Neural Processing Letters (2020) 51:287–302
https://doi.org/10.1007/s11063-019-10091-z

Action Recognition with Multiple Relative Descriptors of Trajectories

Zhongke Liao[1] · Haifeng Hu[1] · Yichu Liu[1]

Published online: 2 August 2019

Abstract
Dense trajectory has become one of the most successful hand-crafted features for action recognition. However, most of the existing dense trajectories based methods ignore the relationship between trajectories. In this paper, we propose multiple relative descriptors of trajectories to model the relative information of pairs of trajectories. Specifically, we present relative motion descriptors and relative location descriptors, which are utilized to capture the relative motion information and relative location information respectively. Moreover, we present relative deep feature descriptors which combine the deep features with hand-crafted features. By aggregating the above descriptors, we obtain the fixed-length representation regardless of the various duration of input video. The experimental results on three standard datasets demonstrate the superiority of our method.

Keywords Action recognition · Dense trajectories · Multiple relative descriptors

1 Introduction

Human action recognition has become a hot topic in computer vision due to its potential applications in video analysis, virtual reality and video surveillance. Although remarkable progress has been made, we still face several technical issues. One of the thorniest issues is that the camera motion can result in relative displacements between human and background, which is easily misclassified as a part of action.

Before the surge of deep learning, the hand-crafted features are widely used for action recognition. In recent years, the dense trajectories based methods [1–5] have been dominant among all the hand-crafted features. Improved Dense Trajectories [2], which suppresses the camera by estimating the camera position and removing the trajectories of background region, shows its impressive performance in action recognition. However, the key limitation of this type of algorithm is that the trajectories are often described by classical hand-crafted

✉ Haifeng Hu
huhaif@mail.sysu.edu.cn

Zhongke Liao
liaozhk5@mail2.sysu.edu.cn

[1] School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China

features such as HOG and HOF, which may not be optimized for visual representation and lack discriminative capacity for action recognition. To solve the problem, many researchers aggregate the hand-crafted and deep features to obtain a discriminative representation. Wang et al. [6] incorporated IDT into a deep learning framework by a linear mapping method. Unfortunately, these methods employ each trajectory separately and ignore the relative information between them.

At present, deep learning has made great progress in the field of action recognition. One of the most representative deep models is two stream networks [7] which contain two independent parts, namely spatial network as well as temporal network respectively. The spatial network aims to extract the appearance features from static RGB image, while the temporal network takes 3D volume of stacking optical flows fields as input to process the temporal information. Wang et al. [8] proposed to aggregate the deep features over snippets sparsely sampled from the video. Zhu et al. [9] proposed a key volume mining deep framework to identify key volumes and conduct classification simultaneously. Varol et al. [10] employed neural networks with long-term temporal convolutions to learn the video representations. Diba et al. [11] embedded temporal linear encoding into CNNs as a new layer to capture the appearance and motion throughout entire videos. However, these CNN based methods typically take the raw video as input without any processing for camera motion, making the spatio-temporal dynamic extraction of human action more challenging.

In this paper, we explore the relative information between trajectories and propose multiple relative descriptors to acquire robust representation. Specifically, improved dense trajectories are extracted firstly, and three different types of relative information, i.e. relative motion information, relative location information, and relative deep information can be obtained from the pairs of the trajectories. By introducing multiple relative descriptors that share the merits of both IDT and deep features, our strategies can depict the relative motion of the background and the foreground. Meanwhile, these relative information helps to capture and encode the variation of motion, which is suitable for classifying two similar kinds of action, as shown in Fig. 1. In order to reduce the computational cost, we apply clustering algorithm to construct a set of trajectory groups, each of which contains several trajectory pairs corresponding to a specific 'codeword'. By aggregating these relative descriptors, we obtain a discriminative representation with fixed length.

The contributions of the paper are as follows:

First, we propose a new method to represent the trajectories using multiple relative descriptors. These descriptors, which represent the relationship between foreground or background trajectories, can capture richer information than using each trajectory separately.

Second, our method not only inherits the excellent property of IDT's robustness to camera motion, but also integrates the relative information between trajectories into CNN to extract more discriminative features.

Third, for short videos, although we can only extract a small amount of trajectories, our method can get more information by considering the relationship between them, namely, the n trajectory pairs contain C_n^2 relative relationships.

Finally, for long videos with a large number of trajectories, we adopt clustering algorithm to map the relative descriptors to the corresponding codewords. As a codeword can represent a motion area or a part of the background, our method can not only encode the dynamics of the foreground effectively, but also reduce the computation cost remarkably.

The rest of the paper is organized as follows. We review the related work in Sect. 2. Section 3 details the multiple relative descriptors of trajectories. Experiment results and discussions on KTH, JHMDB and HMDB51 are provided in Sect. 4. Finally, Sect. 5 concludes this paper.

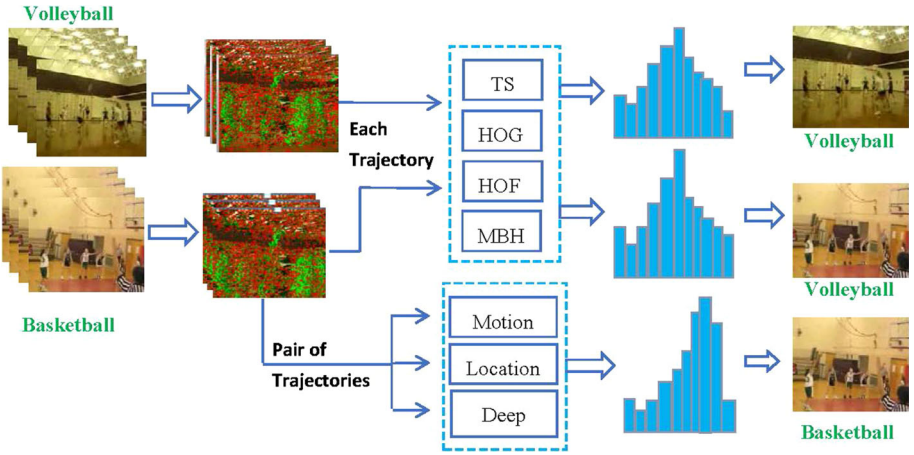

Fig. 1 The illustration of the merits the proposed method. With the classical method which extract trajectory from video clips separately, it may obtain the similar feature vector from two distinctive human actions, e.g., play volleyball and basketball. By introducing latent relative information between trajectories, our model can obtain the correct results, as shown in the bottom line

2 Related Work

In this section, we review the main research efforts in the area of action recognition. Many popular approaches mainly involve three stages: feature extraction, video representation and video classification.

