IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 12, 2018, accepted December 11, 2018, date of publication December 17, 2018,
date of current version January 11, 2019.

Digital Object Identifier 10.1109/ACCESS.2018.2887144

Fall Detection in Videos With Trajectory-Weighted Deep-Convolutional Rank-Pooling Descriptor

ZHIMENG ZHANG, XIN MA, HANBO WU, AND YIBIN LI
School of Control Science and Engineering, Shandong University, Jinan 250061, China

Corresponding author: Xin Ma (maxin@sdu.edu.cn)

This work was supported in part by the National High-Tech Research and Development (863 Program) of China under Grant 2015AA042307 and in part by the National Natural Science Foundation of China under Grant U1706228 and Grant 61673245.

ABSTRACT Automatic fall detection in videos could enable timely delivery of medical service to the injured elders who have fallen and live alone. Deep ConvNets have been used to detect fall actions. However, there still remain problems in deep video representations for fall detection. First, video frames are directly inputted to deep ConvNets. The visual features of human actions may be interfered with surrounding environments. Second, redundant frames increase the difficulty of time encoding for human actions. To address these problems, this paper presents trajectory-weighted deep-convolutional rank-pooling descriptor (TDRD) for fall detection, which is robust to surrounding environments and can describe the dynamics of human actions in long time videos effectively. First, CNN feature map of each frame is extracted through a deep ConvNet. Then, we present a new kind of trajectory attention map which is built with improved dense trajectories to optimally localize the subject area. Next, the CNN feature map of each frame is weighted with its corresponding trajectory attention map to get trajectory-weighted convolutional visual feature of human region. Further, we propose a cluster pooling method to reduce the redundancy of the trajectory-weighted convolutional features of a video in the time sequence. Finally, rank pooling method is used to encode the dynamic of the cluster-pooled sequence to get our TDRD. With TDRD, we get superior result on SDUFall dataset and get comparable performances on UR dataset and Multiple cameras dataset with SVM classifiers.

INDEX TERMS Fall detection, descriptor, deep ConvNets, trajectory attention map, cluster pooling, rank pooling.

I. INTRODUCTION

The number of people 60+ is growing faster than any other age groups in worldwide, which is estimated to increase from 688 million in 2006 to around two billion by 2050 [1]. In China, the population over 65 years old was about 8.87% in 2010 and is expected to rise to 30% in 2050 [2]. As the WHO has reported, falling is a very serious problem among elderly people. About 28-35% of people aged over 65 falls each year, and the percentage goes up to 32-42% for people 70+. Falling is the primary cause of death from injury for seniors aged 79+ [3]. Lack of timely assistance makes death more likely from falling for elderly people living alone. Falling without timely assistance not only causes physical injury, but also causes mental trauma. Seniors who once fell and had to remain on the floor for long time until someone discovered them prefer to lie in bed rather than do exercises out of fear of falling again. A lack of exercise could further lead to other health problems. Fortunately, autonomous fall detection technique provides a potential solution for elderly people.

Mainstream autonomous fall detection approaches may broadly be divided into two categories: non-visual wearable sensors-based (the most common are accelerometers and gyroscopes) and exclusively vision-based [4]. Wearable sensors-based methods require elder people to wear the sensors, which cause inconvenience to them [5]. Without any intrusion of elder people's normal daily life, vision-based methods detect falls by analyzing video stream. Advances in computer vision have made it possible to automatically detect human fall with commodity RGB cameras and depth

cameras. Human shape change is the first clue for fall detection. Human silhouettes are approximated as an ellipse or a bounding box.

Some geometric attributes, such as aspect ratio, orientation [6], edge points [5], and curvature scale space (CSS) features [7] are extracted as fall characteristics. The trajectory of a moving subject is also used for detecting falls with monocular systems [8].

The above vision-based methods need to extract the subject from the background, which is inclined to be influenced by image noise, illumination variation and occlusion [9], [10]. In recent years, deep learning has achieved great success in computer vision. It also has been applied for fall detection. Without the requirement of either a background detection or foreground extraction, deep learning directly learns efficient features from the video streams. Faster R-CNN scene analysis method is used to detect human and furniture [11]. By measuring the changes of the activity characteristics of the detected people, such as human shape aspect ratio, and judging the relations between the people and furniture nearby, the falls on furniture are detected. In [12], a three-dimensional convolutional neural network is applied to encode both the spatial information over each frame and the temporal inf-ormation in adjacent frames. A LSTM (Long Short-Term Memory)-based visual attention map is established for local-izing areas of interest such as moving subjects. However, the attention maps which are computed with an end-to-end network may be not stable in complex surrounding environments limited by the number of training data. In [13], the appearance and motion information of raw input frames are encoded as a dynamic image. With a deep ConvNet, the dynamic images are classified as falling or not. In [14], convolutional neural network is used on optical flow of interest of features for fall detection.

There still remain problems in deep video representations for fall detection at present. Firstly, for human fall detection, we only focus on the human actions and ignore the surrounding environments. Most of the existing fall detection dataset are collected in simple and static indoor environments. Features of subjects of videos are easily to be extracted with deep learning. However, videos may be collected with camera motion (on home service robots) or other various complex scenes in real applications. So the features of subjects may be interfered by these complex surrounding environments. In order to solve this problem, the current mainstream methods take end-to-end trained attention maps [12], [18]–[20] to locate subject in the video. However, as shown in [18], these attention maps may be not stable enough. They may focus on backgrounds even in simple environments. Besides, the end-to-end trained attention maps are hungry of training data. Secondly, redundant frames are common in videos which increase the difficulty of long time human action encoding. To address these problems, this paper first proposes a kind of hand-crafted trajectory attention map to help extract convolutional visual features which only be related with subject areas in videos. It is not limited by the number of training data, camera motion and other complex environments. Then, a new kind of cluster pooling method is proposed to reduce the redundant frames of the videos. Furthermore, rank pooling method [25] is invoked to encode the dynamic of human actions to get Trajectory-weighted Deep-convolutional Rank-pooling Descriptor (TDRD). Compared with other time encoding methods, rank pooling method has advantages of encoding speed and long-time sequence description. Our TDRD contains the advantages of trajectory attention map, cluster pooling and rank pooling. So it can describe the dynamics of human actions in long-time redundant videos effectively. Finally, TDRDs are used for fall detection with a SVM. Extensive experiments have been done, and the proposed descriptor has obtained superior performance on public fall detection dataset.

