Temporal action detection based on two-stream You Only Look Once network for elderly care service robot

Ke Wang, Xuejing Li, Jianhua Yang, Jun Wu and Ruifeng Li

Abstract
Human action segmentation and recognition from the continuous untrimmed sensor data stream is a challenging issue known as temporal action detection. This article provides a two-stream You Only Look Once-based network method, which fuses video and skeleton streams captured by a Kinect sensor, and our data encoding method is used to turn the spatiotemporal temporal action detection into a one-dimensional object detection problem in constantly augmented feature space. The proposed approach extracts spatial–temporal three-dimensional convolutional neural network features from video stream and view-invariant features from skeleton stream, respectively. Furthermore, these two streams are encoded into three-dimensional feature spaces, which are represented as red, green, and blue images for subsequent network input. We proposed the two-stream You Only Look Once-based networks which are capable of fusing video and skeleton information by using the processing pipeline to provide two fusion strategies, boxes-fusion or layers-fusion. We test the temporal action detection performance of two-stream You Only Look Once network based on our data set High-Speed Interplanetary Tug/Cocoon Vehicles-v1, which contains seven activities in the home environment and achieve a particularly high mean average precision. We also test our model on the public data set PKU-MMD that contains 51 activities, and our method also has a good performance on this data set. To prove that our method can work efficiently on robots, we transplanted it to the robotic platform and an online fall down detection experiment.

Keywords
Temporal action detection, skeleton, video, two-stream, Kinect

Date received: 16 April 2021; accepted: 23 July 2021

Introduction
In modern society, more and more people spend their elderly life living alone. These phenomena may cause a potential risk due to the inadequate ability of the elder people to take good care of themselves. That is also the reason why the supervision of the elderly has become a concerning issue. The United States Centers for Disease Control and Prevention provides a research that has shown that the most likely to cause injury or even death to people over 65 years of age is falling. The study showed that elderly people who get help in time after a fall can effectively reduce the risk of death by 80% and the risk of hospitalization for long-term treatment by 26%. To cope with these problems in a better way, an intelligent

State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin, Heilongjiang, China

Corresponding author:
Ke Wang, State Key Laboratory of Robotics and System, Harbin Institute of Technology, Harbin, Heilongjiang 150001, China. Email: wangke@hit.edu.cn
A service robot that can cooperate with people in the family environment is needed. With the help of a reliable algorithm, intelligent service robots can provide better performance in the family environment, which means research that focuses on human action detection becomes more and more crucial.

Human action detection has been paid more and more attention since its massive potential applications in video analysis, surveillance, and other areas. Generally, without automatic action fragment extraction, human action recognition, as a special case of temporal action detection (TAD), requires a well-trimmed video that contains only target action. TAD, however, must determine both time intervals and types of human actions from possibly untrimmed long video sequence. The spatial–temporal properties embedded in TAD provide an additional clue for possible reliable recognition but far more complex than image classification problems. Substantial progress has been made in TAD in the recent years, most of them focused on the field of video analysis, and the mean average precision (mAP) of the state-of-the-art TAD approach, to our knowledge, is approximate to 60% achieved on video database THUMOS.

Recently, researchers introduced additional sensor information for TAD. For example, skeleton data from Kinect sensor that is assumed to be more robust to the illumination change and easier to distinguish from cluttered background compared with some video approaches. Li et al. proposed a skeleton-based approach for TAD, which demonstrates some promising results on gaming three-dimensional (3D) data set. Currently, video- and skeleton-based TAD methods are rarely addressed. However, it is reasonable to extend TAD research by fusing these two kinds of information since Kinect naturally outputs both video and skeleton simultaneously.

In this article, as shown in Figure 1, we propose a TAD framework with two streams, video-based stream and skeleton-based stream. In skeleton-based, the skeleton data were converted into a color map whose width indicates the time line and height is equal to the number of skeletons. With a color map, the problem is transformed into a one-dimensional (1D) object detection problem. We propose an innovative structure with two separate recognition streams, video-stream and skeleton-stream, which are combined by boxes-fusion or layers-fusion manipulation subsequently. Two-stream network will fully utilize the features we get from both video sequence and skeleton sequence. Besides, it can also solve the problem of missing frames effectively.

As for video-stream, 3D-convolutional neural network (CNN) based on motion feature (C3D) extractor is used to extract continuous short-term motion features, which are subsequently encoded into the feature space and was represented as an augmented color map. In skeleton-stream, we apply view-invariant transform on skeleton sequence to extract view-invariant feature (VIF), which may eliminate view change influence and retain more relative motions than some traditional skeleton-based methods. The skeleton VIF sequence is also encoded into color map whose width indicates the time line and height is equal to the number of skeletons, preserving the temporal information and the spatial information of original action. With a color map, we find that it can be easily solved by an object detection framework. Inspired by this idea, we chose the You Only Look Once Version 2 (YOLOv2) detector that well balances the speed and precision. To be suitable for our work, we modify it as a 1D object detection model that only predicts two one center point coordination and the
box’s width for a bounding box. YOLO-based network that provides two information fusion strategies, boxes-fusion and layers-fusion, for TAD of color maps encoded from two input streams. To validate the proposed TAD method, we also build a seven-category data set with 200 human videos and skeleton action sequences collected by Kinect V2 sensor, each sequence contains 3–10 effective actions. The main contributions of this work are summarized as follows. We propose a novel idea that innovatively transforms the TAD problem into a 1D object detection problem based on a two-stream YOLO-based deep learning network that fuses the encoded video and skeleton stream of Kinect sensor. We also investigate the performance of two different fusion methods, boxes-fusion and layers-fusion operating at a different stage of our processing pipeline.

Another highlight lies in the stream processing for video and skeleton data. The resulting red, green, and blue (RGB) feature spaces encoded from C3D features and VIFs not only capture the underlying spatiotemporal structure of action sequence data but also allow a unified representation as to the input of the 1D YOLO network, which accordingly makes it easy to design and reduce the network complexity.

The remainder of this article is organized as follows. In the second section, we briefly review some works about our issue. The third section presents the feature extraction and encoding methods of video and skeleton data. The fourth section provides our two-stream YOLO-based network. We demonstrate some experimental results in the fifth section.