In the feature extraction stage, there are two different types of features, the hand-crafted features and the deep-learned features respectively, which are shown in Fig. 2.

Early hand-crafted features can be divided into two categories: motion based features and appearance based features. Motion-based methods treat the action recognition as temporal classification, which highly rely on human foreground segmentation and body tracking. For example, Yamato et al. [12] modelled the class-specific HMMs and obtain the grid-based silhouette features. Appearance based features explore the spatial discriminative local features [13–15]. Dollar et al. [13] developed an extension of informative features points based on space-time windowed data. Yeffet et al. [14] proposed to fuse the Local Binary Patterns (LBP) with the spatial invariance, which is based on the patch-matching methods. The mostly used appearance features are local descriptors based on space-time interest points (STIPs) [15] and cuboids [16], such as SIFT and SURT and so on. Improved dense trajectories (IDT) [2] with four descriptors, trajectory shape (TS), histograms of oriented gradients (HOG), histograms of optical flow (HOF), and the motion boundary histograms (MBH), has achieved state-of-the-art performance among all the hand-crafted features on standard action datasets.

Due to the great success of convolutional neural network, researchers have focused on applying deep learning to action recognition. For instance, [17,18] concentrated on learning local spatio-temporal convolutional filter, while [7,19,20] incorporated optical flow snippets into the deep learning architecture in order to process temporal information. Karpathy et al. [17] applied CNN to time domain to fuse the local spatio-temporal information. Simonyan et al. [7] incorporate the motion information by using two independent convolutional neural networks for both spatial domain and temporal domain. Feichtenhofer et al. [21] improved the two-stream convolutional neural network by studying a fusion method which makes the

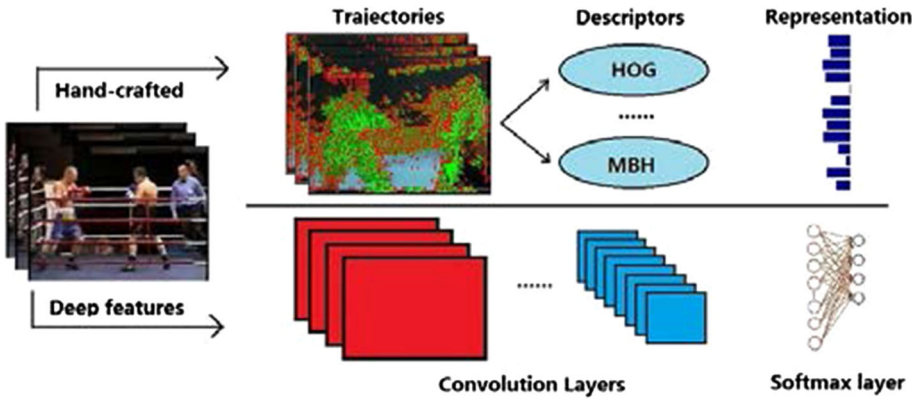

Fig. 2 Two types of features used for action recognition, i.e., hand-crafted features and deep learning features

related channels to be connected. Wang et al. [8] proposed to aggregate the information over snippets sparsely sampled from the video. Zhu et al. [9] proposed a key volume mining deep framework to identify key volumes and conduct classification simultaneously. Varol et al. [10] employed neural networks with long-term temporal convolutions to learn the video representations. Diba et al. [11] embedded temporal linear encoding into ConvNets as a new layer to capture the appearance and motion throughout entire videos. However, the deep networks take the raw video as inputs without properly handling the camera motion, which may bring significant interference to the final classification.

In the stage of video representation, there are three widely used mid-level representation approaches: bag-of-words (BOW) [22], fisher vector (FV) [23] and vector of locally aggregated descriptors (VLAD) [24,25]. According to [2], VLAD outperforms BOW and FV under the same conditions. Generally, VLAD can be viewed as a simplified form of FV encoding, which is a type of feature representation aggregating the descriptors based on a locality criterion in the feature space.

In the stage of classification, classifiers such as support vector machine (SVM), Nearest Neighbor Classifier (NNC) and Random Forest are trained with the obtained representation and then applied to the test data.

3 Proposed Algorithm

In this section, we will introduce the proposed MRDT method, a trajectory based approach that models the dynamic information of a video. Figure 3 illustrates the overview of MRDT. Firstly, we extract improved dense trajectories (IDT) from the video due to their robustness to camera motion. Then we propose multiple relative descriptors of trajectories to model the human action in the video via capturing the relative information between pairs of trajectories. Compared with IDT, MRDT is not only robust to camera motion, but also works well on the videos with arbitrary length.

We start with a brief introduction to IDT and then detail the multiple relative descriptors of trajectories which are designed to model the dynamics of the video. These descriptors are eventually aggregated together and sent to a linear SVM for classification.

