

Feature extraction is the fundamental and critical step in the framework of image and video analysis [21, 23, 24, 26, 32, 42, 46, 47]. Video representations are usually motivated by image features. Compared to images, videos have additional temporal information. How to apply the motion information contained in the time domain of videos is the core issue when designing/learning video representations. There are two main types of features for video description: hand-crafted descriptors and learning-based descriptors, which are presented comprehensively in Section 2.

Hand-crafted and learning-based descriptors have their own advantages which are complementary to each other: the design of hand-crafted descriptors reflects the researcher's observation of visual data and they are easy to explain, while learning-based descriptors usually have higher discriminative capacity and are hard to interpret. How to combine the benefits of these two kinds of features to design good descriptors has been an active research area. On one hand, the experience in designing hand-crafted features can be utilized to guide the devise of deep neural networks. For instance, 3D ConvNets [16, 38] can be considered borrowing the idea from HOG3D [18] or 3D SIFT [30]. One the other hand, some techniques of hand-crafted descriptors are used to post-process deep descriptors. In [41], trajectory-pooled deep-convolutional descriptor (TDD) united dense trajectories of iDT descriptors with two-stream ConvNets [32] and achieved good performance. Motivated by TDD, in this paper we focus on integrating trajectory pooling method with C3D descriptors and present a novel multi-scale trajectory-pooled 3D convolutional descriptor (MTC3D) for action recognition, as shown in Fig. 1. Specifically, multi-scale dense trajectories and C3D features of $conv4b$ and $conv5b$ layers are first computed from the input videos. Then we conduct max pooling on $conv4b$ and $conv5b$ feature maps of C3D to shrink their temporal dimensions to one. In this way, a 16-frame-long trajectory is mapped to 16 points on one corresponding pooled feature map. After two types of normalization techniques, we perform trajectory pooling to the normalized feature maps and obtain the proposed MTC3D descriptors.

The proposed descriptors have two merits: 3D ConvNets has the ability to extract discriminative and shift-invariant features from videos, while the trajectory pooling method captures the temporal information of videos contained in the multi-scale trajectories of

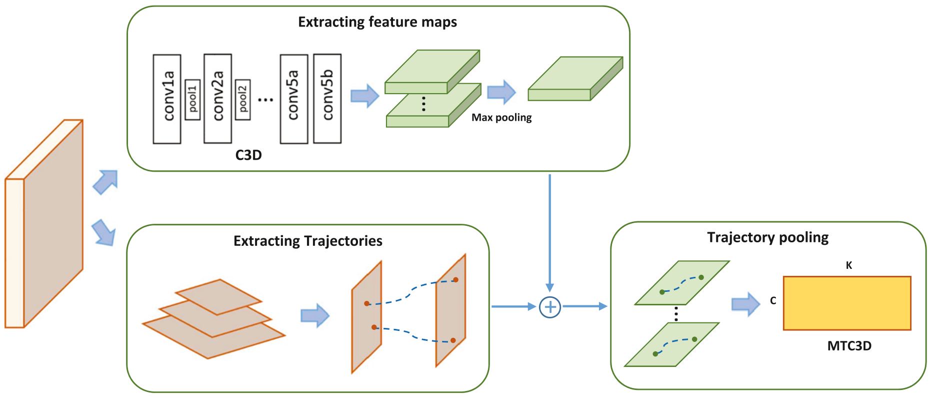


Fig. 1 The process of extracting MTC3D. The MTC3D framework contains three steps: extracting feature maps of C3D and dense trajectories from raw videos and conducting trajectory-constrained pooling method on the extracted feature maps. Finally, MTC3Ds that has $C \times K$ dimensions are obtained, where C is the number of channels of feature maps and K is the number of trajectories of the input video

objects. We should note that our MTC3D is different from the process of TDD. In TDD, the feature maps of two-stream ConvNets have the same number of frames to the input videos and trajectory pooling method is conducted on different frames of feature maps, whereas we pool the points on the feature maps whose temporal dimensions are one in proposed MTC3D. After MTC3Ds are gained, we employ Fisher vector to encode them and feed the encoding results into a linear SVM classifier. We evaluate the performance of MTC3D on two challenging action datasets: HMDB51 [20] and UCF101 [34]. The proposed MTC3D alone achieves 56.0% and 86.6% on HMDB51 and UCF101. MTC3D outperforms C3D (one net) by **4.3%** on UCF101 with the same pre-trained model.When combined with iDT, MTC3D obtains accuracies of 65.0% and 90.4% on HMDB51 and UCF101.

One preliminary version on trajectory-pooled 3D convolutional descriptor (TC3D) was first introduced in our previous work [25]. In this paper we make the following three improvements: (1) we add Section 2 to review hand-crafted descriptors and learning-based descriptors detailedly; (2) we integrate multi-scale motion information into TC3D and put forward MTC3D; (3) we conduct more comparative experiments and carry out error analysis of the results and discuss the advantages and disadvantages of MTC3D.

The remainder of this paper is organized as follows: In Section 2, We give an introduction to hand-crafted descriptors and learning-based descriptors. In Section 3, the proposed multi-scale trajectory-pooled 3D convolutional descriptor is introduced in detail. We report the experimental results on HMDB51 and UCF101 datasets in Section 4. Finally, the whole paper is concluded in Section 5.

2 Related works

There are two main types of features for video description: hand-crafted descriptors and learning-based descriptors, as presented below.