**Related works**

According to the research for Chinese elder people, people’s demand for the performance of elderly care service robots is mainly focused on intelligence and safety. Specific to the realization of functions, fall warning, sanitation, and health monitoring are the most demanded by people. This shows that researches on robot visual system are necessary.

Recently, researches on action recognition had accumulated amounts of achievements in model constructing, temporal information analysis, and high-level semantic recognition. To achieve the robot’s capability to understand and interpret human behavior during human–robot cohabitation, a lot of attempts had been made. Kostavelis et al. proposed a method named the Interaction Unit analysis, which modeled human behaviors based on a Dynamic Bayesian Network, and tried to tackle the comprehensive representation of the high-level structure of human’s behavior for the robot’s low-level sensory input. Ramirez-Amaro et al. presented semantic-based methods for understanding human movements in robotic applications. They made a segmentation of human activities and tried to identify the important features of actions. They also learned different parameters between activities and obtain the mapping between the continuous data and the symbolic and semantic interpretations of the human movements. With the development of computer vision, deep neural networks have performed well in the field of action recognition. Deep learning provides new solutions for the robot to understand and detect human actions.

**Action recognition**

Action recognition is now widely applied in various fields and an impressive progress has been made. Early action recognition methods were mainly based on hand-crafted visual features. Although CNNs can deal with most image classification problems, it may not adapt to classify human action since human action is a 3D spatiotemporal signal sequence. Yeung et al. proposed a two-stream CNNs architecture that learns the spatial motion characteristics of actions through static images and the temporal motion characteristics of actions through inter-frame optical flow. Wang et al. proposed a global spatial–temporal three-stream CNNs architecture. The three-stream CNNs comprise of spatial, local temporal, and global temporal streams generated, respectively, from deep learning single frame, optical flow, and global accumulated motion features in the form of a new formulation named motion stacked difference image. Ji et al. proposed 3D-CNNS to extract the temporal and spatial features of video data. Tran et al. also proposed a simple but effective approach for spatiotemporal feature learning using deep 3D-CNN later. Compared with CNNs, recurrent neural networks (RNNs) can store action temporal correlation information and process time-domain signals better. However, long-term action may not be well handled due to exponential decay of the gradient in time direction. To solve this issue, RNNs with long- and short-term memory (LSTM) were proposed. For example, Donahue et al. put forward an effective unified model that combines LSTM and CNNs to describe human movements.

Besides video, dynamics of human body skeleton also convey significant information. Liu et al. proposed a skeleton visualization method that used CNNs to extract human motion discrimination features. Actually, the joints of human skeleton constitute a graph structure, after graph convolutional networks (GCNs) were developed and spatial–temporal GCNs based on GCN were generalized to human skeleton graphs. Li et al. proposed a CNN-based detection framework and innovatively designed a skeleton transformer that can convert the skeleton sequence into a skeleton image as the input of the network. Liu et al. proposed a novel method to map skeletal data to images that are effectively processed by CNN architectures. The skeletal data were arranged in a two-dimensional (2D) grid to take full advantage of the 2D kernels in CNNs. They also use different colors to represent different joints; this makes it clear to notice the movement of the joints. The joint differences between the consecutive skeleton frames are used to construct the velocity images.
and append them to the location images. All these methods give a strong proof on skeleton-based action detection.

**Temporal action detection**

Existing human action recognition methods usually test their recognition algorithms on some human action database. Generally, these databases all consist of well-trimmed video that only contains one kind of target action, which is difficult to achieve in the real application. Unlike action recognition, one aim of TAD is to locate the action fragments from the input sequence accurately. Guidon et al. proposed the TAD method without over-segmentation of video to locate some simple actions. Some researchers also developed several large-scale video data sets, such as THUMOS. The typical approach used in most systems is to extract a pool of features, which are fed to support vector machines and then apply these classifiers on sliding windows or segment proposals for action prediction.

Recently, deep learning-based methods present some improved performance, and RNN becomes widely used in TAD. Escorcia et al. built a temporal action proposal system based on LSTM. Sun et al. leveraged web images to train the LSTM model when only video-level annotations are available. Such sequential prediction is often time-consuming when processing long-time videos and it does not support joint training of the underlying features extracted. Shou et al. proposed an end-to-end segment-based 3D-CNN framework (S-CNN), which outperformed other RNN-based methods by capturing spatial–temporal information simultaneously. However, the feature adopted by S-CNN is initially designed for snippet-wise action classification and might require an undesired large temporal kernel stride when detecting long action proposals.

**Object detection**

Object detection aims to classify objects and to estimate their locations in images. It has two categories: two-stage and one-stage. For two-stage detectors, the detectors are usually be divided into region proposal network (RPN) and detection head. On the other hand, one-stage detectors usually use a single network to detect objects and can be trained by end-to-end. The former is often more accurate but with low speed, whereas the latter is faster and slightly less accurate. Here are some typical object-detection networks, for example, R-CNN-based architecture received attention and an effective method has been proposed. Compared with R-CNN, fast R-CNN introduces region of interest pooling and integrates all models into one network. Later, faster R-CNN was proposed with a RPN that shares full-image convolutional features with the detection network, which makes nearly cost-free region proposals possible. With the academic’s pursuit of faster and more accurate object detectors, one-stage detector has received more attention. As a regression problem, YOLO-based object detection method spatially separates bounding boxes and associated class probabilities, which enables real-time object detection YOLO9000. This system is improved by model YOLOv2, which can predict multiple box positions and categories at one time, offering a trade-off between speed and accuracy. YOLOv2 does not choose the sliding window or extracts proposals to train the network but selects the whole image to train the model. Therefore, it is better to distinguish between the target and the background area. Other state-of-the-art object detection methods such as YOLOv3, mask R-CNN, and SSD also improve the detection progress in some aspects.