The paper mainly has three contributions as follows: (1) We propose a kind of trajectory attention map to localize areas of subject. On the basis of improved dense trajectory [22], the proposed trajectory attention map is robust to the limited training data, camera motion and other complex environments. (2) We propose a cluster pooling method. This pooling method can effectively reduce the redundancy of the time sequence by clustering along time. (3) We invoke the rank pooling method [25] to propose a new descriptor TDRD. It is computed with deep ConvNets, trajectory attention map, cluster pooling and rank pooling, which can describe the dynamics of human actions in long-time redundant videos effectively.

This paper is organized as follows: Section 1 introduces the background. Section 2 presents the details of the proposed descriptor. Section 3 describes the experiments and discusses the results. Section 4 gives the conclusion.

II. TRAJECTORY-WEIGHTED DEEP-CONVOLUTIONAL RANK-POOLING DESCRIPTOR

Fig. 1 shows the workflow of proposed Trajectory-weighted Deep-convolutional Rank-pooling Descriptor (TDRD). The approach involves five main modules. First, CNN feature maps are extracted from the whole RGB video with a VGG16 [16] network and then are normalized with the spatiotemporal normalization method [15]. Second, improved dense trajec-tories [22] are computed from the RGB video to describe the motion trajectory of the subject. Trajectory attention maps are built with trajectory points to help locate human areas. Third, in each frame, CNN feature map is weighted with its corresponding trajectory attention map to get trajectory-weighted convolutional feature of human region. The trajectory-weighted convolutional feature sequence of all frames can reflect the dynamics of human action. Fourth, cluster pooling is applied on trajectory-weighted convolutional feature sequence to reduce the redundancy. Fifth, rank pooling [25] is invoked to encode the dynamic of cluster-pooled sequence to get TDRD.

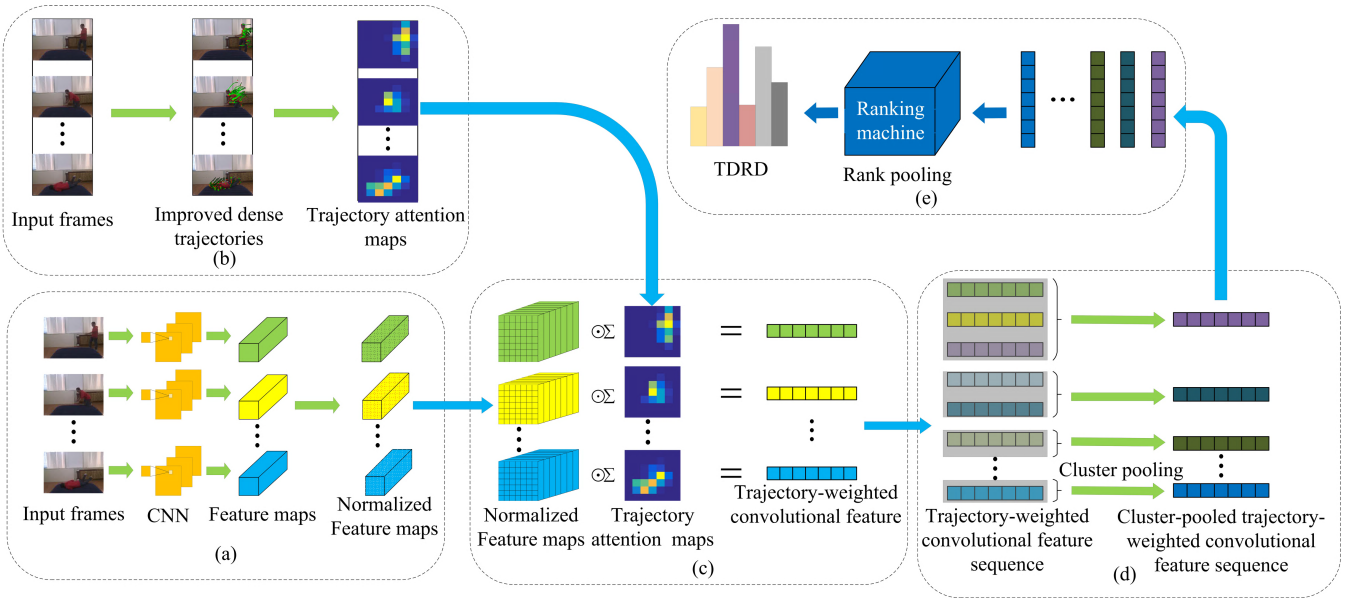

FIGURE 1. **The pipeline of trajectory-weighted deep-convolutional rank-pooling descriptor. (a) CNN feature maps are extracted from the whole RGB video and normalized by spatiotemporal normalization method. (b) The process of building trajectory attention map. (c) Normalized CNN feature maps are weighted with their corresponding trajectory attention maps to get trajectory-weighted convolutional features. (d) Cluster pooling on trajectory-weighted convolutional feature sequence (e) Rank pooling along the cluster-pooled trajectory-weighted convolutional feature sequence to get TDRD.**

In the following, we describe details of the five modules.

A. CONVOLUTIONAL NETWORKS AND CNN FEATURE MAPS

Considering that current fall detection datasets are small and simple, VGG16 [16] is enough to represent the visual feature. So we take VGG16 as the basic deep ConvNet in this paper. VGG16 has five group convolutional layers and 3 fully-connected layers. Max pooling layer follows each group convolutional layer. The resolution of the input in VGG16 is 224 × 224. we also replace two fully-connected layers with one fully-connected layer with 1024 units to reduce the size of network and weaken the effect of fully-connected layers.

Then VGG16 is taken as a generic feature extractor on each frame of input video to obtain the video CNN feature maps. Each pixel of the feature map encodes the color and edge features of its corresponding receptive field [17]. We extract CNN feature maps after the 5^{th} group convolutional layer in VGG16 because it has appropriate size of receptive field and encodes more complex features. Normalization proves to be an effective strategy in designing features, so we also normalize the CNN feature maps with spatiotemporal normalization [15]. CNN feature maps of whole video after spatiotemporal normalization are written as a 4D tensor $F \in R^{C\times T\times H\times W}$, where C, T, H, and W represent channel, video duration, height and width respectively.

B. TRAJECTORY ATTENTION MAP

Human actions may be happened in various regions of the video frames. CNN feature map contains the spatial structure of the input image, so we further propose a kind of trajectory attention map (see Fig. 2) to locate subject on CNN feature map.