Hand-crafted descriptors Hand-crafted descriptors contain Histograms of Oriented Gradients (HOG) [4], Histograms of Optical Flows (HOF) [21], Motion Boundary Histograms (MBH) [5], HOG3D [18], 3D SIFT [30], and so on. The process of extracting hand-crafted descriptors mainly can be divided into two steps: the first step is to detect interest points using some interest-point detector [13] or in a dense way, and then the local information (e.g., pixel value, optical flow) or its gradient in the neighborhood of detected points is aggregated to construct a histogram. HOG [4], HOF [21] and MBH [5] describe the information of image gradients, optical flow and motion boundaries (i.e., gradients of optical flow) respectively. HOG3D [18] and 3DSIFT [30] imitate the process of HOG [4] and SIFT [24] and compute histograms of 3D spatio-temporal gradients. The most successful hand-crafted descriptor for action recognition so far is Improved Dense Trajectories (iDT) [39]. In essence, it extracts special spatio-temporal interest areas using dense trajectories where HOG, HOF, and MBH descriptors are calculated. iDT outperforms other hand-crafted descriptors in almost all the public action datasets. After getting hand-crafted descriptors, feature encoding methods such as bag-of-words (BoW) model [10], sparse coding [9, 48] or Fisher Vector [29] are applied to learn higher-level features and enhance the recognition results. These hand-crafted descriptors contain researchers' observation and experience and thus achieve great successes in this area. However, designing a good hand-crafted feature is difficult and time-consuming, and rely on expert knowledge. Moreover, hand-crafted descriptors are usually only applicable to certain applications and do not generalize well.

Learning-based descriptors A transformation from raw input to the representation is learned by machine learning methods for learning-based descriptors. At first, some shallow learning techniques were used in this domain. In [22], a stacked convolutional Independent Subspace Analysis network was proposed to learn invariant spatio-temporal features from videos. With the developments of deep learning, convolutional neural networks (CNN) [19], which is inspired by the behavior of the animal visual cortex, has been proved to be an effective feature learning technique in action recognition [17, 32]. CNN can be used to process videos in an end-to-end way or provide deep features as the input of feature encoding approaches and classifiers. Some deep descriptors (e.g., Deep ConvNets [17]) learned high-level features from raw videos directly by 2D convolutions. Convolutional 3D descriptors (C3D) [38] employed 3D convolution and 3D pooling operations to model the temporal information of the videos better and give superior results. Two-stream ConvNets [32] used two networks to handle the spatial and temporal information separately and the inputs of its spatial and temporal stream ConvNets are RGB frames and optical flow fields. Driven by the success on speech translation [12] and machine translation [36], Long Short-Term Memory (LSTM) [14] that is a special type of recurrent neural networks has been applied to model video sequences recently. In [8], Donahue et al. utilized LSTM to learn long-term dependencies in videos and developed Long-term Recurrent Convolutional Networks for three vision tasks (i.e., activity recognition, image description, and video description). Ng et al. [44] compared different convolutional temporal feature pooling architectures and LSTM to explore a better way of feature aggregation in time domain. In [35], the LSTM Encoder-Decoder framework was used to learn video representations in an unsupervised way. Sharma et al. [31] merged a soft attention based model into multi-layered LSTM, which learned to focus on the spatial areas in each frame that were relevant for the recognition task. These deep descriptors work well due to the high discriminative capacity and good generalization ability of deep neural networks. But in a sense, deep learning techniques are a black box and the features they learn are not easy to interpret.

3 Multi-scale trajectory-pooled 3D convolutional descriptors

In this section, we elaborate a new multi-scale trajectory-pooled 3D convolutional descriptor for video representation, as shown in Fig. 1. We first explain the extraction process of dense trajectories and 3D convolutional feature maps from the raw videos. Then, the feature map normalization and trajectory pooling steps are described in detail. We finally introduce the multi-scale strategy we use.

3.1 Dense trajectories

We adopt improved trajectories [39], which is originally used to compute iDT descriptor, to extract dense trajectories due to its good performance. Improved trajectories is a modified version of dense trajectories [40]. In dense trajectories, feature points are first sampled on a grid spaced by 5 pixels. Then each point is tracked by median filtering in a dense optical flow field $w = (u_t, v_t)$.

$$P_{t+1} = (x_{t+1}, y_{t+1}) = (x_t, y_t) + (M * w)|_{(\overline{x}_t, \overline{y}_t)} \tag{1}$$

where $P_t = (x_t, y_t)$ represents the feature point at frame t, M is the kernel for median filtering, and $(\overline{x}_t, \overline{y}_t)$ is the rounded position of (x_t, y_t). After the dense optical flow field is calculated, points of adjacent frames are linked to get the trajectories. To avoid drifting problem, the length of a trajectory is limited to 15 frames in [39]. Static trajectories and trajectories with sudden large displacements are also removed to make the obtained dense trajectories more robust.

Compared with dense trajectories, camera motion is considered in improved trajectories to enhance the performance. Camera motion is calculated based on the assumption that two adjacent frames are associated by a homography [37]. To compute the homography matrix, two complementary methods (i.e., SURF descriptor [2] and dense optical flow) are combined to find the matches between two frames at first. Afterward, the RANSAC approach [11] is applied to estimate the homography. Eventually, camera motion is removed to get a better optical flow that is more focused on foreground moving objects. In this way, the trajectories generated by background camera motion are suppressed to get small displacements and then removed by a thresholding method. In the proposed descriptor, the length of a trajectory is set at 16 frames to match the temporal length of the input clips of C3D. Given a video V, we get dense trajectories

$$T(V) = \{T_1, T_2, \cdots, T_K\} \tag{2}$$

where T_k represents the k^{th} trajectory of the video V:

$$T_k = \left\{\left(h_1^k, w_1^k, d_1^k\right), \left(h_2^k, w_2^k, d_2^k\right), \cdots, \left(h_P^k, w_P^k, d_P^k\right)\right\} \tag{3}$$

where (h_p^k, w_p^k, d_p^k) denotes the p^{th} point in trajectory T_k and P represents the length of a trajectory.