**Discussions**

Detecting target action in an untrimmed video is still a thorny problem, majority of existing methods are focused on the field of video analysis, and the best mAp, to our knowledge, is approximate 60% on video database THUMOS. In practice, especially some robotic applications such as human–robot interaction, video-based TAD lacks robustness due to changes of environment illumination. Most action recognition methods mentioned above are usually based on only images or skeleton and need a well-trimmed data sequence, which means the target action must be localized manually or predefined accordingly. Although skeleton-based and RGB video-based methods have achieved many progresses on action recognition research, TAD is still a new temporal–spatial issue arising from action recognition field. As far as we know, there are only a few reports on the video analysis, but the works based on the fusion of multisource, especially of video and skeleton from Kinect sensor, are much less reported. Therefore, a more efficient and accurate method that can perform well both on video and on skeleton is researchable. To address this problem in a unified way, it is necessary to transform different transient sensor information into augmented RGB-like images that capture invariant context or slow change properties in original data. Actually, TAD focuses on the detection and recognition of special action segments from continuous temporal action sequence. Compared with other complicated spatiotemporal modeling methods, if action sequence is transformed into an augmented image, each action segment may form like an object arranged in time order, and it is much more easier to detect the slice of action data in RGB-like images through a YOLO-based object detection framework. This one-stage model with good balance of speed and precision is much more suitable for portability and deployment on robot platform than some big networks such as YOLOv3.

**Data encoding**

In this section, to ensure the consistency of the proposed networks architecture for two different data channels, we encode two types of data. As for video data, we designed a
Table 1. Structure of our C3D network.

| Layer (type)     | Filters | Size            | Output          |
|------------------|---------|-----------------|-----------------|
| Convolutional3D  | 64      | $3 \times 3 \times 3$ | $16 \times 112 \times 112$ |
| Max_pooling3D    | $1 \times 2 \times 2$ | $16 \times 56 \times 56$ |
| Convolutional3D  | 128     | $3 \times 3 \times 3$ | $16 \times 56 \times 56$ |
| Max_pooling3D    | $2 \times 2 \times 2$ | $8 \times 28 \times 28$ |
| Convolutional3D  | 128     | $3 \times 3 \times 3$ | $8 \times 28 \times 28$ |
| Max_pooling3D    | $2 \times 2 \times 2$ | $4 \times 14 \times 14$ |
| Convolutional3D  | 256     | $3 \times 3 \times 3$ | $4 \times 14 \times 14$ |
| Max_pooling3D    | $2 \times 2 \times 2$ | $2 \times 7 \times 7$ |
| Convolutional3D  | 256     | $3 \times 3 \times 3$ | $2 \times 7 \times 7$ |
| Max_pooling3D    | $2 \times 2 \times 2$ | $1 \times 4 \times 4$ |
| Fully Connected  | 1024    |                 | 1024            |
| Fully Connected  | 512     |                 | 512             |
| Fully Connected  | $3 \cdot (75)$ | $3 \cdot (75)$ |
| Softmax          | 8       |                 | 8               |

C3D: three-dimensional convolutional neural networks.

C3D feature extractor. The extracted video features are encoded into color maps according to a certain rule. As for skeleton data, we apply view-invariant transform to extract VIFs. Similarly, the VIF is also encoded into color maps according to a certain rule.

**Video data encoding**

Each video in the data set contains T frames, we set $V = \{t_i\}_{i=1}^T$ to represent it. We divided it into $(T - n_u - n_t)/n_i$ consecutive video units, where $n_u$ represents the frame number of a unit and $n_t$ represents the number of frames between two units. Then we can represent a unit as $u = \{t_i\}_{i=n_u}^{n_u+n_t}$. In this function, $s$ means the start frame, and $s + n_u$ represents the end frame. As the unit overlapped each other, the next unit is represented as $u = \{t_i\}_{i=s+n_u}^{s+n_u+n_t}$.

We used a visual encoder to process each unit to get a unit-level representation $M = E(u)$. In our method, we use C3D as the feature extractor.

We use C3D network to extract abundant spatiotemporal features from a given input video unit. The 3D convolutional deep networks perform well in extracting model appearance and motion features simultaneously. The structure of our C3D network is presented in Table 1. The input of the model is a video unit with dimension $R_{n_u \times H \times W \times 3}$, where $W$ and $H$, respectively, represent the height and width of video frame.

Du et al.\(^8\) empirically find that $3 \times 3 \times 3$ convolutional kernel has best performance among the set of explored finite structures. Therefore, all 3D convolutional kernels are $3 \times 3 \times 3$ and the stride in the spatial and temporal dimensions is set to 1. The 3D convolutional layers use $relu$ as the activation function. The 3D pooling layers are denoted from pool1 to pool5. All pooling kernels are $2 \times 2 \times 2$, except that pool1 is $1 \times 2 \times 2$.

The first fully connected layer has 1024 output neurons, and the second fully connected layer has 512 output neurons, both using relu as the activation function. Different from Du et al.\(^8\), we add an additional layer that has $3N$ neurons to extract the high-dimensional spatiotemporal information of the video unit while the activation function of this layer is tanh. The last layer of the C3D model outputs the categories of action. It has $C + 1$ output neurons, with $C$ represents the number of action classes in the data set. It is remarkable that we have one more action category in our data set, since the background is regarded as a class.

After training the C3D network, we remove the last layer; therefore, we can extract feature with dimension $3N$ from video. Then, we encode these consecutive feature vectors into a feature space, which is represented as color maps.

Like an RGB image, which has three channels, we write feature vector obtained by the $f$th unit through C3D extractor as a $N \times 3$ matrix $M^f_{video}$

\[
M^f_{video} = \begin{bmatrix}
m^f_{1,1} & m^f_{1,2} & m^f_{1,3} \\
\vdots & \vdots & \vdots \\
m^f_{N,1} & m^f_{N,2} & m^f_{N,3} 
\end{bmatrix}
\]

Convert it into a column of an RGB image

\[
R^f_{video} = \begin{bmatrix}
m^f_{1,1} & m^f_{1,2} & \ldots & m^f_{1,3} \\
m^f_{2,1} & m^f_{2,2} & \ldots & m^f_{2,3} \\
\vdots & \vdots & \ddots & \vdots \\
m^f_{N,1} & m^f_{N,2} & \ldots & m^f_{N,3} 
\end{bmatrix}
\]

\[
G^f_{video} = \begin{bmatrix}
m^f_{1,1} & \ldots & m^f_{1,2} \\
m^f_{2,1} & \ldots & m^f_{2,2} \\
\vdots & \ddots & \vdots \\
m^f_{N,1} & \ldots & \ldots & m^f_{N,2} 
\end{bmatrix}
\]

\[
B^f_{video} = \begin{bmatrix}
m^f_{1,1} & m^f_{1,2} & \ldots & m^f_{1,3} \\
m^f_{2,1} & m^f_{2,2} & \ldots & m^f_{2,3} \\
\vdots & \vdots & \ddots & \vdots \\
m^f_{N,1} & \ldots & \ldots & m^f_{N,3} 
\end{bmatrix}
\]