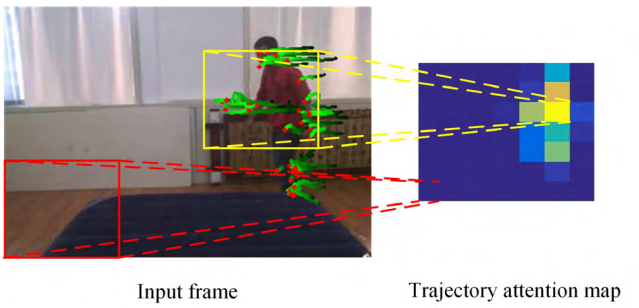

FIGURE 2. **Calculation of trajectory attention map. The red points in frame are trajectory points and the green lines represent motion direction of trajectories. Red and yellow rectangular boxes are receptive field of corresponding pixels in trajectory attention map.**

Attention map [18]–[20] is widely applied on CNN feature map to help locate subject in the video. It has the same height and width with CNN feature map. In each frame, attention map is corresponding to the CNN feature map. Existing attention maps are computed by an end-to-end network, which need a large number of training data and may not be always precisely in complex environments [18]. Our trajectory attention map is built based on dense trajectories [21] and improved dense trajectories [22]. Dense trajectories and improved dense trajectories describe motion trajectory of subject accurately even with complex surrounding environments, so trajectory points are always distributed in the subject region. We build each pixel of trajectory attention map by

counting trajectory points in corresponding receptive field, so the larger the subject region contained in the receptive field, the greater the pixel value of trajectory attention map. Thus, our trajectory attention map is the best fit to the action locality.

We first review the process of extracting dense trajectories [21]: Dense feature points are first sampled on first frame of the video by a grid partition with interval of 5 pixels. Then, eigenvalue of auto-correlation matrix of each feature point is calculated and a threshold is set to filter feature points in homogeneous areas. Finally, each feature point on first frame is regarded as a starting point of one trajectory. We track these feature points along time to get dense trajectories. Suppose that $p_l^t(l = 1, \ldots, L)$ represents one starting feature point of l^{th} trajectory in position (x, y) at t^{th} frame. the next position of tracked trajectory point p_l^{t+1} in $(t + 1)^{th}$ frame can be calculated as (1).

$$(x^{t+1}, y^{t+1}) = \left(x^t, y^t\right) + \left(M * \omega^t\right)|_{(x^t, y^t)} \tag{1}$$

$\omega^t = (u^t, v^t)$ is the dense optical flow in t^{th} frame and u^t, v^t represent horizontal and vertical components respectively. M represents the median filtering kernel which can help decide the direction of trajectory. The dense optical flow around trajectory point in t^{th} frame help determine the position of next trajectory point in $(t+1)^{th}$ frame. As in [21], to avoid trajectory drifting, we set the same max length of each trajectory 15. Finally, trajectories with no position change will be removed.

Improved dense trajectory [22] is an advanced version of dense trajectory which can overcome the problem of camera motion. It assumes that the relationship between two cons-ecutive frames can be described by a homography matrix. In order to estimate homography matrix, SURF features [23] and dense optical flow features are first extracted. In two conse-cutive frames, the extracted features are matched with each other. Then RANSAC algorithm [24] is used to estimate homography matrix according to these matched features. The l^{th} improved dense trajectory started at t^{th} frame can be written as

$$\left\{p_l^t, p_l^{t+1}, \ldots, p_l^{t+N-1}\right\} = \{\left(x^t, y^t\right), \left(x^{t+1}, y^{t+1}\right), \ldots, (x^{t+N-1}, y^{t+N-1})\}$$

N is trajectory length. All improved dense trajectory points in t^{th} frame is $P^t = (p_1^t, p_2^t, \ldots, p_L^t)$.

Based on improved dense trajectory points in each frame, we further build our trajectory attention map. We build each pixel of trajectory attention map by counting trajectory points in corresponding receptive field. Trajectory attention maps of a video can be written as $A \in R^{T \times H \times W}$, H and W represent height and width of trajectory attention map. T is the video duration. $a_{i,j}^t(t = 1, \ldots, T; i = 1, \ldots, H; j = 1, \ldots, W)$ is one pixel at position (i, j) of trajectory attention map in t^{th} frame. $a_{i,j}^t$ is computed by counting the trajectory points in spatial receptive field as (2).

$$a_{i,j}^t = N_{P^t \in \Omega_{i,j}^t}(P^t) \tag{2}$$

$\Omega_{i,j}^t$ is the spatial receptive field of $a_{i,j}^t$ in t^{th} frame. N represents statistical function which can count the number of trajectory points. Then, we normalize the trajectory attention map as (3). Finally, the trajectory attention maps can be written as $A = (\overline{a}_{ij}^t)_{T \times H \times W}$.

$$\overline{a}_{ij}^t = \frac{a_{ij}^t}{\sum_{i,j} a_{ij}^t} \tag{3}$$

C. TRAJECTORY WEIGHTED CONVOLUTIONAL FEATURE

To compute the convolutional visual feature of human region in each frame, CNN feature map is weighted with its corresponding trajectory attention map to get trajectory-weighted convolutional feature. The trajectory-weighted convolutional feature in each frame only encodes the visual feature of human region.

Trajectory attention maps of whole video are written as $A = (\overline{a}_{ij}^t)_{T \times H \times W}$. CNN feature maps of whole video are written as $F = (f_{ijc}^t)_{C \times T \times H \times W}$. In each frame, each channel of CNN feature map is weighted with the corresponding trajectory attention map. Trajectory-weighted convolutional feature in t^{th} frame is $U^t \in R^C$. It can be computed as (4).

$$U^t = \forall_{c \in C}(\sum\nolimits_{i,j} \overline{a}_{ij}^t f_{ijc}^t) \tag{4}$$

$\overline{a}_{ij}^t$ represents pixel value in position (i, j) of trajectory attention map in t^{th} frame. f_{ijc}^t represents pixel value in position (i, j, c) of CNN feature map in t^{th} frame. Trajectory-weighted convolutional features of all video frames can be written as a time sequence $U = \{U^1, U^2, \ldots U^T\}$, T is video duration. This trajectory-weighted convolutional feature sequence reflects the dynamic of human action.