3.2 Convolutional feature maps

We employ 3D ConvNets [16, 38] to learning features from videos in MTC3D. 3D convolution and 3D pooling operations are adopted in 3D ConvNets. 3D convolution is the natural extension of 2D convolution. Both 3D convolution and 2D convolution can have multi-dimensional inputs, and the differences exist in the outputs. The outputs of 2D convolution are two-dimensional feature maps, whether its output has two or more dimensions, as shown in Fig. 2a and b. In contrast, the output volumes of 3D convolution can have multiple dimensions, as illustrated in Fig. 2c. In other words, 3D convolution conserves the temporal information of the input videos. Hence, we can utilize multiple 3D convolutional layers to handle the spatial and temporal information of the inputs in a hierarchical way simultaneously.

The architecture of C3D is illustrated in Tables 1 and 2. 3D convolution and pooling kernels with a size of $S \times S \times T$ are used, where S and T represent the spatial and temporal size of the kernels. C3D net has 8 convolution layers, which have $3 \times 3 \times 3$ convolutional

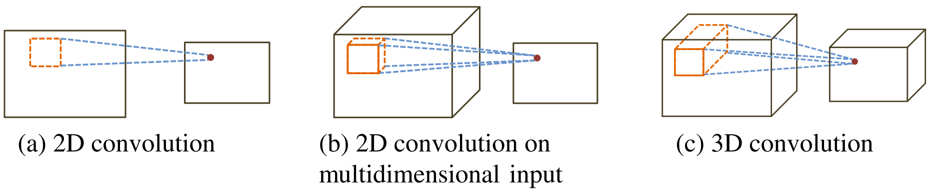


Fig. 2 2D and 3D convolution. **a** 2D convolution on two-dimensional input. **b** 2D convolution on multidimensional input. **c** 3D convolution on multidimensional input. The outputs of 2D convolution are always two-dimensional feature maps, while 3D convolution has multidimensional outputs

Table 1 The convolutional layers of the C3D Architecture

Layer	conv1a	conv2a	conv3a	conv3b	conv4a	conv4b	conv5a	conv5b
Size	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$	$3 \times 3 \times 3$
Stride	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$	$1 \times 1 \times 1$
Channel	64	128	256	256	512	512	512	512
Ratio	1	1/2	1/4	1/4	1/8	1/8	1/16	1/16

C3D net has 8 convolution layers. The kernel sizes of all of the convolution layers are $3 \times 3 \times 3$, with stride $1 \times 1 \times 1$. Ratio represents the spatial map size ratio

filters, with stride $1 \times 1 \times 1$. The kernel size of $pool1$ layer is $2 \times 2 \times 1$, with stride $2 \times 2 \times 1$. The other 4 max-pooling layers have $2 \times 2 \times 2$ pooling kernels, with stride $2 \times 2 \times 2$. In our experiments C3D net is used to cope with videos as convolutional feature extractors, not in an end-to-end way. Specifically, we compute feature maps of $conv4b$ and $conv5b$ layers from the input videos and the full-connected layers are abandoned.

We denote the size of the inputs or feature maps by $H \times W \times D \times C$, 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. Then the size of the input clips of C3D net is $112 \times 112 \times 16 \times 3$. The $conv4b$ and $conv5b$ feature maps has a size of $14 \times 14 \times 4 \times 512$ and $7 \times 7 \times 2 \times 512$ respectively. Whereafter, we conduct a max-pooling operation to reduce the temporal sizes of $conv4b$ and $conv5b$ feature maps to one. Finally given a clip V, the representation $F_v \in \mathbb{R}^{H \times W \times C}$ are gained, where H and W are 7 or 14 and C is 512.

3.3 Feature map normalization and trajectory pooling

Given the representation F_v, two types of normalization approaches (not shown in Fig. 1) are adopted as in TDD. The first one is spatiotemporal normalization. The result of a convolutional layer for each channel can be viewed as a spatiotemporal block. Spatiotemporal normalization is conducted by dividing the feature map values by the maximum value of the spatiotemporal block for each channel.

$$\widetilde{F}_{st}(h, w, c) = F(h, w, c)/max_{h,w}F(h, w, c) \tag{4}$$

The second normalization method is channel normalization, and the feature map values are divided by the maximum value in the same spatio-temporal position across different channels.

$$\widetilde{F}_{ch}(h, w, c) = F(h, w, c)/max_{c}F(h, w, c) \tag{5}$$

Table 2 The pooling layers of the C3D Architecture

Layer	pool1	pool2	pool3	pool4	pool5
Size	$2 \times 2 \times 1$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$
Stride	$2 \times 2 \times 1$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$	$2 \times 2 \times 2$
Channel	64	128	256	512	512
Ratio	1/2	1/4	1/8	1/16	1/32

C3D net has 5 pooling layers, which have $2 \times 2 \times 2$ pooling kernels, with stride $2 \times 2 \times 2$, except for $pool1$ layer. Ratio represents the spatial map size ratio

After normalization, the values of the points on feature maps are aligned into a same interval. In experiments, these two normalization approaches are used separately and their results $\widetilde{F}_{st}(h, w, c)$ and $\widetilde{F}_{ch}(h, w, c)$ are fused to further enhance the performance.