Representations of all units are arranged in chronologicial order to represent the whole video. Next, a mapping function is used to convert C3D features into RGB components. The arranged float matrix is quantified to integral image representation. Since the activation function of our last fully connected layer is $tanh$, the mapping function is defined as follows

\[
p_{video} = \text{floor}(255 \cdot \frac{p_{video} + 1}{2})
\]

where $p_{video}$ represents the pixel value and floor represents the rounding down function.

As shown in Figure 2, the top is a video in our data set. Below is the corresponding RGB image encoded from the features extracted by C3D. It can be observed from the encoded color map that there are some similar sequences, since several consecutive video units are similar. We can also see that there are many mutations in the color map, which is due to the large change in the state of the video unit. Generally, these mutations reflect that the object has undergone a change of motion during the period of time.

**Skeleton data encoding**

To make the skeleton sequence robust to viewpoint changes, Raptis et al.\(^48\) converted each skeleton into a...
standard pose. This method effectively eliminates the influence of viewpoint change and eliminates the partial relative action between the original skeletons. For example, the rotation of the waist will be eliminated by this transformation. Liu et al.34 proposed a sequence-based view-invariant transform to solve the problem of view variations. However, this method eliminates the relative action between the original skeleton and the space. Since the coordinate system is fixed on the human body, actions like body rotation cannot be described. Compared with the traditional skeleton-based transformation method, we propose a coordinate transformation based on the camera coordinate system, which eliminates the influence of the viewpoint change by retaining more relative actions in the original skeleton joints.

In general, the skeleton sequence we collect is based on the camera coordinate system, so the local coordinates \((x, y, z)\) are sensitive to the change of viewpoint. As shown in Figure 3(a), although one person is doing the same action, the position in the space changes greatly due to the different viewpoint if the camera is fixed.

In detail, given \(F\) frames of skeleton sequence, we can represent the \(n\)th skeleton joint of the \(f\)th frame as \(p_{nf} = (x_{nf}, y_{nf}, z_{nf})\), \(f \in (1, \ldots, F)\), \(N\) denotes the number of bone joints in each skeleton, and its value is determined by the skeleton estimation algorithms. For example, Kinect V2 can provide \(N = 25\) joint position information. Each joint \(p_{nf}\) contains five components: three local coordinates \((x, y, z)\), time label \(f\), and joint label \(n\).

Our method doses a translation to the camera coordinate system and makes its origin on the human body. After that, we choose the spinal joint point of the human body as the coordinate origin, whose joint number is 1. In the new coordinate system, the \(n\)th skeleton joint of the \(f\)th frame is represented as \(VIF_{nf} = \frac{x_{nf} - x_{1f}}{C_0}, \frac{y_{nf} - y_{1f}}{C_0}, \frac{z_{nf} - z_{1f}}{C_0}\).

This method has three advantages: first, the viewpoint influence can be eliminated. Even if the distance and angle between people and the camera are different, with a coordinate system that moves with the human body, it will not affect our algorithm. As shown in Figure 3(a) and (b), in the camera coordinate system, actions performed at different positions turn out to be the same after coordinate transformation, which means actions are independent of positions. Second, the relative action features of the joints are more obvious. Figure 3(c) shows the movement of the human’s right shoulder, right elbow, and right wrist joints, but since they are all in the camera coordinate system, the joint

---

**Figure 2.** Illustration of our video encoding method. We arrange the features extracted by C3D in order. Then they are encoded into a color map. C3D: three-dimensional convolutional neural networks.

**Figure 3.** Illustration of our view-invariant transform. Figures (a) and (c) show the camera coordinate system. Figures (b) and (d) show our coordinate system.
rotation is not described as it is in Figure 3(d). Third, as the coordinate system is not connected to the joints of the human body, rotation can also be well described.

From VIF to color maps. Since the position information \((x, y, z)\) of each joint is corresponding to each channel of RGB space. Like video-stream, we rearrange the \(f\)th VIF as a \(N \times 3\) matrix

\[
M^f_{sk} = \begin{bmatrix}
  x^f_1 & y^f_1 & z^f_1 \\
  x^f_2 & y^f_2 & z^f_2 \\
  \vdots & \vdots & \vdots \\
  x^f_N & y^f_N & z^f_N
\end{bmatrix}
\]

(6)

Convert it into a column of color maps

\[
R^f_c = [x^f_1, x^f_2, \ldots, x^f_N - x^f_1] \\
G^f_c = [y^f_1, y^f_2, \ldots, y^f_N - y^f_1] \\
B^f_c = [z^f_1, z^f_2, \ldots, z^f_N - z^f_1]
\]

(7) (8) (9)

To maintain time consistency with the C3D features input, the skeleton sequences are down sampled. Next, a mapping function is created to convert the spatial coordinates into RGB components

\[
p_{sk} = \text{floor}(255 \times \frac{p_{sk} - c_{\text{min}}}{c_{\text{max}} - c_{\text{min}}})
\]

(10)

where \(p_{sk}\) represents the pixel value. \(c_{\text{max}}\) is the maximum value of all the joints coordinates in training data set, whereas \(c_{\text{min}}\) is the minimum. With our coordinate transformation, it will not exceed the height of the human body.

Figure 4 shows that all the images encoded by the frame skeleton sequence are finally arranged in chronological order to represent the entire action sequence. The horizontal axis of the new RGB graph represents temporal information, whereas the vertical axis represents the spatial information. This RGB graph makes the global discrimination obvious and also shows the clear spatial distribution of the action features.