D. CLUSTER POOLING

Redundant frames are common in videos, which increase the difficulty of time encoding for human actions. We propose a new kind of cluster pooling method to reduce the redundant frames in videos. Cluster pooling is applied on frame-level visual features of a video in time sequence. It clusters the feature sequence along time based on sequence change detection.

Cluster pooling method first clusters several non-covered homogeneous sequence segments. Then it computes mean value of each segment as the representation. Finally, mean values of all clustered segments are combined to be a new sequence. This new sequence reduces the redundancy of original sequence. We apply cluster pooling on trajectory-weighted convolutional feature sequence to get no-redundant cluster-pooled trajectory-weighted convolutional feature sequence (see Fig. 3). Suppose that $U = \{U^1, U^2, \ldots U^T\}$ is trajectory-weighted convolutional feature sequence. Cluster pooling first initializes 1^{th} clustered

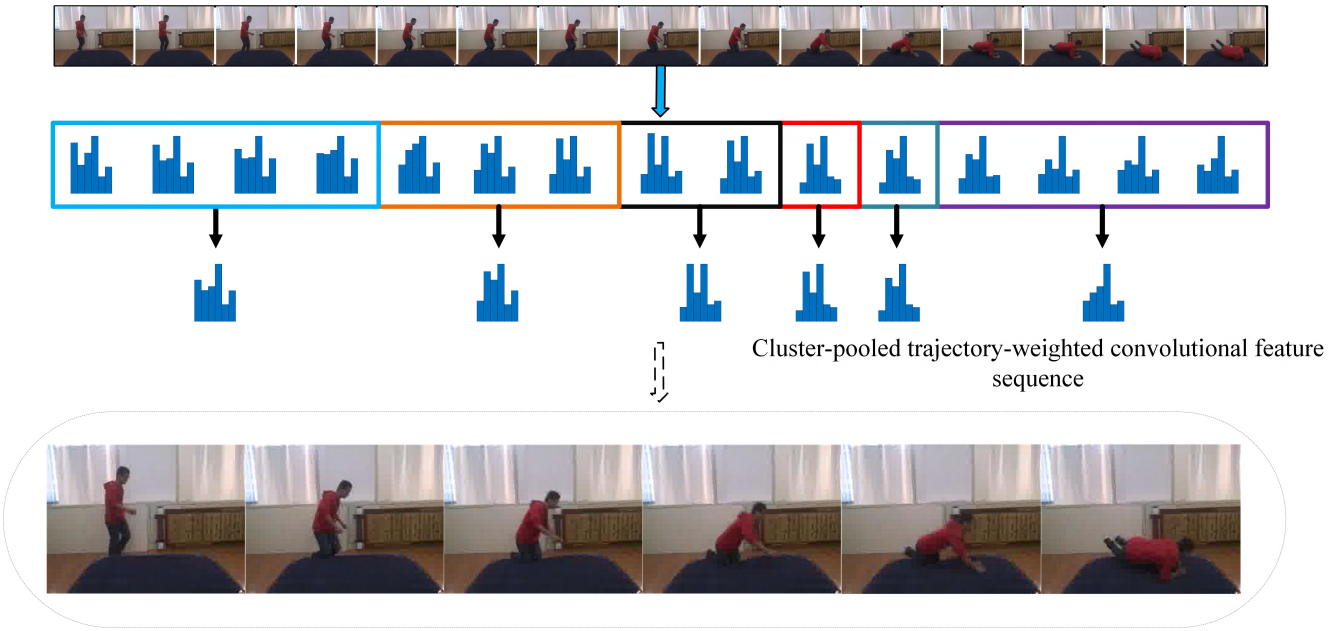

FIGURE 3. Cluster pooling on trajectory-weighted convolutional feature sequence. The color rectangular boxes represent clustered segments.

segment $\overline{U}^1 = \{U^1\}$. Its segment mean vector is $\mu^1 = M(\overline{U}^1)$. M represents mean fun- ction. Then, we traverse the next trajectory-weighted convolutional feature and compute the distance with segment mean vector as (5).

$$d = D(\mu^1, U^2) \tag{5}$$

$\boldsymbol{D}$ represents distance metric function. Next, we set a distance threshold $\boldsymbol{\sigma}$ to decide if we append $\boldsymbol{U^2}$ to current segment $\overline{\boldsymbol{U}}^{\boldsymbol{1}} = \{\overline{\boldsymbol{U}}^{\boldsymbol{1}}, \boldsymbol{U^2}\}$ or start a new segment $\overline{\boldsymbol{U}}^{\boldsymbol{2}} = \{\boldsymbol{U^2}\}$. If $\boldsymbol{d} \leq \boldsymbol{\sigma}$, we append current trajectory-weighted convolutional feature to current segment and re-compute the segment mean vector, else, we start a new segment. Finally, we traverse the trajectory-weighted convolutional feature sequence from 1 to $\boldsymbol{T}$ to get several non-covered clustered segment. Mean vectors of all clustered segments are combined to be a cluster-pooled trajectory-weighted convolutional feature sequence. The algorithm is described in TABLE 1.

E. TRAJECTORY-WEIGHTED DEEP-CONVOLUTIONAL RANK-POOLING DESCRIPTORS

Finally, we invoke rank pooling method [25] to encode the cluster-pooled trajectory-weighted convolutional fea- ture seq-uence of the video to get our Trajectory-weighted Deep-convolutional Rank-pooling Descriptor (see Fig. 4). Compar-ed with other time encoding methods, rank pooling method has advantages of encoding speed and long-time sequence describing. Our new descriptor TDRD is computed with deep ConvNets, trajectory attention map, cluster pooling and rank pooling, so it has advantages in describing the dynamic of human actions in long-time redundant videos effectively.

Rank pooling method takes a linear rank function to encode time sequence. Every element of the time sequence is a vector.

TABLE 1. The algorithm of cluster pooling.

Algorithm: Cluster Pooling

Input: Trajectory-weighted convolutional feature sequence $U = \{U^1, U^2, \ldots U^T\}$. Distance metric function D. Distance threshold σ. Mean function M. Empty output sequence $\overline{U}$.

Output: $\overline{U} = \{M(\overline{U}^1), M(\overline{U}^2), \ldots M(\overline{U}^{\overline{T}})\} = \{\mu^1, \mu^2, \ldots \mu^{\overline{T}}\}$ is new sequence after cluster pooling. $\overline{T} \leq T$.