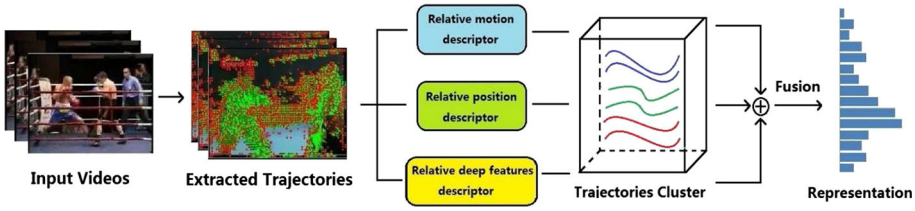

Fig. 3 The overview of proposed MRDT. It includes the following steps: (i) Improved dense trajectories extraction. (ii) Relative relationship exploration. (iii) Trajectories cluster. (iv) Different relative descriptors fusion

3.1 Improved Dense Trajectories

As an extension of dense trajectory, IDT [2] can effectively suppress camera motion without losing the motion information of foreground. The first step for IDT extraction is to densely sample a set of points in 8 spatial scales with a grid step size of 5 pixels from video frames. The tracking of the sampled point can be defined as follow:

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

where $P(x_t, y_t)$ is the tracked point of trajectory at tth frame, $*$ is the convolutional operation, ϕ is dense optical flow field of the tth frame, and M is the median filter kernel, and $(\overline{x}, \overline{y})$ denotes the round position of (x, y). The length of trajectory is set to 15 frames to avoid the drifting problem. Furthermore, the static trajectories as well as those with suddenly large displacements are removed.

Homography matrix is adopted in IDT to characterize the motion of two consecutive frames. First, SURF [26] features as well as motion vectors extracted from optical flow are used to generate sufficient candidate matches of two adjacent frames. Then the Homography matrix is estimated by RANSAC [27] algorithm. Using the estimated homography matrix to recalculate the optical flow, *warped flow* can be obtained, which can suppress various camera motion e.g. pan, tilt and zoom [2]. The effects of IDT are shown in Fig. 4. We can learn that IDT can remove the background trajectories caused by the camera motion, and preserve the foreground trajectories that contain more discriminative information.

Different from the original IDT, we only use two descriptors (e.g., HOG and MBH [28]) of the trajectories to capture the appearance and motion information.

In summary, for a given video V, it can be represented as:

$$T(V) = \{T_1, T_2, \ldots, T_k\} \tag{2}$$

$$T_k = \{(x_1^k, x_1^k), (x_2^k, x_2^k), \ldots, (x_L^k, x_L^k)\} \tag{3}$$

where K is the number of video trajectories, and T_k is the kth trajectory. (x_l^k, x_l^k) denotes the position of the lth point in trajectory T_k, while L is the length of trajectory ($L = 15$). The design of multiple relative descriptors for the extracted trajectories will be introduced in the following parts.

3.2 Relative Relationship of Trajectories

Most trajectories based methods merely design descriptors for each trajectory separately, which ignore the significance of the relationships between them. However, these relationships with a lot of useful information can be used to build a more powerful representation. Based

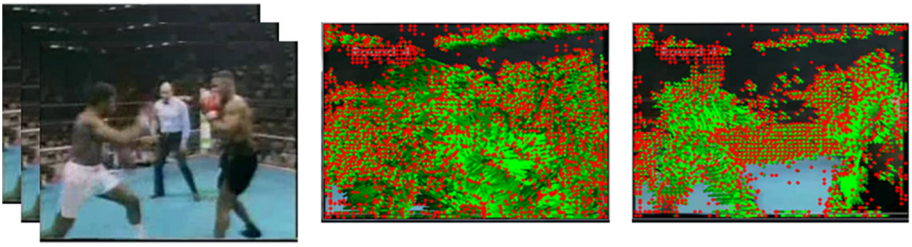

Fig. 4 Left: RGB sequence from the raw video; middle: Dense Trajectories that contain the camera motion; right: Improved Dense Trajectories (IDT) that suppress the camera motion. The red dots are dense trajectory positions and green lines are the trajectories. (Color figure online)

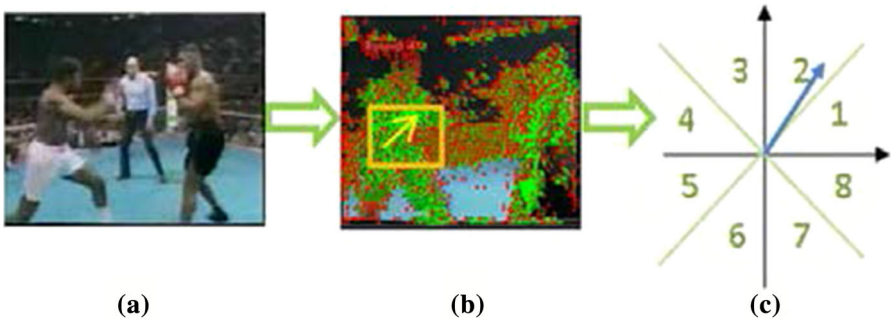

Fig. 5 A proposed method to quantify the vectors which are used to represent the relative relationship of trajectories. **a** denotes the input video. The relative motion/location of two trajectories in **b** is mapped to a 8-bins relative feature by quantizing the local descriptors of the two trajectories in **c**

on above considerations, we explore the relative relationships between trajectories to obtain discriminative representations. Specifically, three relative descriptors, i.e. relative motion descriptors, relative location descriptors and relative deep features descriptors, are proposed to model the dynamics of the video.

Relative Motion Descriptors We propose relative motion descriptors to characterize the relative motion information between a pair of trajectories. Since each trajectory contains L points, we define the motion information of a trajectory as follows:

$$M_{T_k} = P_L^k(x_L^k, y_L^k) - P_1^k(x_1^k, y_1^k) \tag{4}$$

where M_{T_k} represents the motion of the trajectory, P_l^k is the lth points of the kth trajectory. Then the relative motion between a pair of trajectories can be represented as:

$$\Delta M_{i,j} = M_{T_i} - M_{T_j} \tag{5}$$

By quantifying relative motion information, we obtain the relative motion descriptors. As shown in Fig. 5, the angle plane of $\Delta M_{i,j}$ will be divided into N bins from 45° to 360°, each of them corresponds to an angle interval of $360°/N$. Therefore, the relative motion $\Delta M_{i,j}$ is quantized into an N-dimensional vector $RMD_{i,j}$, which takes the advantage of the motion orientation and magnitude.