In C3D net, spatial and temporal padding are implemented on the convolutional layers to make its inputs and outputs have the same size. And the effect of the padding is that it create the mappings between the points in videos and those on feature maps. For example, the point with coordinate (h, w, d) in clip V corresponds to that with coordinate $(r \times h, r \times w)$ on the obtained representation F_v, where r is the spatial map size ratio calculated in advance, as shown in Tables 1 and 2. In this way, the points on the trajectories are mapped to those on current representations directly when conducting trajectory pooling.

Given a normalized feature map $\widetilde{F}$ and a trajectory T_k, trajectory pooling is carried out as follows:

$$D(T_k, \widetilde{F}) = \max_p \widetilde{F}\left(\overline{\left(r \times h_p^k\right)}, \overline{\left(r \times w_p^k\right)}, c\right) \tag{6}$$

where r is the spatial map size ratio, $\left(r \times h_p^k, r \times w_p^k\right)$ is mapped from the corresponding p^{th} point $\left(h_p^k, w_p^k, d_p^k\right)$ of original video in trajectory T_k, $\overline{(\cdot)}$ is the rounding operation. $D(T_k, \widetilde{F}) \in \mathbb{R}^{C \times K}$ is the designed trajectory-pooled 3D convolutional descriptor (TC3D), where C is the number of channels and K is the number of trajectories.

3.4 Multi-scale extension

Above we introduce the process of extracting TC3D on single scale. Following the idea of iDT, we compute the trajectories for multiple scales and put forward the multi-scale extension of TC3D, that is, the proposed multi-scale trajectory-pooled 3D convolutional descriptor (MTC3D). Specifically, We first densely sample the feature points for 8 spatial scale by a factor of $1/\sqrt{2}$. Then the feature points are tracked in each scale over 16 frames. Hence given a video V, we acquire multi-scale dense trajectories

$$\widehat{T}(V) = \{T_1, T_2, \cdots, T_{K_1}, \cdots, T_1, T_2, \cdots, T_{K_M}\} \tag{7}$$

where $\{T_1, T_2, \cdots, T_{K_m}\}$ represents the computed trajectories in m^{th} scale, and K_m is the number of trajectories. Then the proposed multi-scale trajectory-pooled 3D convolutional descriptor is $\widehat{D}(T_k, \widetilde{F}) \in \mathbb{R}^{C \times \widehat{K}}$, where $\widehat{K} = \sum_{m=1}^{M} K_m$.

4 Experiments

In this section, we test the proposed TC3D and MTC3D on two public datasets: HMDB51 [20] and UCF101 [34]. We first introduce the datasets and the implementation details. Afterward, the exploration experiments and the comparisons to other methods are presented in turn. Finally, we conduct the error analysis and the discussions on the merits and demerits of the proposed descriptors.

4.1 Datasets

Two challenging action datasets are employed in our experiments: HMDB51 and UCF101, as shown in Fig. 3. The HMDB51 dataset has 6766 video clips taken from movies, YouTube, Google videos, etc. This dataset contains five types of actions from general facial actions

Fig. 3 Sample frames from HMDB51 (first row) and UCF101 datasets (second row)

to body movements for human interaction and covers 51 action categories.We employ three training/testing splits and report average accuracy over three splits as in [20] . The UCF101 dataset involves realistic videos collected from YouTube. It includes 13320 video sequences and has 101 action classes. Each class has 25 groups and the sequences in the same group may share some common characteristics (e.g., similar background). We use the three training/testing splits as in [34] and also report average accuracy.

4.2 Implementation details

In the experiments, feature maps of *conv*4*b* and *conv*5*b* layers of C3D net are extracted, whose sizes are $14 \times 14 \times 4 \times 512$ and $7 \times 7 \times 2 \times 512$. When computing features maps from the HMDB51 and UCF101 datasets, we employ a C3D model that is pre-trained on Sports-1M and released by Tran et al. in [38]. After max pooling operation in temporal dimension, a representation whose size is $7 \times 7 \times 512$ or $14 \times 14 \times 512$ is acquired. We conduct spatiotemporal and channel normalization to the representation, and the two normalized representations and the original representation are utilized to compute different MTC3Ds, which will be fused to boost the experimental results. The 16-frame-long dense trajectories from videos are extracted because the input of C3D net is 16-frame-long clip. Then we obtain MTC3D with a size of $512 \times K$ by trajectory pooling, where K is the number of trajectories in this video. Next, PCA is used to reduce MTC3Ds to 128 dimensions to cut down the time and space overhead. After we get MTC3Ds from the videos, Fisher vector [29] is applied to encode them. We first build a dictionary of visual words by GMM with $G(G = 256)$ mixtures. Then we assign MTC3D to their nearest visual words and gain a vector with $2 \times 128 \times 256$ dimensions. At last, linear SVM is employed as the classifier.