Steps:

1. Initialize 1^{th} segment $\overline{U}^1 = \{U^1\}$. Its mean vector $\mu^1 = M(\overline{U}^1)$.
2. **for** $t = 2, \ldots, T$ **do**
3. Compute distance between current vector and current segment mean vector d = D(μ^1, U^t).
4. **if** $d \leq \sigma$ **then**
5. Append current vector to current segment $\overline{U}^1 = \{\overline{U}^1, U^t\}$.
6. Re-compute segment mean vector $\mu^1 = M(\overline{U}^1)$.
7. **else**
8. Start a new segment $\overline{U}^2 = \{U^t\}$.
9. **end if**
10. **end for**

Linear rank function takes a linear transformation on every element to project it to one value. Rank pooling restricts the projected value preserve the chronological order, i.e., the projected value of later time is larger than early one. Finally, the parameter of linear rank function is used to represent input time sequence. The cluster-pooled trajectory-weighted convo-lutional feature sequence is $\overline{U} = \{\mu^1, \mu^2, \ldots \mu^{\overline{T}}\}$. It is first smoothed as (6).

$$\begin{cases} m^i = \frac{1}{i} \sum_{\tau=1}^{i} \mu^\tau, \quad i = 1, 2, \ldots \overline{T} \\ v^i = \frac{m^i}{\| m^i \|} \end{cases} \tag{6}$$

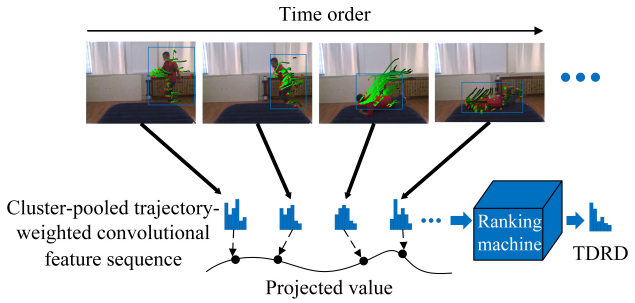

FIGURE 4. Rank pooling method on cluster-pooled trajectory-weighted convolutional feature sequence.

$V = \{v^1, v^2, \ldots v^{\overline{T}}\}$ is smoothed sequence, $v^t \in R^C$. Linear rank function is written as φ. So the linear transformation can be written as $\varphi(v; d) = d^T.v$, where $d \in R^C$ represents parameter of linear rank function. After linear transformation, the projected value still maintain its time order, that is,when $t_i > t_j$, it constrain $\varphi\left(v^{t_i}; d\right) > \varphi\left(v^{t_j}; d\right)$. The objective function of rank pooling can be written as (7).

$$\begin{cases} min\dfrac{1}{2}||d||^2 + C\sum_{\forall t_i > t_j} \varepsilon_{ij} \\ s.t.d^T\left(v^{t_i} - v^{t_j}\right) \geq \delta - \varepsilon_{ij} \\ \varepsilon_{ij} \geq 0 \end{cases} \tag{7}$$

$\frac{1}{2}||d||^2$ represents penalty term and ε_{ij} represents slack variable. δ is one positive number and its default value is 1 in [25], d is the parameter of linear rank function. We compute optimal d^* as our TDRD.

III. EXPERIMENT

In this section, we first describe the datasets that we use for human fall detection. Next, some details of our experiment will be introduced. Then, we do some exploratory experiments according to the contributions in this paper. We illustrate some trajectory attention maps, investigate threshold of cluster pooling, and evaluate the effect our trajectory attention map. Finally, we evaluate the performance of our TDRD and compare with other methods on SDUFall dataset, UR dataset and Multiple cameras dataset.

A. DATASET

1) SDUFall DATASET [7]

SDUFall dataset is collected from a Kinect camera with 1.5 meters high. It contains six actions: falling down, bending, squatting, sitting, lying and walking. 20 young volunteers do each action 10 times in different conditions, and there are total 1200 RGB videos and 1200 depth videos. What's more, it contains rich surrounding environment such as illumination variation, direction change, and position change.

2) UR FALL DETECTION DATASET (URFD) [26]

UR dataset contains 30 kinds of fall actions which are recorded with two Microsoft Kinect cameras from different fixed view. We take all videos of different view, so there are total 60 fall image sequences. Besides, 40 daily living activity sequences are also recorded with only one device.

3) MULTIPLE CAMERAS FALL DATASET [35]

Multiple cameras fall dataset contains 24 scenarios which are recorded synchronously with 8 cameras from different fixed view. In each scenario, it contains simulated falls and normal daily activities such as lying, crouching, sitting on a sofa, walking, and so on.

B. IMPLEMENTATION DETAILS

In our experiment, we choose support vector machine (SVM) as our classifier. we choose SDUFall dataset, UR dataset and Multiple cameras dataset to test our Trajectory-weighted Deep-convolutional Rank-pooling Descriptor (TDRD). On SDUFall dataset, 3/5 of dataset is randomly sampled as training set and the rest 2/5 is regarded as test set in each class. Considering that it only has video label, so we sample 10 frames with interval of 2 start from video end in the training set to train CNN network. The video length of the SDUFall dataset is 120-250 and we compute our TDRD along the whole video. On UR dataset, the video length is 100-150 and we also compute TDRD along the whole video. We take 5-folds cross validations. Every frame has its label so we choose all frames of the video in training set to train CNN network. On Multiple cameras dataset, every long-term video contains fall and non-fall actions, so we clip 184 fall video segments and 216 no-fall video segments. The no-fall videos contain 80 walking, 40 crouch down, 48 get seated and 48 others (lying, moving up) video segments. The average video length is 50 frames. We also take 5-folds cross validations. Every frame has its label so we choose all frames of the video segments in training set to train CNN network. We also compute TDRD along the whole video segments of Multiple cameras dataset. On UR and Multiple cameras datasets, we perform fall-versus-nonfall binary classification and evaluate the sensitivity and specificity. On SDUFall dataset, we evaluate average classification accuracy for six actions. Sensitivity and specificity are computed as (8) and (9):

$$\text{Sensitivity} = \frac{TP}{TP + FN} \tag{8}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{9}$$

In training stage, we fine-tune the network pre-trained on ImageNet [27] with stochastic gradient descent (SGD) algorithm. The learning rate is initialized to 0. 001, which will multiply 0.1 if the test loss not reduce. The momentum, weight decay, and batch size are set to 0.9, 0.0001, and 50. In order to prevent overfitting, we take drop out with ratio of 0.5 after fully-connected layer. Besides, data augment methods such as horizontal flip are also used. We extract the feature map after 5^{th} group convolutional layer. The spatial size of the CNN feature map is 7×7.