Relative Location Descriptors The position of a trajectory is quite crucial, which is closely related to the region where the human action may occur. The position of a trajectory can be denoted as (6):

Table 1 The architecture of VGG16

Layer	Size	Stride	Channel	Map size ratio
conv1_1	3×3	64	1	1
conv1_2	3×3	64	1	1
pool1	2×2	64	2	1/2
conv2_1	3×3	128	1	1/2
conv2_2	3×3	128	1	1/2
pool2	2×2	128	2	1/4
conv3_1	3×3	256	1	1/4
conv3_2	3×3	256	1	1/4
conv3_3	3×3	256	1	1/4
pool3	2×2	256	2	1/8
conv4_1	3×3	512	1	1/8
conv4_2	3×3	512	1	1/8
conv4_3	3×3	512	1	1/8
pool4	2×2	512	2	1/16
conv5_1	3×3	512	1	1/16
conv5_2	3×3	512	1	1/16
conv5_3	3×3	512	1	1/16
pool5	2×2	512	2	1/32
fc6	–	4096	–	–
fc7	–	4096	–	–
fc5	–	101	–	–

$$\widetilde{P_i} = \left(\frac{\Delta x_1 + \Delta x_2 + \cdots + \Delta x_L}{L}, \frac{\Delta y_1 + \Delta y_2 + \cdots + \Delta y_L}{L} \right) \tag{6}$$

where $(\Delta x_i, \Delta y_i)(i = 1, \ldots, L)$ denotes the position of the ith points of the trajectory, and $\widetilde{P_i}$ is the mean position of all points. Similar to the extraction of relative motion descriptors, the relative location between two trajectories is calculated as follows:

$$\Delta P_{i,j} = \widetilde{P_i} - \widetilde{P_j} \tag{7}$$

Then, the $\Delta P_{i,j}$ is quantized into a N-dimensional vector $RLD_{i,j}$, which incorporates the relative location of two trajectories. If the direction of relative location $\Delta P_{i,j}$ is assigned to one of N bins, the corresponding value in the vector will set to one while others are zero.

Relative Deep Feature Descriptors Deep ConvNets have proven their superiority in many areas, such as face recogniton and object detection. In our work, we choose 16-layer VGG16 [29] as the backbone of two stream networks to extract the deep features of the trajectories. As a deep ConvNet with 13 convolutional layers and 3 fully connected layers, VGG16 has higher precision and better generalization capacity. The architecture of VGG16 is shown in Table 1.

Two stream networks consist of two independent networks, the spatial network and the temporal network respectively. The spatial network aims to capture the appearance information of the video, with the RGB frame ($224 \times 224 \times 3$) as input. The temporal network is applied to model the dynamic information, and its input is the stacking optical flows fields ($224 \times 224 \times 2K$, K is the number of stacks). Notably, the parameters of two networks are

not shared. Once the training of two stream networks is completed, we treat them as generic feature extractors to obtain the feature maps from the last convolutional layer. The detail of both the implementation and the training will be introduced in Sect. 4. Therefore, for a video V, we can extract two types of feature maps:

$$C_S(V) = \{C_1^s, C_2^s, \ldots, C_I^s\} \tag{8}$$

$$C_S(T) = \{C_1^t, C_2^t, \ldots, C_I^t\} \tag{9}$$

where $C_i^s, C_i^t \in \mathbb{R}^{H_i \times W_i \times D \times Ch_i}$ is the ith feature maps of the spatial and temporal network respectively, and H_i, W_i, D, Ch_i denote the height, width, video duration and the number of channels respectively.

The feature map normalization is then applied to reduce the influence of illumination. Given a feature map $C \in \mathbb{R}^{H \times W \times D \times Ch}$, the value of C can be normalized as:

$$C_{Norm} = \frac{C}{max(V_{ST}^j)} \tag{10}$$

where $max(V_{ST}^j)$ is the maximum value in the spatio-temporal domain of the feature map in jth channel. This operation ensures that the value of the points in the same position ranges in the same scale through all channels.

To obtain the deep features of a trajectory, the points of improved dense trajectories are mapped into these normalized feature maps. In order to boost the performance, we construct the multi-scale deep feature of the trajectories. Specifically, we extract the multi-scale pyramids of video frames and optical flow fields, and feed these pyramids into two stream networks to obtain multi-scale convolutional feature maps. Given the feature maps C_S and C_T from spatial network and temporal network respectively, the deep feature of a trajectory can be defined as follows:

$$\begin{aligned} Td_S(T_k) &= \sum_{s=1}^{M} \sum_{l=1}^{L} C_S^{\sigma_s}(\overline{(r_m \times \sigma_s \times x_l^k)}, \overline{(r_m \times \sigma_s \times y_l^k)}) \\ Td_T(T_k) &= \sum_{s=1}^{M} \sum_{l=1}^{L} C_T^{\sigma_s}(\overline{(r_m \times \sigma_s \times x_l^k)}, \overline{(r_m \times \sigma_s \times y_l^k)}) \end{aligned} \tag{11}$$

where (x_l^k, y_l^k) is the position of the lth point in trajectory T_k, and r_m is the map size ratio with respect to input size. $\overline{()}$ is the rounding operation. M is the total number of the scale, and σ_s denotes the sth scale of the feature maps. From formula (11) we can obtain $Td_S(T_k)$ and $Td_T(T_k)$, which are the deep features of T_k from spatial network and temporal network respectively. In practice, we use 4 scales with $\sigma_s = \sqrt{2}^{s-4}$, where s ranges from 1 to 4. Finally, we define the relative deep feature descriptor between a pair of trajectories as follows:

$$RDFD_{i,j}^m = Td_m(T_i) + Td_m(T_j)(m = S, T) \tag{12}$$

Fusion of Multiple Relative Descriptors We aggregate three types of relative descriptors between a pair of trajectories into a fixed-dimension representation as follows:

$$f(T_i, T_j) = [RMD_{i,j}, RLD_{i,j}] \times \left\| \Delta M_{i,j} \right\| \times \left\| \Delta P_{i,j} \right\| \times \left| RDFD_{i,j} \right| \tag{13}$$

where $\Delta M_{i,j}$, $\Delta P_{i,j}$ are the magnitude of relative motion and relative location respectively.