**Two-stream YOLO-based network**

TAD based on Kinect can be handled by using our two-stream YOLO-based networks. The spatiotemporal skeleton sequences and action videos are encoded into two separated RGB-like image streams. In general, YOLO uses the entire image as the input of the network and directly output the position and category of the bounding box and the confidence in the output layer. It predicts the positions and categories of multiple boxes simultaneously, enabling real-time end-to-end object detection and recognition. In this work, our networks are used to predict the action bounding box and corresponding categories and fuse them to refine the final result. Compared with standard YOLO-based object detection in 2D images, our TAD method is much simpler since the action detection problem is transformed into a 1D detection of the time slice of action.

**YOLO-based TAD**

In this work, each channel of our two-stream YOLO-based network receives an encoded RGB-like image stream, the width of each input image is augmented with time \(T\), and the height is equal to the number of Kinect skeletons \((N = 25)\). In general, a typical outputs of YOLO network include four increments \(x, y, t_w, t_h\) and \(C\), the detected action category and its corresponding probability \(p(C) \in (0, 1)\). However, in our 1D method, only two parameters \(x\) and \(t_w\) are used to refine the anchor box, as shown in Figure 5, and the detected \(i\)th action bounding box are described with parameters \(b_{i,x}\) and \(b_{i,w}\) by using the following equations

**Figure 4.** Illustration of our skeleton sequence encoding method. We arrange the VIFs in time order. Then they are encoded into a color map. VIF: view-invariant feature.
where \( c_{i,x} \) is the grid cell offset of the \( i \)th bounding box and 
\[
\sigma(x) = \frac{1}{1 + e^{-x}}
\]
represents the small offset from \( c_{i,x} \). 

The image is divided into \( S \) grids (\( S = 50 \) in this article), and each of them is used to predict action bounding box. Therefore, we can get a tensor with size \( S/C_2 \times A/C_2 \times (3 + C) \) as the output. In inference, we use \( C \times p(C) \) to find the location where a human action happens most likely and can easily obtain the action duration of \( i \)th detected action from \( t_{i,s} \) to \( t_{i,e} \) by using the following equations

\[
b_{i,x} = \sigma(x) + c_{i,x} \tag{11}
\]

\[
b_{i,w} = p_{i,w} e^v \tag{12}
\]

\[
t_{i,s} = b_{i,x} - \frac{b_{i,w}}{2} \tag{13}
\]

\[
t_{i,e} = b_{i,x} + \frac{b_{i,w}}{2} \tag{14}
\]

Another network output is the confidence score \( o \in [0, 1] \) that reflects the confidence in the box containing some action. We used logistic regression to predict the confidence score for each bounding box. The confidence scores should be 1 if the anchor box overlaps a ground truth action more than other anchor boxes. Only one anchor box is assigned to each ground truth action.

**Two-stream network**

Figure 6 shows the structure of our network. The input of video-stream is the image encoded from the video features. The input of skeleton-stream is the image encoded from the skeleton sequence.

We implement this model as a fully convolutional network to reduce the number of parameters. The structure of the two streams is the same. Each stream consists of four blocks and a convolutional layer with \( 1 \times 1 \) kernel. Each block consists of three convolutional layers and one max pooling layer. The batch normalization layer is added after each convolutional layer in the network, as it can achieve significant improvements in convergence while eliminating the requirements for other forms of regularization. Batch normalization also helps to normalize the model. We choose *LeakyReLU* as activation function

\[
y = \max(0, x) + \alpha \min(0, x) \tag{15}
\]
where α is a very small positive number to preserve the information of negative axis. To improve the effect of short-term action detection, we reshape the feature maps of the previous block and merge them with the feature maps of the last block.

Consider two kinds of fusion methods, boxes-fusion and layers-fusion, respectively. The boxes-fusion directly fuses the predicted bounding boxes from each stream and aims to solve the problem of missing frames. The layers-fusion is to fuse the second-to-last layers of two streams and then add a convolutional layer to output the bounding boxes. This method aims to fusing the features of two streams deeply and get some new features that would be better to detect. Finally, we use nonmaximum suppression (NMS) algorithm to filter the bounding boxes from two proposed fusion methods and generate the action detection result.

**Loss function**

The YOLOv2 algorithm uses the mean square error (MSE) loss function as it treated object detection as a regression problem. Our loss function has two main parts: position error and classification error.

The position error corresponds to the position error of the predicted bounding box, we use weight λcoord = 5. Classification error distinguishes whether a bounding box contains a target or not. For the latter case, weight λnoact = 0.5 is used. MSE will treat different sizes of bounding boxes equally. In practice, smaller bounding box would be more sensitive than the larger one to the coordinate error. Therefore, width prediction ̃w_i of bounding box is changed to its square root, which means the predicted value becomes (̃x_i, ̃w_i). The bounding box with no action only has the error of confidence score since the coordinate error cannot be calculated, and the classification error term is calculated only when the grid has an action.

Finally, the total loss function is defined as follows

\[
\text{Loss} = \lambda_{\text{coord}} \sum_{i=0}^{S} \sum_{j=0}^{A} y_{ij} (x_i - ̂x_i)^2 \\
+ \lambda_{\text{coord}} \sum_{i=0}^{S} \sum_{j=0}^{A} y_{ij} (\sqrt{w_i} - \sqrt{̂w_i})^2 \\
+ \sum_{i=0}^{S} \sum_{j=0}^{A} y_{ij} (x_i - ̂x_i)^2 \\
+ \lambda_{\text{noact}} \sum_{i=0}^{S} \sum_{j=0}^{A} y_{ij} (x_i - ̂x_i)^2 \\
+ \sum_{i=0}^{S} \sum_{j=0}^{A} y_{ij} (p_i - ̂p_i)^2 \tag{16}
\]

where 1_{y_{ij}} in the first item, means that there is an action in the i-th grid. The bounding box width error is represented by the second term, while the third one is the confidence error for whether or not the target locates in the bounding box. Confidence error is the fourth one. The classification error term for the grid that contains the target is represented at the last.

**Experiments**

To validate the algorithm presented in this article, we collected and manually annotated two data sets. We use separate training methods for two streams. For video-stream, we first train the C3D model. After extracting the features of video, we use the trained C3D model to encode the video as color maps. Then, these color maps are used to train the video-stream. For skeleton-stream, we encode the skeleton sequence into color maps and then use these color maps to train the two streams. Finally, we use two methods to fuse the two streams and some experimental results are presented accordingly.