C. EXPLORATION EXPERIMENTS

1) ILLUSTRATION OF TRAJECTORY ATTENTION MAP

We first visualize some trajectory attention maps on SDUFall dataset (shown in Fig. 5). From Fig. 5 we can see that in trajectory attention maps, values of pixels are mainly distributed around human areas. The pixels distributed in surrounding environments are almost zero. So our trajectory attention map can locate the position of human accurately in different environments.

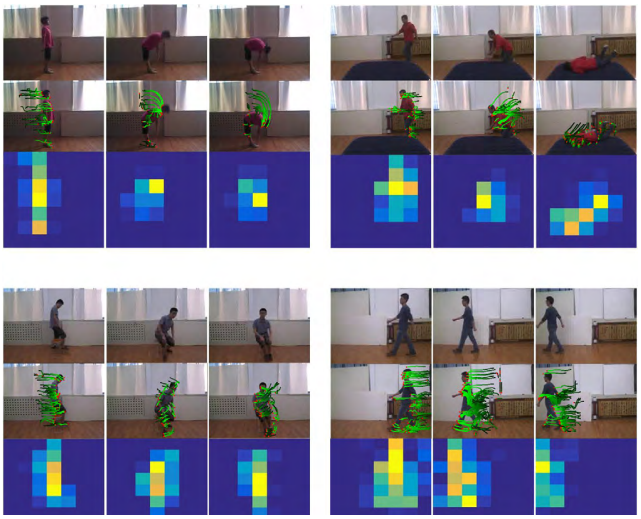

FIGURE 5. Illustration of trajectory attention maps. In each sub-figure, the top rows are raw video frames, the middle rows are improved dense trajectories and the bottom rows are trajectory-weighted attention maps.

2) THRESHOLD OF CLUSTER POOLING

Threshold is an important parameter in cluster pooling. It decides the length and amount of cluster segments. We first count the average ratio of reduced length of the sequence with the change of threshold on SDUFall dataset. Then, we compute the average classification accuracy for six actions on SDUFall dataset with threshold varying from 0 to 1 to choose the best threshold. Two results are shown in Fig. 6. From Fig. 6(a) we can see that with the increase of threshold, the cluster-pooled sequence is getting shorter and shorter. So our cluster pooling method can reduce the redundancy of sequence effectively. Fig. 6(b) shows that 0.7 achieves the best average accuracy 96.04% on SDUFall dataset, so we fix the threshold as 0.7 in our experiments.

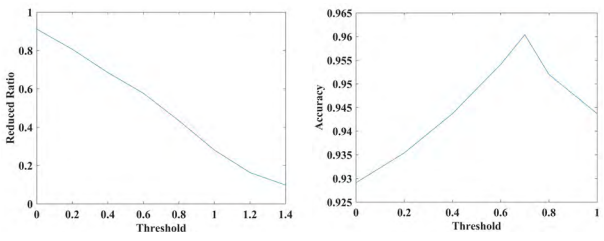

FIGURE 6. Exploration of different threshold in cluster pooling on SDUFall dataset. Left: The average reduced ratio of length of sequence. Right: Performance trend with threshold varying.

3) EVALUATION OF TRAJECTORY ATTENTION MAP

To further evaluate the effect of our trajectory attention map, we remove the trajectory attention map and replace trajectory- weighted convolutional feature with other three kinds of frame-level features. The first one takes spatial global max pooling on CNN feature map in each frame. The second one takes spatial global average pooling on CNN feature map in each frame. The third frame-level feature replace trajectory-weighted convolutional feature with fully-connected feature directly. These three frame-level features not consider the position of human in videos, of course, they also can not encode the visual feature of human region as our trajectory-weighted convolutional feature. Then the three frame-level features are encoded as three descriptors by cluster pooling and rank pooling method. We call these three contrastive descriptors as Max-pooling Deep-convolutional Rank-pooling Descriptor (MDRD), Average-pooling Deep-convolutional Rank-pooling Descriptor (ADRD), and Fully-connected Deep-convolutional Rank-pooling descriptor (FDRD). The experimental results of average classification accuracy for six actions on SDUFall dataset are shown in TABLE 2. From TABLE 2 we can see that our TDRD gets the best results of 96.04% and it is 4%, 2.1%, and 1.5% higher than MDRD, ADRD, and FDRD. We think the main reason is that human action is a key point for human fall detection, but simple max pooling, average pooling and fully connection frame-level features not consider the position of human. So they can not encode the visual feature of human region. However, our trajectory attention map can locate human position and can help compute the trajectory-weighted convolutional feature at human areas, so our trajectory attention map is more effective.

TABLE 2. Accuracy.

	TDRD	MDRD	ADRD	FDRD
accuracy	0.960	0.920	0.939	0.945

Note:
TDRD: Trajectory-weighted Deep-convolutional Rank-pooling Descriptor.
MDRD: Max-pooling Deep-convolutional Rank-pooling Descriptor.
ADRD: Average-pooling Deep-convolutional Rank-pooling Descriptor.
FDRD: Fully-connected Deep-convolutional Rank-pooling descriptor.

D. COMPARISON WITH OTHER EXISTING METHODS

We first illustrate the confusion matrix of TDRD on SDUFall dataset, as shown in Fig. 7, from which we can see that our TDRD get superior performance for all six actions. Then, we compare our method with other existing methods on SDUFall dataset (shown in Table 3), UR dataset (shown in Table 4) Multiple cameras dataset (shown in TABLE 5). From Table 3 we can see that our TDRD gets the best result

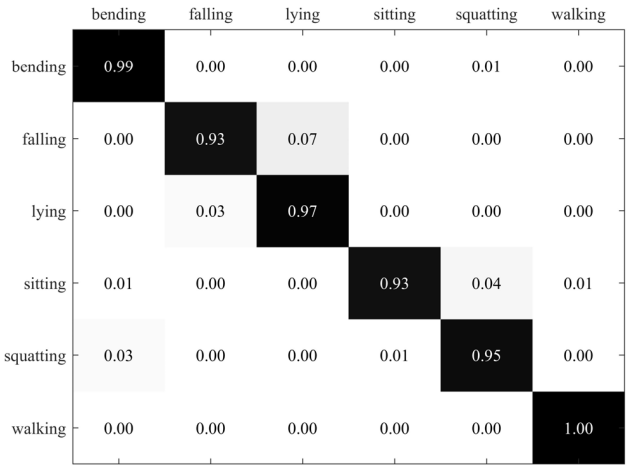

FIGURE 7. Confusion matrix of TDRD on SDUFall dataset.