As the number of trajectories varies in different videos, we need the fixed-length representation regardless of the duration of the video. Inspired by the bag-of-features [30], we

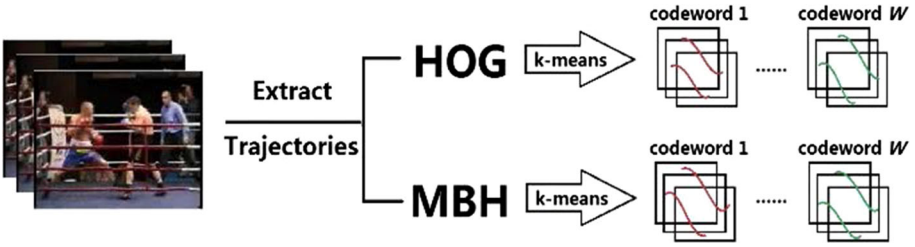

Fig. 6 The process of clustering trajectories through k-means and mapping them to corresponding codewords

introduce k-means [31] to cluster the trajectory descriptors to obtain a codebook with W codewords, in which all the trajectories are mapped to their corresponding codewords. In our work, we only use HOG and MBH for trajectories in clustering. The procedure of clustering is shown in Fig. 6. Given a pair of trajectories (T_i, T_j), we can obtain the corresponding codeword pair (C_i, C_j). Then we define representation F to be the summation of the multiple relative descriptor vectors $f(T_i, T_j)$, which denotes the relative relationship of codeword pairs, shown as follow:

$$F(C_i, C_j) = \sum_{(T_i,T_j)\to(C_i,C_j)} f(T_i, T_j) \tag{14}$$

Therefore, we reduce the computational cost by clustering a large number of trajectories into a small quantity of codeword pairs, as described in formula (14). Since W codewords are generated by k-means, the final representation between all codewords can be expressed with matrix M:

$$M = \begin{bmatrix} 0, & F_{1,2}, & \dots, & F_{1,W} \\ F_{2,1}, & 0, & \dots, & F_{2,W} \\ \vdots & & \ddots & \vdots \\ F_{W,1}, & F_{W,2}, & \dots, & 0 \end{bmatrix} \tag{15}$$

where M is an antisymmetric matrix. In our work, only the upper triangle of the matrix is utilized to represent the video. Thus, the final representation of MRDT is a $\frac{W\times(W-1)}{2} \times N$-dimensional vector.

4 Experiments

In this section, we first make a brief introduction of the datasets and the experimental protocols, and then discuss the performance of the proposed method as well as its comparisons with other state-of-the-art methods.

4.1 Experimental Settings

Datasets To test the effectiveness of MRDT, we conducted experiments on three standard datasets: JHMDB [32], KTH [33] and HMDB51 [34]. As shown in Fig. 7, KTH dataset contains six types of actions: handclapping, boxing, walking, jogging, running and hand waving. The dataset has a total of 599 videos, and all videos for each class are grouped into

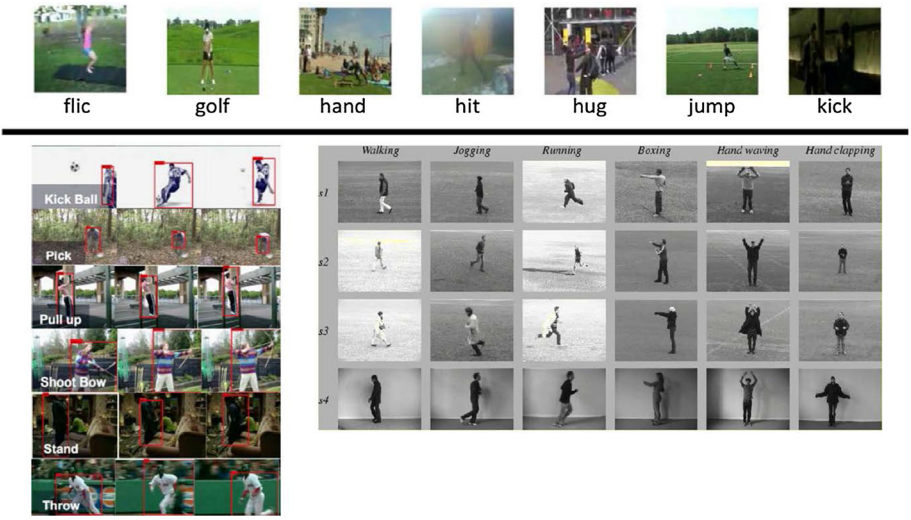

Fig. 7 Example frames of a few action classes in HMDB51 (first row), JHMDB (left side of second row), and KTH (right side of second row) datasets

25 subjects. As in Baccouche et al. [35], we randomly selected 16 subjects for training and the remaining 9 subjects for testing.

JHMDB contains 21 types of actions, such as golf, jump and pick, etc. This dataset has a total of 928 videos. Figure 7 shows a few frames from several categories in the JHMDB dataset. We follow [33] and divide the dataset into three train/test splits for evaluation. The final performance is reported by the average across three splits.

HMDB51 contains 51 kinds of actions, with a total of 6766 videos and at least 100 video clips in each category. Moreover, most of the videos in this dataset involve a mass of camera motion, e.g. pan and zoom, which makes the action recognition more challenging. The dataset provides three train/test splits. In each split, 70 clips are used for training and 30 clips for testing.

Implementation details We use the video extension version [8] of the Caffe toolbox [36] to implement the proposed MRDT.

TV-L1 [37] is applied to compute optical flow, then we discretize the values of optical flow fields into integers and set their range as 0–255 just like images. VGG16, pre-trained on ImageNet, is used as backbone of two stream networks to acquire the deep features of the trajectories. UCF101 [38], one of the biggest action datasets containing 13,320 videos, is selected to train two stream networks. We use stochastic gradient descent solver with cross-entropy loss to optimize the networks, whose batch size is 128 and weight decay is 0.00001. For spatial network, we first resize the frame to make the smaller side as 256, and then randomly crop a 224×224 region from the frame. It then undergoes random horizontal flipping. The dropout ratios are set to 0.8 for all fully-connected layers. For the temporal network, the input is the volume of stacking optical flows fields ($224 \times 224 \times 20$), and the dropout ratios for fully connected layers are set to 0.9. The initial learning rate for both networks is 0.001 and decreases by a factor of 10 after 30K iterations. All the videos in UCF101 are used to train our deep models.