**HITVS-v1 data set**

Most of existing data sets for TAD are video-only, such as THUMOS 2014 data set⁴⁹ and MEXAction2 data set.⁵⁰ Due to the particularity of our method, we built our own data set Harbin Institute of Technology Vision and Skeleton-version 1 HITVS-v1. HITVS-v1 is labeled manually at the frame level; we divide different actions in a video making the proposal accuracy up to one frame. Tagging format is start frame, end frame, and action class. All scenarios and human actions are coherent. The video of current existing data sets is basically built by using a montage method of different scenes and different human motion videos, and the coherence is very poor. Our data set is more consistent with the actual scene. Our two-stream approach requires one-to-one video and skeleton information, and the skeleton data are strictly corresponding to the video data, which are rare in current existing data sets.

We build the data set HITVS-v1 in a domestic scene. We use student dormitory with an area of about 10 m², which contains daily necessities such as chairs, desks, and single beds to simulate the family environment. We use Kinect V2 to collect data from two viewpoints, which can improve the collection efficiency and the generalization of the model. The data set contains two types of data: skeleton coordinate data (.txt) and RGB video (.avi). There are 2 × 100 files for each channel. We set the sampling frequency to FPS = 15 fps. There are 7 activities: eat, drink, read, sit down, stand up, lying down, and get up and 5 subjects, each subject performs 7 types of actions randomly for 20 times. Each time we collect 1000 frames and the duration is T ≈ 66.7 s. Each sequence contains 310 effective actions.

As shown in Figure 7(a), we take action sequences from two different viewpoints to highlight the advantages of our representation. Besides the change of viewpoint, five subjects’ actions were collected since there are significant

---

*Wang et al.*
variations among different subjects when they perform the same action, as shown in Figure 7(b).

**Train C3D**

The C3D model is not pretrained but trained directly with our own data set. For our data set HITVS-v1, the number of categories is eight (seven class actions and one background class). The input of the network is $16 \times 112 \times 112$. The skeleton data of our data set have 25 human body joints. To keep the input dimensions of the two streams consistent, we set $N = 25$.

Since the samples of the background action are much more than other actions, the number of each class of action in the mini-batch is set the same during the training. We use the Adam-gradient descent method with a learning rate of 0.001. When the loss of the validation set is no longer lowered for 10 consecutive times, the iteration stops. Figure 8 shows the training process and the confusion matrix for our data set. Finally, the accuracy in the test set of our data set is $88.22\%$. Due to the limited amount of data, our model performs overfitting as the training epoch increases, but we finally choose the model when the mAP reaches the highest value, and at this point, the model is not overfitting, so this phenomenon will not have an excessive impact on our experimental results.
Data enhancement

Our database is a moderate scale database; therefore, it is necessary to increase the samples from original data. We then propose a data enhancement method described as follows. For a data sequence with $L$ frames, the frame rate is FPS and the duration is $t = L / FPS$. We set the sequence duration to $T$ and encode the data sequence into a color map. Thus, the total number of consecutive frames can be represented as $f_c = T / FPS$. In general, $T < t$. We want to encode the starting frames of the skeleton data as $f_s$ and the number of cutoff frames as $f_e = f_s + f_c$ and ensure it to satisfy $f_e < L$. Therefore, we can set the sampling rate as $d$ and start sampling from the first frame. Figure 9 shows our data augmentation method.

Train video-stream network

We use the trained C3D model to encode the video of the entire data set into color maps. Since our data set has 1000 frames per video, we set $n_i = 2$ and $f_s = 800$. A video of consecutive 800 frames is encoded as a $400 \times 25$ RGB image.

To automatically find good prior anchor boxes from the input stream, we follow the same idea in the YOLOv2 detection framework, which adopts K-means clustering to predict the prior shape of action anchor box by using an IOU-based distance. Let $b_j$ be the $j$th candidate anchor box representing action duration, we then define the distance $d_{ij}$ between $b_j$ and the $i$th cluster centroid $c_i$ as follows

$$d_{ij} = 1 - \text{IOU}(b_j, c_i), j \in M, i \in K$$

where $M$ is the number of actions in the training data set and $K$ is the number of possible actions randomly chosen in a stream image grid. Through this algorithm, we can get $K$ good prior anchor boxes. When training our network, we assign $K$ good prior anchor boxes to each grid to predict action durations.

In this article, for HITVS-v1, we set $K = 6$, $p_w \in \{1.625, 3.5, 8.375, 11.5, 15.625\}$. We use the NMS algorithm to filter the detected actions and set confidence threshold = 0.3. Since there are less than 10 actions in a sequence, we set max boxes = 10. The actions in our data set do not have a large overlap; we set IOU threshold = 0.1 (temporal intersection over union).

The length of sample actions ranges from 10 to 140 frames in our data set. Therefore, the setting of the receptive field has a great influence on the accuracy rate. The original receptive field of our network is 105. We increase the receptive field by dilated convolution and expand the receptive field to 121 and 137 to test the video-stream.

The AP@IOU = 0.5 (average precision) of video-stream for each class in our data set is shown in Figure 10. When the receptive field is 105, 121, 137, the mAP of video-stream is 74.64%, 80.79%, and 77.86%, respectively. The impact of receptive field on our method is relatively large. However, it does not mean that the bigger the receptive field, the higher the precision. The precision may be related to the duration of the actions in our data set.

Train skeleton-stream network

In our data set, we set $n_i = 2$ and $f_s = 800$. To maintain time consistency with the video-stream input, the skeleton sequences are down-sampled. A skeleton sequence of length 800 is encoded into a $400 \times 25$ RGB image.

The anchor boxes are the same as video-stream. We also use the dilated convolution method to expand the receptive field. The experimental results are shown in Figure 11. When the receptive field is 105, 121, and 137, the mAP
of skeleton-stream is 74.22%, 78.88%, and 72.81%, respectively. This is similar to the experimental results obtained by video-stream, which is the highest when the receptive field is 121.