TABLE 3. Comparison between our TDRD with other methods on SDUFALL dataset.

Method	Video type	Accuracy	Sensitivity	Specificity
Ma, Xin [7]	depth		77.14%	91.15%
Erdem [29]	depth	89.63%		
Merrouche[30]	depth		90.76%	93.52%
N. Jacob [28]	depth	95%(ROC, SDU-Filled)		
Fan[31]	RGB	81.33%		
TDRD	RGB	**96.04%**	**93.01%**	**99.50%**

of average accuracy for six actions, sensitivity, and specificity on both depth-based methods and RGB-based methods. We get 96.04% average classification accuracy, which is about 14.71% higher than the best RGB-based method [31] and 1.04% higher than the best depth-based method [28]. We also get superior sensitivity and specificity. The specificity is almost equal to 100%.

In Table 4, we can see that our TDRD gets 100% sensitivity and 95.00% specificity on UR dataset. UR dataset only has total 100 videos. The results show that our TDRD can get comparable performance on very small dataset. In Table 5, our TDRD gets 90.21% sensitivity and 92.59% specificity

TABLE 4. Comparison between our TDRD with other methods on UR dataset.

Method	Video type	Sensitivity	Specificity
Kwolek[26]	Depth+Accel	100%	96.67%
Yixiao [33]	RGB	100%	97.25%
Yixiao [32]	RGB	100%	**100%**
N. Lu [12]	RGB	99.27% (acc)	
TDRD	RGB	**100%**	95.00%

TABLE 5. Comparison between our TDRD with other methods on Multiple cameras dataset.

Method	Video type	Accuracy	Sensitivity	Specificity
Yixiao [32]	Arbitrary view		100%	100%
K. Fan [31]	Arbitrary view	98.25%		
Y. Fan [13]	Arbitrary view		97.1%	97.9%
N. Lu [12]	Arbitrary view	99.73%	96.65%	99.85%
C. Ge [36]	Arbitrary view		89.40%	93.23%
TDRD	Arbitrary view		90.21%	92.59%

on Multiple cameras dataset. Through experimental analysis, we found that the main reason why our TDRD performs not very well is that the improved dense trajectory features extracted in lying action are very poor. Lying contains a lot of static postures, so our TDRD is lack of skills of describing these long-time static postures. We will further improve our algorithm to overcome this shortcoming in the future.

E. TIME COST ANALYSIS

We evaluate the training time and runtime of the proposed method. TDRD extraction is implemented in c++ and python while SVM is implemented in MATLAB. We conduct experiments on a PC with Intel (R) i5-4590 CPU, 16 GB RAM and a single Nvidia 1080ti GPU. The CNN network training duration on SDUFall dataset, UR dataset and Multiple cameras da- taset are respectively about 65, 30 and 35 minutes. The TDRD feature building runs at about 2 fps. The main reason is that when extracting improved dense trajectory, we directly use the program provided in [22]. This program additionally computes other features besides the improved dense trajectory. So it is very necessary to rewrite the program with c/c++ if our system is conducted on real-time videos.

IV. CONCLUSION

In this paper, we first present a kind of attention map named trajectory attention map. On the basis of improved dense trajectory, the proposed trajectory attention map can help locate the subject and is robust to the complex environment. Then, we further propose a kind of video descriptor named Trajectory-weighted Deep-convolutional Rank-pooling Desc-riptor through fusing trajectory attention maps, CNN feature maps, cluster pooling and rank pooling. Our descriptor has advantages in describing the dynamics of human actions in long-time redundant videos effectively. Our TDRD gets the superior performance on SDUFall dataset and gets comparable performance on UR dataset and Multiple cameras dataset. Our algorithm achieves great results in single fall detection, but in multiplayer scenes, it fails to detect human fall actions effectively. Besides, our TDRD is also lack of skills of describing long-time static postures. We will continue to improve our algorithm to overcome these shortcomings in the future.

There are more and more human fall detection algorithms based on deep learning. But the fall detection dataset, which is enough large and has complex scenes or camera motion in real applications and home service robots, is still not arise. we will seek to collaborate with Internet and our mobile robot in the future. Moreover, we may also try to change our method to be an end-to-end trained network.

REFERENCES

[1] *WHO Global Report on Falls Prevention in Older Age*, World Health Org., Geneva, Switzerland, 2008.

[2] *Report of the Six National Population Census*, Nat. Bureau Statist. China, Beijing, China, 2011.

[3] C. Griffiths, C. Rooney, and A. Brock, "Leading causes of death in England and Wales—How should we group causes," *Health Statist. Quart.*, vol. 28, no. 9, pp. 1–12, 2005.

[4] Z. Zhang, C. Conly, and V. Athitsos, "A survey on vision-based fall detection," in *Proc. Int. Conf. Pervasive Technol. Rel. Assistive Environ.*, 2015, pp. 1–7.

[5] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Robust video surveillance for fall detection based on human shape deformation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 5, pp. 611–622, May 2011.

[6] V. Vishwakarma, C. Mandal, and S. Suralm, "Automatic detection of human fall in video," in *Proc. 2nd Int. Conf. Pattern Recognit. Mach. Intell. (PReMI)*, Kolkata, India, 2007, pp. 616–623.

[7] X. Ma, H. Wang, B. Xue, M. Zhou, B. Ji, and Y. Li, "Depth-based human fall detection via shape features and improved extreme learning machine," *IEEE J. Biomed. Health Inform.*, vol. 18, no. 6, pp. 1915–1922, Nov. 2014.

[8] H. Nait-Charif and S. J. McKenna, "Activity summarisation and fall detection in a supportive home environment," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR)*, vol. 4, Aug. 2004, pp. 323–326.

[9] N. Thome, S. Miguet, and S. Ambellouis, "A real-time, multiview fall detection system: A LHMM-based approach," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 18, no. 11, pp. 1522–1532, Nov. 2008.