Table 2 Evaluation of parameter Nbins on JHMDB dataset

Nbins	4	8
split1 (%)	51.33	55.49
split2 (%)	51.65	58.00
split3 (%)	50.92	53.86
Average(%)	51.33	55.78

After the training on UCF101 is completed, we fine-tune the deep models on KTH, JHMDB and HMBD51. In this procedure, we only update the parameters of three fully connected layers and those of the last convolutional layer to avoid severe overfitting. The training settings are the same as those of UCF101. Then we remove the three fully connected layers of the trained networks and only use the feature maps from the last convolutional layer to build our MRDT. In order to deal with multi-class classification problem, we apply linear support vector machine (SVM) with the LIBSVM toolbox and set the parameter C as 100.

All the experiments are conducted on Tianhe-2A, one of the fastest supercomputers in the world. Tianhe-2A possesses 16,000 computational nodes, each with 24 CPU cores and 128 GB RAM. We use four nodes to train the ConvNets, two nodes for optical flow computation and two nodes for the IDT extraction as well as MRDT implementation.

4.2 Quantitative Evaluation Results of MRDT

In this section, we conduct several experiments to support our claim that the proposed MRDT can capture more discriminative information by modelling the relationship between trajectories. Firstly, we test the effects of different parameters $Nbins$ and $NCenter$ on recognition performance, which denote the number of quantization bins of the proposed relative descriptors and the number of codewords for k-means respectively. Secondly, we evaluate MRDT on KTH, JHMDB and HMDB51 to validate the effectiveness of introducing relative information between trajectories. Then, we analyze the runtime of our method on HMDB51 and test its performance across datasets from JHMDB to HMDB51. Finally, we compare our approach with state-of-the-art methods on three standard datasets.

Selection of parameter Nbins and NCenter In order to acquire proper parameters $Nbins$ and $NCenter$, we explore the performance of the different parameter settings on JHMDB.

Table 2 shows the experimental results of different $Nbins$ with a fixed number of codewords at 100. With 8 bins, Relative motion descriptor and relative location descriptor (RM+RL) achieves the accuracy of 55.78%, 4.45% higher than that of RM+RL with only 4 bins. Since more bins will increase the dimension of the descriptor, we set $Nbins$ to 8 as default, in order to balance the accuracy and the computational efficiency.

Then we fix $Nbins$ at 8 and change $NCenter$. As shown in Table 3, setting $NCenter$ at 200 in the stage of clustering achieves the best performance. We finally set $NCenter$ to 200 as default. The experimental results indicate that the reasonable choice of $Nbins$ and $NCenter$ can yield better performance with less computation cost.

Evaluation of MRDT on JHMDB, HMDB51 and KTH We carry out ablation studies to evaluate the performance of MRDT on JHMDB, HMDB51 and KTH, in order to verify the effectiveness of introducing relative information between trajectories. The baseline is the trajectories without any kinds of relative descriptors. Then RM+RL, i.e. combining relative

Table 3 Evaluation of parameter NCenter on JHMDB dataset

NCenter	100	200	300
split1 (%)	55.49	59.63	57.30
split2 (%)	58.00	60.49	58.94
split3 (%)	53.86	59.60	57.91
Average(%)	55.78	59.90	58.05

Table 4 The performance of MRDT on JHMDB, HMDB51 and KTH datasets

Datasets	Baseline (%)	RM+RL (%)	MRDT (%)
JHMDB	49.72	59.90	61.73
HMDB51	44.35	53.23	56.13
KTH	80.25	92.67	95.27

Table 5 The performance of MRDT with different scale RDFD

Scale	1	2	3	4
JHMDB (%)	61.73	62.21	63.68	65.13
HMDB51 (%)	56.13	57.17	58.23	59.87
KTH (%)	95.27	95.72	96.57	97.77

motion descriptor and relative location descriptor, is evaluated. Finally, we test the performance of MRDT that consists of all relative descriptors.

As shown in Table 4, both RM+RL and MRDT can achieve better performance than baseline on three datasets, which demonstrates that the proposed MRDT can capture more discriminative information. Moreover, we analyze the results to illustrate the superiority of our model. First, RM+RL outperforms the baseline, suggesting that the relative relationships between trajectories are critical to boost the performance. Second, MRDT has higher accuracy than RM+RL, indicating that relative deep features are complementary to other two relative hand-crafted features.

We further report the performance of MRDT with multi-scale relative deep feature descriptor (RDFD) in Table 5. With the help of multi-scale RDFD, MRDT can obtain more deep features to model the relationship between trajectories, thus achieving higher accuracy. It is worth noting that multi-scale RDFD means higher computational costs, which can significantly reduce the real-time performance of our model.

The runtime analysis We list the average inference runtime of our method on each video in Table 6. The experiments are conducted on HMDB51. Firstly, the optical flow fields are calculated from the video, as they are indispensable for both improved dense trajectories and two stream networks. Then we extract the improved dense trajectories and obtain the deep features using two stream networks. Afterwards, we model the relationship between trajectories via multiple relative descriptors. Finally, a linear SVM is used for classification.

As shown in Table 6, optical flow extraction is time-consuming because of the dense calculation between each two consecutive frames. As we extract the deep feature of each frame and each stacking optical flow field, the process of CNN feature extraction takes about 3.86 seconds per video. The FPS of IDT is 12.53, which is almost twice that of optical flow extraction. It's worth pointing out that MRDT works very fast, whose FPS is 57.14. In

Table 6 The runtime on HMDB51

Step	Optical flow	IDT	CNN	MRDT	SVM	Total
Runtime	14.02 s	7.46 s	3.86 s	1.64 s	0.41 s	27.39 s
FPS	6.67	12.53	24.19	57.14	227.33	3.41

Table 7 The performance of MRDT across datasets (trained on JHMDB, tested on HMDB51)

Dataset	Accuracy (%)
JHMDB	65.13
HMDB51	59.04
HMDB51[a]	59.87

[a]denotes the results without cross-dataset testing

general, our method lacks real-time capability with only 3.41 FPS, due to the fact that both IDT and optical flow extraction are time-consuming.