Table 3 The performance of different trajectory pooling methods (with *conv*5 features and spatiotemporal normalization) on three splits of HMDB51 dataset

HMDB51	Split1	Split2	Split3	Ave
Sum pooling	52.6	50.4	49.7	50.9
Max pooling	52.6	51.8	49.6	51.3

Max pooling method outperforms average pooling method in the experiments

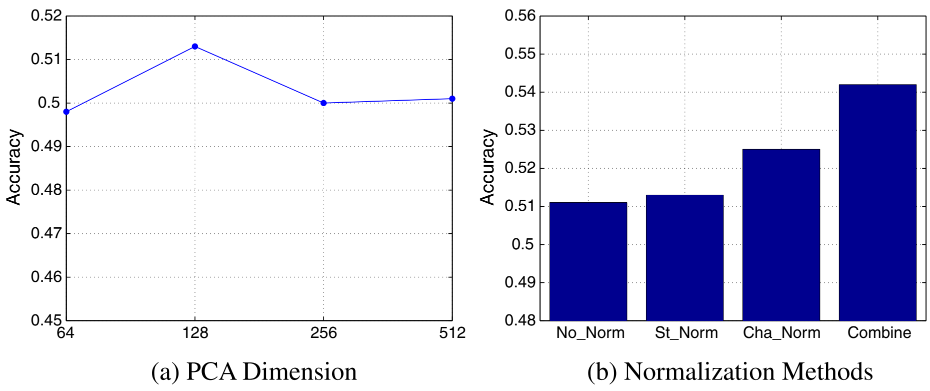


Fig. 4 The recognition results of different PCA dimensions and normalization methods with $conv5$ features on HMDB51. Dimension 128 and the fusion of normalization methods obtain the best results respectively

4.3 Exploration experiments

In this section, TC3D is used to explore the impact of different settings in steps of the proposed pipeline, due to its lower time and space costs compared to MTC3D. We first evaluate the performance of sum pooling and max pooling methods in trajectory pooling step on three splits of HMDB51. TC3D with $conv5$ features and spatiotemporal normalization are used in the experiments and the results are summarized in Table 3. The average accuracy of max pooling is 0.4 higher than sum pooling. Therefore max trajectory pooling is chose in the proposed descriptor.

We employ TC3D with $conv5$ features and spatiotemporal normalization and investigate the impact of different PCA dimensions in Fig. 4a. Dimension 128 gets the best performance among them. Thus, TC3Ds and MTC3Ds are reduced to 128 dimensions and then fed into Fisher vector in the whole experiments. In Fig. 4b, we use TC3D with $conv5$ features and describe the average accuracy of different normalization methods. St_Norm and Cha_Norm represents spatiotemporal normalization and channel normalization respectively. No_Norm stands for the original representation without normalization. Combination of them is 3.1% better than No_Norm, which demonstrates the effects of the normalization methods.

Table 4 reports the recognition results of TC3D with different convolutional layers. We see that the combination of $conv4$ and $conv5$ improves the average accuracy, which indicates that TC3Ds with different layers are complementary to each other. Table 5 illustrates the average accuracy of TC3D and MTC3D. MTC3D computes multi-scale dense trajectories and captures richer motion information and outperforms TC3D on these two datasets.

Table 4 The average accuracy of TC3D with different convolutional layers on HMDB51 and UCF101

Datasets	conv4	conv5	conv4 + conv5
HMDB51	49.7	54.2	55.5
UCF101	83.2	83.1	86.5

The combination of features from different layers improves the recognition accuracy significantly

Table 5 The average accuracy of TC3D and MTC3D on HMDB51 and UCF101

Datasets	TC3D	MTC3D
HMDB51	55.5	56.0
UCF101	86.5	86.6

MTC3D captures multi-scale information and surpasses TC3D on two datasets

4.4 Comparison to the state of the art

We compare the proposed TC3D and MTC3D with other algorithms and summarize the action recognition accuracy in Table 6. The upper part shows the recognition methods whose inputs are only RGB videos. The lower part presents other algorithms that take both RGB frames and precomputed optical flow fields as inputs. We can observe that TC3D and MTC3D combined with Fisher vector and linear SVM perform much better than HOG descriptor and other RGB videos based deep neural networks, containing Deep networks [17], Spatial stream network [32], LRCN [8] and LSTM composite model [35]. TC3D and MTC3D also outperform C3D [38] and conv4 and conv5 spatial layers of TDD [41]. MTC3D and C3D (1 net) use the same pre-trained model in the whole experiments and MTC3D performs **4.3%** better than C3D (1 net) on UCF101. The results indicate

Table 6 Action recognition results on HMDB51 and UCF101

Method	HMDB51	UCF101
HOG [39] + FV	40.2	72.4
Deep networks [17]	–	65.4
Spatial stream network [32]	39.0	72.6
LRCN [8]	–	71.1
LSTM composite model [35]	44.1	75.8
C3D (1 net) [38]	–	82.3
Spatial conv4 and conv5 of TDD [41] + FV	50.0	82.8
TC3D + FV	**55.5**	**86.5**
MTC3D + FV	**56.0**	**86.6**
iDT [39] + FV	57.2	84.7
Two-stream networks [32]	59.4	88.0
LRCN [8]	–	82.9
LSTM composite model [35]	–	84.3
Conv. pooling on long clips [44]	–	88.2
LSTM on long clips [44]	–	88.6
TDD [41] + FV	63.2	90.3
TC3D and iDT + FV	**64.5**	**90.1**
MTC3D and iDT + FV	**65.0**	**90.4**

The upper part shows the recognition methods whose inputs are only RGB frames. The lower part presents the algorithms that take both RGB frames and optical flow fields as inputs. MTC3D outperforms C3D (1 net) by **4.3%** on UCF101 with the same pre-trained model