[10] D. Anderson, R. H. Luke, J. M. Keller, M. Skubic, M. Rantz, and M. Aud, "Linguistic summarization of video for fall detection using voxel person and fuzzy logic," *Comput. Vis. Image Understand.*, vol. 113, no. 1, pp. 80–89, 2008.

[11] W. Min, H. Cui, H. Rao, Z. Li, and L. Yao, "Detection of human falls on furniture using scene analysis based on deep learning and activity characteristics," *IEEE Access*, vol. 6, pp. 9324–9335, Jan. 2018.

[12] N. Lu, Y. Wu, L. Feng, and J. Song, "Deep learning for fall detection: 3D-CNN Combined with LSTM on video kinematic data," *IEEE J. Biomed. Health Inform.*, to be published, doi: 10.1109/JBHI.2018.2808281.

[13] Y. Fan, M. D. Levine, G. Wen, and S. Qiu, "A deep neural network for real-time detection of falling humans in naturally occurring scenes," *Neurocomputing*, vol. 260, pp. 43–58, Oct. 2017.

[14] Y. Z. Hsieh and Y. L. Jeng, "Development of home intelligent fall detection IoT system based on feedback optical flow convolutional neural network," *IEEE Access*, vol. 6, pp. 6048–6057, 2018.

[15] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 4305–4314.

[16] S. Karen and A. Zisserman. (2014). "Very deep convolutional networks for large-scale image recognition." [Online]. Available: https://arxiv.org/abs/1409.1556

[17] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, Zurich, Switzerland, 2014, pp. 818–833.

[18] S. Shikhar, R. Kiros, and R. Salakhutdinov. (2015). "Action recognition using visual attention." [Online]. Available: https://arxiv.org/abs/1511.04119

[19] R. Girdhar and D. Ramanan, "Attentional pooling for action recognition," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 33–45.

[20] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. M. Snoek, "VideoLSTM convolves, attends and flows for action recognition," *Comput. Vis. Image Understand.*, vol. 166, pp. 41–50, Jan. 2018.

[21] H. Wang, A. Kläser, C. Schmid, and C. L. Liu, "Action recognition by dense trajectories," in *Proc. CVPR*, Providence, RI, USA, Jun. 2011, pp. 3169–3176.

[22] H. Wang and C. Schmid, "Action recognition with improved trajectories," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sydney, NSW, Australia, 2013, pp. 3551–3558.

[23] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proc. Eur. Conf. Comput. Vis.* Berlin, Germany, 2006, pp. 404–417.

[24] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[25] B. Fernando, E. Gavves, J. Oramas M., A. Ghodrati, and T. Tuytelaars, "Rank pooling for action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 773–787, Apr. 2017.

[26] B. Kwolek and M. Kepski, "Human fall detection on embedded platform using depth maps and wireless accelerometer," *Comput. Methods Programs Biomed.*, vol. 117, no. 3, pp. 489–501, Dec. 2014.

[27] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2012, pp. 1097–1105.

[28] N. Jacob, S. S. Khan, and A. Mihailidis. (2018). "DeepFall–non-invasive fall detection with deep spatio-temporal convolutional autoencoders." [Online]. Available: https://arxiv.org/abs/1809.00977

[29] E. Akagündüz, M. Aslan, A. Şengür, H. Wang, and M. C. İnce, "Silhouette orientation volumes for efficient fall detection in depth videos," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 3, pp. 756–763, May 2017.

[30] F. Merrouche and N. Baha, "Depth camera based fall detection using human shape and movement," in *Proc. IEEE Int. Conf. Signal Image Process. (ICSIP)*, Beijing, China, Aug. 2016, pp. 586–590.

[31] K. Fan, P. Wang, and S. Zhuang, "Human fall detection using slow feature analysis," *Multimedia Tools Appl.*, vol. 2, pp. 1–28, Jan. 2018.

[32] Y. Yun and I. Y.-H. Gu, "Human fall detection in videos by fusing statistical features of shape and motion dynamics on Riemannian manifolds," *Neurocomputing*, vol. 207, pp. 726–734, Sep. 2016.

[33] Y. Yun and I. Y.-H. Gu, "Human fall detection in videos via boosting and fusing statistical features of appearance, shape and motion dynamics on Riemannian manifolds with applications to assisted living," *Comput. Vis. Image Understand.*, vol. 148, pp. 111–122, Jul. 2016.

[34] L. Sun, K. Jia, K. Chen, D. Y. Yeung, B. E. Shi, and S. Savarese, "Lattice long short-term memory for human action recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2166–2175.

[35] A. Edouard, C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau, "Multiple cameras fall data set," DIRO-Univ. Montréal, Montréal, QC, Canada Tech. Rep. 1350, 2010.

[36] C. Ge, I. Y.-H. Gu, and J. Yang, "Human fall detection using segment-level CNN features and sparse dictionary learning," in *Proc. IEEE 27th Int. Workshop Mach. Learn. Signal Process. (MLSP)*, Tokyo, Japan, Sep. 2017, pp. 1–6.

ZHIMENG ZHANG received the B.E. degree in automation from the Shandong University of Technology, Shandong, China, in 2016. He is currently pursuing the master's degree with the School of Control Science and Engineering, Shandong University, Shandong. His current research interests include action recognition, computer vision, and deep learning.

XIN MA received the B.S. degree in industrial automation and the M.S. degree in automation from Shandong University, Shandong, China, in 1991 and 1994, respectively, and the Ph.D. degree in aircraft control, guidance, and simulation from the Nanjing University of Aeronautics and Astronautics, Nanjing, China, in 1998. She is currently a Professor with Shandong University. Her current research interests include artificial intelligence, machine vision, human–robot interaction, and mobile robots.

HANBO WU received the B.S. degree in automation and the M.S. degree in control engineering from Shandong University, Shandong, China, in 2014 and 2017, respectively, where she is currently pursuing the Ph.D. degree with the School of Control Science and Engineering. Her current research interests include computer vision, deep learning, and action recognition.

YIBIN LI received the B.S. degree in automation from Tianjin University, Tianjin, China, in 1982, the M.S. degree in electrical automation from the Shandong University of Science and Technology, Shandong, China, in 1990, and the Ph.D. degree in automation from Tianjin University, China, in 2008. From 1982 to 2003, he was with the Shandong University of Science and Technology. Since 2003, he has been the Director of the Center for Robotics, Shandong University. His research interests include robotics, intelligent control theories, and computer control system.