To verify our VIF-based skeleton-stream network, we test the precision with skeleton-stream-121. As no coordinate transformation is used, the feature is defined as follows

\[ R_f^i = [x_1^f, x_2^f, \ldots, x_N^f] \]  
\[ G_f^i = [y_1^f, y_2^f, \ldots, y_N^f] \]  
\[ B_f^{i} = [z_1^f, z_2^f, \ldots, z_N^f] \]

We encode the skeleton sequences in two ways. Then we train skeleton-stream network with the encoded images. Figure 12 shows the recall–precision curves for two kinds of actions. When there is no coordinate transformation on the skeleton sequences, the mAP is 76.51%. Our method can increase the mAP to 80.79%, which proves that our coordinate transformation method can effectively improve the precision.

**Two-stream fusion**

The boxes-fusion is to fuse the predicted bounding boxes separately generated from two streams. The layers-fusion is to fuse the second-to-last layers of two streams and add a convolutional layer. We fix the parameters of other layers and train this convolutional layer separately. Finally, the NMS algorithm is used to filter the bounding boxes after two streams fusion.

The test results with \( \text{IOU} = 0.5 \) are presented in Table 2. We tested video-stream, skeleton-stream, boxes-fusion, and layers-fusion separately. When the receptive field is 105, the mAP of layers-fusion is greater than that of video-stream and skeleton-stream. When the receptive field is 121, the mAP of layers-fusion is less than the others. When the receptive field is 137, the mAP of layers-fusion is between the others.

However, the mAP of boxes-fusion is always greater than the others. As can be seen from Figure 13, the AP of

| Receptive field | 105 | 121 | 137 |
|-----------------|-----|-----|-----|
| Video-stream    | 74.64% | 78.88% | 77.86% |
| Skeleton-stream | 74.22% | 80.79% | 72.81% |
| Boxes-fusion    | 84.11% | 87.01% | 84.09% |
| Layers-fusion   | 75.00% | 77.22% | 74.72% |

TAD: temporal action detection; mAP: mean average precision; HITVS: high-speed interplanetary tug/cocoon vehicles.
each class of video-stream and skeleton-stream is different. The two streams are complementary; therefore, using the fusion of two streams to detect can improve the final TAD performance.

Figure 13 shows the IOU-mAP curves for the four methods when the receptive field is 121. We can see that the boxes-fusion detection effect is better than other methods when the IOU is less than 0.67. Layers-fusion detection effect is better than other methods when IOU is greater than 0.67.

As shown in Figures 14 and 15, we present two TAD examples in which two encoded input streams of untrimmed Kinect data, which include action video and skeleton streams, are fed into our network. The ground truth of each action (lying down, get up, read, stand up, eat, drink, and sit down) is shown below, notice that the duration and time slice of each action are different from each other. We illustrate four different TAD methods based on, video-stream only, skeleton-stream only, boxes-fusion, and layers-fusion. As human action in Kinect is a high-dimensional spatiotemporal signal, which includes video and skeleton information, TAD based on single information is difficult. Notice that, using video sequence or skeleton sequence alone for TAD will cause large errors, there are missed detections of some actions. However, based on boxes-fusion of our two-stream fusion method, the missed actions (read in Figure 14 and eat in Figure 15) are detected correctly. Another observation reveals that the boxes-fusion method sometimes performs better than the layers-fusion method, since no action detections are missed in both figures.

The reason why boxes-fusion performs better than layers-fusion possibly is due to the misalignment of semantic features when different feature layers, which do not have strong relationships, are fused together, while the boxes-fusion directly uses the bounding boxes generated from two separated stream channels.

**Stabilization performance**

In this experiment, we present stable performance comparison between video-based method and our two-stream fusion method based on a Kinect data stream over 58 s. In this test data stream, the actions include read, eat, sit down, stand up, drink, lying down, get up, and other, which mean doing nothing. As shown in Figure 16(a), some video keyframes are sampled from data streams in temporal order. We here use the box-fusion method for our two-stream fusion method, and the networks’ parameters are chosen as before. As shown in Figure 16(b), the black line represents the ground truth, the red line corresponds to our box-fusion method, and the blue line is the video-based method, the latter two lines are slightly offset from each other for clarity. The observation reveals that our method almost tracks the ground truth, and no actions are recognized incorrectly except that there are
slight errors on the start or end time of some actions. Video-based method is sensitive to the changes between two consecutive actions, and it can be seen that, in the video-based method, significant misclassifications happened near the boundary of two consecutive frames during these transitions. Some sampled frames during the changes of action states are shown in Figure 16(c), the first row corresponds to some images misclassified by the video-based method, whereas the second row shows the corresponding frames classified by our method, and the recognized action class with confidence score for each frame is overlapped the left top of each image. The first three images from the first row demonstrate that similar images grabbed from other action video sequences are recognized incorrectly as read, get up and eat. The people just finished drink action and do nothing, and at this point, the video-based method just thinks the people are planned to stand up while our method remains the decision. The last column image tells that there are some confusing frames in action lying down and get up, they occur at different points of time, and the video-based method cannot distinguish the difference; however, our method presents the stable result. Therefore, our method is much more stable in TAD performance than the video-based method.

Comparisons with the state-of-the-arts

Although there are many attempts for TAD, TAD methods based on the fusion of video and skeleton sequence are very rare. Hence, we compared the quality of our work to the methods7,8,51 that are closely related to our work but tests on a quite different data set.

As presented in Table 3, these methods use well-trimmed video clips as their train set, whereas untrimmed video clips as validation set and test set. Even though their data set contains a richer variety of actions, the well-trimmed train set will reduce the training difficulty. Besides, skeleton sequence contains abundant action information, and the lack of skeleton data is not conducive to the improvement of results. As HITVS-v1 contains both video and skeleton data, our method was trained and tested on them, which makes ours clearly perform better. Without fusion, single stream from video and skeleton achieves mAP@IOU = 0.5 to 78.88% and 80.79%, respectively. Based on multisource fusion, the results of two-stream boxes-fusion can significantly improve mAP@IOU = 0.5 to 87.01%.