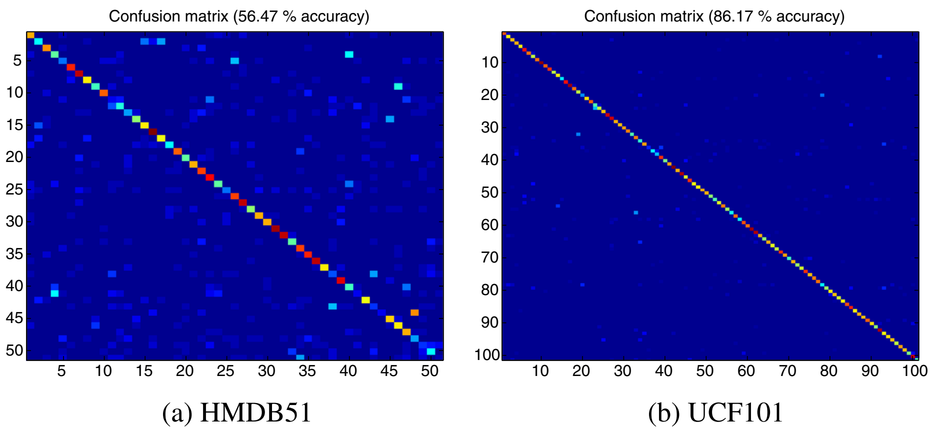


Fig. 5 The confusion matrices of the recognition results of TC3D on Split1 of HMDB51 and UCF101 datasets

that trajectory pooling method captures the inherent nature of temporal dimension and promotes the recognition accuracy. When united with iDT descriptors, MTC3D performs better than other deep learning methods whose inputs are RGB frames and optical flow fields and achieves state-of-the-art results. The confusion matrices of the recognition results using TC3D on Split1 of HMDB51 and UCF101 are illustrated to give an intuitive view in Fig. 5.

4.5 Error analysis

Some misclassified samples of HMDB51 dataset are displayed in Fig. 6. We illustrate four main reasons about the misclassifications. The first one is that a video may contain multiple

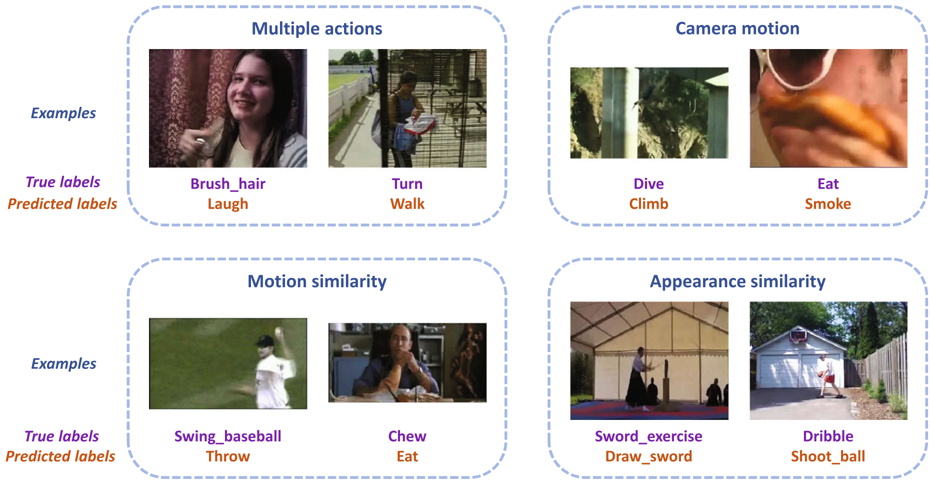


Fig. 6 Misclassified samples of HMDB51 dataset. The first line under the examples is the true labels, and the second line represents the predicted labels. There are four reasons that cause the misclassifications: multiple actions (top left), camera motion (top right), motion similarity (bottom left), and appearance similarity (bottom right)

actions, which is an inherent problem for action recognition task. Two example videos are shown on the top left and in the first one a girl is brushing her hair and laughing at the same time. It is hard to classify the video correctly, even to human labelers. The second reason is camera motion as shown on the top right. Camera motions (e.g., pan, tilt and zoom) produce the background motion and also interfere the foreground motion, which degrades the recognition results. Shot changes can also fall into this category roughly, which sometimes result in unpredictable classification outputs.

Motion similarity is the third reason as shown on the bottom left of Fig. 6. Some actions share similar body motions. For example, both swinging baseball and throwing may contain the motions of holding the object over the head and throwing it out. These motions generate similar motion-based features, which makes the classification extremely difficult. The last one is appearance similarity shown on the bottom right. Two actions can have the same scene, background or objects, which leads to similar appearance-based features. For instance, both dribbling and shooting ball occur at the basketball court and relate to the basketball and the basketball stand.

Camera motion increases the intra-class variation, while motion and appearance similarity reduces inter-class distance. And the first reason that multiple actions exist in one video has both of these two roles. The four reasons mentioned above bring great difficulties and challenges for action recognition task. From the misclassified examples, we can see that the mistakes are reasonable and the proposed descriptor indeed ”understands” the video samples. These recognition errors also indicate potential directions on how to further improve the discriminative ability and the robustness of descriptors next.

4.6 Discusion

The proposed MTC3D extracts discriminative deep features from the inputs and meanwhile captures the temporal information of videos by the trajectory pooling method. Furthermore, compared to TDD [41], there is no need to train temporal network for optical flow frames when extracting MTC3D.

However, MTC3D performs worse than some recent works, such as TSN [43] and TLE [7]. A primary reason is that new techniques are used in these works. For example, the main idea of TSN is that the input video is divided into several segments which are processed by different spatial and temporal stream ConvNets and the class scores of these segments are fused to obtain a video-level prediction. TLE follows the idea of TSN and adds a temporal encoding layer besides. These techniques (i.e., segmenting videos and adding feature encoding layer in the network) can also be incorporated into MTC3D to further improve the recognition accuracy. In this paper, we focus on integrating trajectory pooling method with C3D descriptors and thus do not utilize the above techniques. Another reason is that we make use of the C3D model pre-trained on Sports-1M directly owing to the limits of our computing power and storage capacity. For example, the spatial size of $conv4b$ and $conv5b$ feature maps in C3D net is only 14×14 and 7×7, which affects the performance of MTC3D. Thus adopting new techniques in our pipeline and training new 3D ConvNets will be our future work.