The performance across datasets For further evaluation, we test the robustness of MRDT across two datasets that contain complex camera motion. Specifically, we train our model on JHMDB and then test it on HMDB51, as the categories in JHMDB are the same as part of those in HMDB51. All the videos in JHMDB are used for training to avoid overfitting. The training settings are the same as Sect. 4.1. After training, we test our model only on test splits of HMDB51, and list the results in Table 7. Notably, only the accuracy of the shared categories in these two datasets are reported.

As shown in Table 7, the performance drops by 6.09% when we transfer MRDT directly from JHMDB to HMDB51, mainly due to the fact that the number of videos for testing in HMDB51 is more greater than that of videos in JHMDB. But it can be seen clearly that the accuracy across datasets is very closed to the original result that we post in Table 5, which demonstrates that the performance of MRDT does not depend on specific dataset and the robustness of MRDT to camera motion is excellent.

Comparison to state-of-the-art methods The comparisons of the MRDT with other algorithms on KTH, JHMDB and HMDB51 are listed in Tables 8, 9 and 10 respectively.

As shown in Table 8, MRDT outperforms the previous methods on KTH. The LSTM model with HOF and HOG3D achieve 90.7% and 89.93%, respectively. The accuracy of 3DCNN model is 91.04%; when combined with the LSTM, the accuracy increases to 94.39%. MRDT outperforms 3DCNN+LSTM by 3.3%.

We also compare MRDT with existing methods on JHMDB dataset. From Table 9, we can learn that our method outperforms both IDT [2] and deep learning approaches. Gkioxari et al. [39] incorporated pose estimation into their deep model with an accuracy of 62.50%. Cheron et al. [40] not only fine-tuned their model on JHMDB, but also used a detection framework for action modelling. With the deep models that are only treated as generic feature extractors on JHMDB, MRDT achieves higher accuracy than those of CNN based methods.

Finally, we make comparisons between MRDT and other algorithms on HMDB51. Wang et al. [2] utilized four different descriptors (e.g., TS, HOG, HOF and MBH) for the trajectories, while we only HOF and MBH for clustering. Simonyan et al. [7] propose an end-to-end model and fine-tune their networks on HMDB51. Although their deep models are also pre-trained on UCF101, MRDT still outperforms [7].

Table 8 Comparisons on KTH dataset

Method	Accuracy (%)
Rodriguez et al. [41]	81.50
Jhuang et al. [42]	91.70
LSTM + HOF [43]	90.70
LSTM + HOG3D [44]	89.93
1-order dRNN+HOG3D [44]	93.28
2-order dRNN+HOG3D [44]	93.96
3DCNN [35]	91.04
3DCNN+LSTM [35]	94.39
Ours	97.77

Table 9 Comparisons on JHMDB dataset

Method	Accuracy (%)
Wang et al. [1]	56.60
Simonyan et al. [7]	56.50
Gkioxari et al. [39]	62.50
Cheron et al. [40]	61.10
Ours	65.13

Table 10 Comparisons on HMDB51 dataset

Method	Accuracy (%)
Wang et al. [2]	55.9
Simonyan et al. [7]	58.50
Wang et al. [29]	55.1
Tran et al. [18]	51.9
Ours	59.87

The comparisons on three datasets demonstrate the superiority of MRDT, which indicates that introducing the relative information between trajectories, along with combining the deep features with these information, can significantly boost the performance.

5 Conclusion

This paper proposes multiple relative descriptors of trajectories for action recognition. Unlike the traditional trajectories based approaches which describe each trajectory separately, our method considers the relative information between trajectories. Specifically, relative motion descriptors and relative location descriptors are utilized to capture the relative motion information and relative location information respectively. Moreover, we use VGG16 to obtain the relative deep features of the trajectories, which shares the advantages of deep features with hand-crafted features. Experiments results on standard datasets demonstrate the effectiveness of our MRDT. The only drawback is that our method is not so efficient due to the poor real-time performance of optical flow calculation and IDT extraction. In the future, we will focus on fast and robust representations that can be applied in real time.

Acknowledgements This work was supported in part by the National Natural Science Foundation of China (61673402, 61273270, 60802069), in part by the National Key R&D Program of China under Grant 2018YFB1601101, the Natural Science Foundation of Guangdong Province (2017A030311029, 2019B010140002), the Science and Technology Program of Guangzhou of China (201704020180), and the Fundamental Research Funds for the Central Universities of China.