5 Conclusion and future work

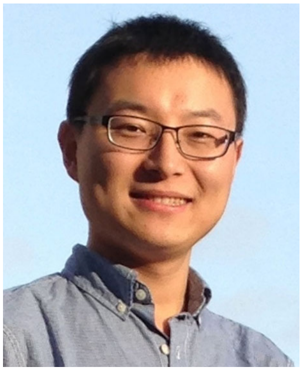
In this paper, we combine C3D with dense trajectories and present a new multi-scale trajectory-pooled 3D convolutional descriptor for action recognition. We take advantage of both 3D ConvNets that extracts high-level features from videos and trajectory pooling strategy that utilizes important motion information. Experiments validate the superior performances of the proposed descriptor on two challenging datasets. Based on the discussion section above, we will imitate C3D net to design our own 3D ConvNets that is more suitable for trajectory pooling method and add the feature encoding layer in the network in future.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No. 61472103).

References

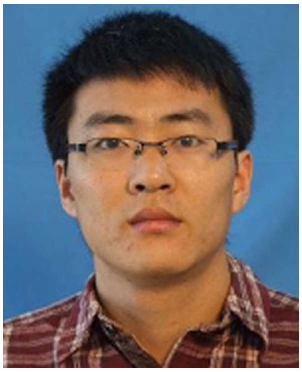
1. Aggarwal JK, Ryoo MS (2011) Human activity analysis: a review. ACM Comput Surv (CSUR) 43(3):16
2. Bay H, Tuytelaars T, Van Gool L (2006) Surf: speeded up robust features. In: Computer vision–ECCV 2006, pp 404–417
3. Boiman O, Irani M (2007) Detecting irregularities in images and in video. Int J Comput Vis 74(1):17–31
4. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: IEEE Computer society conference on computer vision and pattern recognition, 2005. CVPR 2005, vol 1. IEEE, pp 886–893
5. Dalal N, Triggs B, Schmid C (2006) Human detection using oriented histograms of flow and appearance. In: Computer vision–ECCV 2006, pp 428–441
6. Demiris Y, Khadhouri B (2006) Hierarchical attentive multiple models for execution and recognition of actions. Robot Autonom Syst 54(5):361–369
7. Diba A, Sharma V, Van Gool L (2016) Deep temporal linear encoding networks. arXiv:1611.06678
8. Donahue J, Anne Hendricks L, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2625–2634
9. Fanello SR, Gori I, Metta G, Odone F (2013) Keep it simple and sparse: real-time action recognition. J Mach Learn Res 14(1):2617–2640
10. Fei-Fei L, Perona P (2005) A bayesian hierarchical model for learning natural scene categories. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005, vol 2. IEEE, pp 524–531
11. Fischler MA, Bolles RC (1981) Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun ACM 24(6):381–395
12. Graves A, Jaitly N (2014) Towards end-to-end speech recognition with recurrent neural networks. In: Proceedings of the 31st international conference on machine learning (ICML-14), pp 1764–1772
13. Harris C, Stephens M (1988) A combined corner and edge detector. In: Alvey vision conference, vol 15, no 50. Manchester, pp 5210–5244
14. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
15. Jhuang H, Serre T, Wolf L, Poggio T (2007) A biologically inspired system for action recognition. In: IEEE 11th international conference on computer vision, 2007. ICCV 2007. IEEE, pp 1–8
16. Ji S, Xu W, Yang M, Yu K (2013) 3d convolutional neural networks for human action recognition. IEEE Trans Pattern Anal Mach Intell 35(1):221–231
17. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Fei-Fei L (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1725–1732
18. Klaser A, Marszałek M, Schmid C (2008) A spatio-temporal descriptor based on 3d-gradients. In: BMVC 2008-19th British machine vision conference. British Machine Vision Association, pp 275–1
19. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems, pp 1097–1105
20. Kuehne H, Jhuang H, Garrote E, Poggio T, Serre T (2011) Hmdb: a large video database for human motion recognition. In: 2011 IEEE international conference on computer vision (ICCV). IEEE, pp 2556–2563
21. Laptev I, Marszałek M, Schmid C, Rozenfeld B (2008) Learning realistic human actions from movies. In: IEEE conference on computer vision and pattern recognition, 2008. CVPR 2008. IEEE, pp 1–8