References

1. Wang H, Klaser A, Schmid C, Liu CL (2013) Dense trajectories and motion boundary descriptors for action recognition. Int J Comput Vis 103:60–79
2. Wang H, Schmid C (2013) Action recognition with improved trajectories. In: International conference on computer vision, pp 3551–3558
3. Jain M, Jegou H, Bouthemy P (2013) Better exploiting motion for better action recognition. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 2555–2562
4. Ramana Murthy OV, Goecke R (2013) Ordered trajectories for large scale human action recognition. In: Proceedings of IEEE conference on computer vision and pattern recognition works, pp 412–419
5. Seo JJ, Son J, Kim H, Neve WD, Ro YM (2015) Efficient and effective human action recognition in video through motion boundary description with a compact set of trajectories. In: Proceedings of IEEE international conference on automatic face and gesture recognition works, pp 1–6
6. Wang LM, Qiao Y, Tang XO (2015) Action recognition with trajectory-pooled deep-convolutional descriptors. In: IEEE conference on computer vision and pattern recognition IEEE computer society, pp 4305–4314
7. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. Adv Neural Inf Process Syst 1(4):568–576
8. Wang L, Xiong Y, Wang Z, Qiao Y, Lin D, Tang X (2016) Temporal segment networks: towards good practices for deep action recognition. In: ECCV, 22(1):20–36
9. Zhu W, Hu J, Sun G, Cao X, Qiao YA key volume mining deep framework for action recognition. In: Computer vision and pattern recognition. IEEE, pp 1991–1999
10. Varol G, Laptev I, Schmid C (2016) Long-term temporal convolutions for action recognition. IEEE Trans Pattern Anal Mach Intell 99:1
11. Diba A, Sharma V, Gool LV (2017) Deep temporal linear encoding networks. In: CVPR, pp 1541–1550
12. Yamato J, Ohya J, Ishii K (1992) Recognizing human actions in time-sequential images using hidden Markov model. In: IEEE conference on compute vision and pattern recognition, pp 379–385
13. Dollar P, Rabaud V, Cottrell G, Belongie S (2005) Behavior recognition via sparse spatio-temporal features. In: VS-PETS
14. Yeffet L, Wolf L (2009) Local trinary patterns for human action recognition. In: Proceedings of the IEEE international conference on computer vision, pp 492–4976
15. Zhu Y, Chen W, Guo G (2014) Evaluating spatiotemporal interest point features for depth-based action recognition. Image Vis Comput 32(8):453–464
16. Klaser A, Marszalek M, Schmid C (2008) A spatio-temporal descriptor based on 3D-gradients. In: BMVC
17. Karpathy A, Toderici G, Shetty S, Leung T, Sukthankar R, Li FF (2014) Large-scale video classification with convolutional neural networks. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1725–1732
18. Tran D, Bourdev L, Fergus R, Torresani L, Paluri M (2015) Learning spatiotemporal features with 3D convolutional networks. In: Proceedings of ICCV
19. Donahue J, Hendricks LA, Guadarrama S, Rohrbach M, Venugopalan S, Saenko K, Darrell T (2015) Long-term recurrent convolutional networks for visual recognition and description. In: Proceedings of CVPR
20. Ng JY-H, Hausknecht M, Vijayanarasimhan S, Vinyals O, Monga R, Toderici G (2015) Beyond short snippets: deep networks for video classification. In: Proceedings of CVPR
21. Feichtenhofer C, Pinz A, Zisserman A (2016) Convolutional two-stream network fusion for video action recognition, pp 1933–1941
22. Sivic J, Zisserman A (2003) Video Google: a text retrieval approach to object matching in videos. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 1470–1477
23. Sanchez J, Perronnin F, Mensink T, Verbeek J (2013) Image classification with the fisher vector: theory and practice. Int J Comput Vis 105:222–245
24. Gong Y, Wang L, Guo R, Lazebnik S (2014) Multi-scale orderless pooling of deep convolutional activation features. In: European conference on computer vision, pp 392–407

25. Jegou H, Douze M, Schmid C, Perez P (2010) Aggregating local descriptors into a compact image representation. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 3304–3311
26. Bay H, Ess A, Tuytelaars T (2008) Speeded-up robust features (SURF). Comput Vis Image Underst 110(3):346–359
27. Fischler A, 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
28. Dalal N, Triggs B, Schmid C (2006) Human detection using oriented histograms of flow and appearance. In: European conference on computer vision. Springer, Heidelberg, pp 428–441
29. Wang X, Gao L, Song J, Shen H (2016) Beyond frame-level cnn: saliency-aware 3-D CNN with LSTM for video action recognition. IEEE Signal Process Lett 24(4):510–514
30. Liu J, Luo J, Shah M (Jun. 2009) Recognizing realistic actions from videos in the wild. In: Proceedings of IEEE conference on CVPR, pp 1996–2003
31. Hartigan JA, Wong MA (1979) Algorithm AS 136: a K-means clustering algorithm. J R Stat Soc 28(1):100–108
32. Jhuang H, Gall J, Zuffi S, Schmid C, Black MJ (2013) Towards understanding action recognition. In: ICCV
33. Schuldt C, Laptev I, Caputo B (2004) Recognizing human actions: a local SVM approach. In: Proceedings of the 17th international conference on pattern recognition, vol 3, pp 32-36
34. Kuehne H, Jhuang H, Garrote E, Poggio T, Serre T (2011) HMDB: a large video database for human motion recognition. In: Proceedings of ICCV
35. Baccouche M, Mamalet F, Wolf C, Garcia C, Baskurt A (2011) Sequential deep learning for human action recognition. In: Human behavior understanding, pp 29–39
36. Jia Y, Shelhamer E, Donahue J, Karayev S, Long J, Girshick RB, Guadarrama S, Darrell T (2014) Caffe: convolutional architecture for fast feature embedding, CoRR, arXiv:1408.5093
37. Zach C, Pock T, Bischof H (2007) A duality based approach for realtime TV-L 1 optical flow. In: Pattern recognition, pp 214–223
38. Soomro K, Zamir AR, Shah M (2012) UCF101: a dataset of 101 human actions classes from videos in the wild, Technical Report CRCV-TR-12-01, UCF Center for Research in Computer Vision
39. Gkioxari G, Malik J (2015) Finding action tubes. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 759–768
40. Cheron G, Laptev I, Schmid C (2015) P-CNN: pose-based CNN features for action recognition. In: IEEE international conference on computer vision, pp 3218–3226
41. Rodriguez M, Ahmed J, Shah M (2008) Action MACH: a spatio-temporal maximum average correlation height filter for action recognition. In: IEEE conference on computer vision and pattern recognition, pp 1–8
42. Jhuang H, Serre T, Wolf L, Poggio T (2007) A biologically inspired system for action recognition. In: IEEE 11th international conference on computer vision, pp 1–8
43. Grushin A, Monner DD, Reggia JA, Mishra A (2013) Robust human action recognition via long short-term memory. In: International joint conference on neural networks, pp 1–8
44. Veeriah V, Zhuang NF, Qi GJ (2015) Differential recurrent neural networks for action recognition. In: IEEE international conference on computer vision, pp 4041–4049