22. Le QV, Zou WY, Yeung SY, Ng AY (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. In: 2011 IEEE Conference on computer vision and pattern recognition (CVPR). IEEE, pp 3361–3368
23. Liu AA, Su YT, Nie WZ, Kankanhalli M (2017) Hierarchical clustering multi-task learning for joint human action grouping and recognition. IEEE Trans Pattern Anal Mach Intell 39(1):102–114
24. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vis 60(2):91–110
25. Lu X, Yao H, Sun X, Zhang S, Zhang Y (2017) Trajectory-pooled 3d convolutional descriptors for action recognition. In: Pacific rim conference on multimedia
26. Nie W, Liu A, Li W, Su Y (2016) Cross-view action recognition by cross-domain learning. Image Vis Comput 55:109–118
27. Poppe R (2010) A survey on vision-based human action recognition. Image Vis Comput 28(6):976–990
28. Rautaray SS, Agrawal A (2015) Vision based hand gesture recognition for human computer interaction: a survey. Artif Intell Rev 43(1):1–54
29. Sánchez J, Perronnin F, Mensink T, Verbeek J (2013) Image classification with the fisher vector: theory and practice. Int J Comput vis 105(3):222–245
30. Scovanner P, Ali S, Shah M (2007) A 3-dimensional sift descriptor and its application to action recognition. In: Proceedings of the 15th international conference on multimedia. ACM, pp 357–360
31. Sharma S, Kiros R, Salakhutdinov R (2015) Action recognition using visual attention. arXiv:1511.04119
32. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems, pp 568–576
33. Snoek CG, Worring M (2008) Concept-based video retrieval. Found Trends Inf Retriev 2(4):215–322
34. Soomro K, Zamir AR, Shah M (2012) Ucf101: a dataset of 101 human actions classes from videos in the wild. arXiv:1212.0402
35. Srivastava N, Mansimov E, Salakhutdinov R (2015) Unsupervised learning of video representations using lstms. In: International conference on machine learning, pp 843–852
36. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks. In: Advances in neural information processing systems, pp 3104–3112
37. Szeliski R (2006) Image alignment and stitching: a tutorial. Founda Trends Comput Graph Vis 2(1):1–104
38. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision, pp 4489–4497
39. Wang H, Schmid C (2013) Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision, pp 3551–3558
40. Wang H, Kläser A, Schmid C, Liu CL (2011) Action recognition by dense trajectories. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, pp 3169–3176
41. Wang L, Qiao Y, Tang X (2015) Action recognition with trajectory-pooled deep-convolutional descriptors. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4305–4314
42. Wang F, Qi S, Gao G, Zhao S, Wang X (2016) Logo information recognition in large-scale social media data. Multimed Syst 22(1):63–73
43. Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X, Van Gool L (2016) Temporal segment networks: towards good practices for deep action recognition. In: European conference on computer vision. pp 20–36
44. Yue-Hei Ng J, Hausknecht M, Vijayanarasimhan S, Vinyals O, Monga R, Toderici G (2015) Beyond short snippets: deep networks for video classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4694–4702
45. Zhao S, Chen L, Yao H, Zhang Y, Sun X (2015) Strategy for dynamic 3d depth data matching towards robust action retrieval. Neurocomputing 151:533–543
46. Zhao S, Yao H, Gao Y, Ji R, Xie W, Jiang X, Chua TS (2016) Predicting personalized emotion perceptions of social images. In: Proceedings of the 2016 ACM on multimedia conference. ACM, pp 1385–1394
47. Zhao S, Yao H, Gao Y, Ji R, Ding G (2017) Continuous probability distribution prediction of image emotions via multitask shared sparse regression. IEEE Trans Multimed 19(3):632–645
48. Zhu Y, Zhao X, Fu Y, Liu Y (2011) Sparse coding on local spatial-temporal volumes for human action recognition. Comput Vis–ACCV 2010:660–671

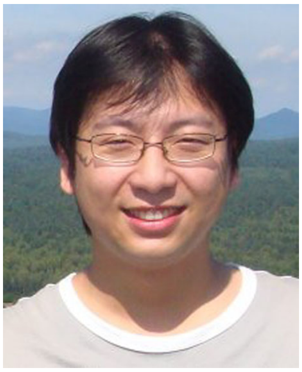


Xiusheng Lu is currently a Ph.D. candidate at Harbin Institute of Technology, Harbin, China. His research interests include action recognition and deep learning.

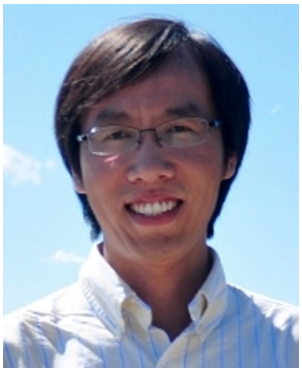
Hongxun Yao received the B.S. and M.S. degrees in Computer Science from the Harbin Shipbuilding Engineering Institute, Harbin, China, in 1987 and in 1990, respectively, and received Ph.D. degree in Computer Science from Harbin Institute of Technology, Harbin, China, in 2003. Currently, she is a Professor with School of Computer Science and Technology, Harbin Institute of Technology. Her research interests include computer vision, multimedia computing, and human-computer interaction. She has published five books and over 200 scientific papers.



Sicheng Zhao received Ph.D. degree in Computer Science from Harbin Institute of Technology, Harbin, China, in 2016. He is currently a Postdoctoral Research Fellow in the School of Software, Tsinghua University, Beijing, China. His research interests include affective computing, social media analysis and multimedia information retrieval.



Xiaoshuai Sun received the B.S. degree in Computer Science from Harbin Engineering University in 2007. He received the M.S and PH.D degree in Computer Science and Technology from Harbin Institute of Technology in 2009 and 2015 respectively. He is currently an assistant professor School of Computer Science and Technology, Harbin Institute of Technology. He was a Postdoctoral researcher in the University of Queensland (2015-2016). He was a Research Intern with Microsoft Research Asia (2012-2013) and also a winner of Microsoft Research Asia Fellowship in 2011. He owns 2 authorized patents and has authored over 70 referred journals and conference papers in IEEE Transactions on Image Processing, Pattern Recognition, ACM Multimedia, and IEEE CVPR.



Shengping Zhang received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2013. He is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology at Weihai. He has been a Postdoctoral Research Associate with Brown University, Providence, RI, USA, and a Visiting Student Researcher with University of California at Berkeley, CA, USA. He has authored or co-authored over 50 research publications in refereed journals and conferences. His research interests include sparse coding and its applications in computer vision. Dr. Zhang is also an Associate Editor of Signal Image and Video Processing.