Experiment on PKU-MMD data set

To better prove the effectiveness of our TAD approach, we conduct experiments on the PKU-MMD public data set.52 This 3D human behavior public data set is a new large-scale data set for continuous multimodality 3D human action understanding and covers a wide range of complex human activities. It contains 1076 long video sequences in 51 action categories, performed by 66 subjects in 3 camera views. The data set provides multimodality data sources, including RGB, depth, infrared radiation, and skeleton,
which are quite consistent with our methods. The data set has two parts: cross-view evaluation and cross-subject evaluation. Cross-view evaluation aims to test the robustness in terms of transformation, whereas the cross-subject evaluation aims to test the ability to handle intra-class variations among different actors. Our four methods are

![Table 3. Comparison with the state-of-the-arts.](image)

| Methods                     | Data set                  | Video | Skeleton | mAP@IOU0.5    |
|-----------------------------|---------------------------|-------|----------|--------------|
| Shou et al.                  | THUMOS2014                | ✓     | —        | 19.0%        |
| Yeung et al.                 | THMOS2014                 | ✓     | —        | 17.1%        |
| Chunhui et al.              | THMOS2014                 | ✓     | —        | 19.0%        |
| Yeung et al.                 | ActivityNet Sports        | ✓     | —        | 36.7%        |
| Chunhui et al.              | MPII Cooking              | ✓     | —        | 14.9%        |
| Chunhui et al.              | Max Planck Institute for Informatics (MPII) | ✓ | — | 61.1% |
| Zhu and Newsam              | PKU-MMD                   | ✓     | ✓        | 69.9%(cv)|45.2%(cs) |
| Zhu and Newsam              | PKU-MMD                   | ✓     | ✓        | 54.5%(cv)|47.9%(cs) |
| Zhu and Newsam              | PKU-MMD                   | ✓     | ✓        | 47.6%(cv)|44.4%(cs) |
| Our video-stream            | PKU-MMD                   | ✓     | —        | 86.41%(cv)|76.91%(cs) |
| Our skeleton-stream         | PKU-MMD                   | —     | ✓        | 84.77%(cv)|76.49%(cs) |
| Boxes-fusion                | PKU-MMD                   | ✓     | ✓        | 91.32(cv)%|80.90%(cs) |
| Layers-fusion               | PKU-MMD                   | ✓     | ✓        | 83.73%(cv)|74.15%(cs) |
| Our video-stream            | HITV-v1                   | ✓     | —        | 80.79%      |
| Our skeleton-stream         | HITV-v1                   | —     | ✓        | 78.88%      |
| Boxes-fusion                | HITV-v1                   | ✓     | —        | 87.01%      |
| Layers-fusion               | HITV-v1                   | ✓     | ✓        | 77.22%      |

mAP: mean average precision; HITVS: high-speed interplanetary tug/cocon vehicles.
all tested on both evaluations. We fixed the receptive field as 121 and IOU as 0.5, and the results in Figure 17 show that box-fusion performs best among the four methods, video-streams is the next, and layers-fusion in the worst. The highest mAP can reach 80.90% on cross-subject evaluation and 91.32% on cross-view evaluation. We also present some comparisons with the work\textsuperscript{52} in Table 3, which proves that our method provides an excellent performance for TAD based on PKU-MMD data set.

**Fall down detection**

For an elderly care service robot, it is of great importance to detect dangerous human actions such as falling or fainting. When the robot predicts that the behavior of interactive person is abnormal, it will capture the abnormal state picture or video, turn on the remote monitoring mode to communicate with the remote person, and execute the corresponding action accordingly. This requires our detection method to have a good online detection effect. To prove the significance of our method for elderly care service robot, we conduct an experiment on fall down detection, as shown in Figure 18. Our experiment platform is an intelligent service robot, as shown in Figure 19; Kinect is used to capture human actions. Figure 20 shows a user interface (UI) for fall down detection. The interface mainly includes current video frame overlapped with skeleton node, coded color map, and recognition result.

As shown in Figure 21, in this experiment, volunteers are asked to walk back and forth within the robot viewpoint and randomly perform different actions such as sit, squat, stand, wave, walk, and fall down. The robot will determine whether the observed people have fallen by using its recognition results based on Kinect. The robot outputs all the recognized actions except fall down as “not-down,” while classifies the action as “down” in UI when the volunteer falls down completely. Considering that only one recognition result will not make a proper decision; therefore, a
simple voting scheme (three positive decisions out of five) is used which may lead to a certain lag in the recognition process, as shown in Figure 21(b).

We also present our robot with 20 data sequences from different people by using our algorithm; among them, 10 sequences are from falling down while others are background sequences, and we get recall = 1 and accuracy = 95% on these sequences. Figure 22 is an example sequence for our online detection. The result shows that our model can detect falling actions very well, which provides the possibility to realize the fall warning function of the elderly care robot.

Figure 20. An interactive interface for fall down detection.

Figure 21. Online fall down detection experiment. (a) The results of detection based on different actions. (b) A certain lag in the recognition process.

Figure 22. Fall down detection from a data sequence sample.
Conclusion

In this article, we proposed a TAD deep learning network, which incorporates video and skeleton information from Kinect sensor. We adopt 3D-CNN to extract C3D motion features from video and apply to view-invariant feature transformation on skeleton data. Both types of features are encoded into RGB-like feature space, so that the TAD problem can be naturally transformed into the object detection problem. We then use the proposed Two-Stream YOLO-based network and fuse every stream based on Boxes-Fusion or Layers-Fusion methods. Experimental results show that it is feasible to convert the TAD problem into the object detection problem. Both video-stream and skeleton-stream can effectively locate the action time and classify the corresponding action. The two kinds of data streams are complementary. When two streams are fused, the precision can be improved and the final recognition results will be more stable than the video-based method. Another observation reveals that as IOU gets larger, the precision turns higher for boxes-fusion and turns lower for layers-fusion. We also found that the receptive field has a great impact on accuracy. Receptive field too big or too small will reduce the accuracy. To prove the practicability of our model, we also carried out a fall detection experiment based on an elderly care service robot. The result shows that our model can detect falling actions very well, which provides the possibility to realize the fall warning function of the robot. However, our approach has to train C3D networks and to extract VIF features before using two-stream networks; this is not as convenient as an end-to-end model. Another issue is that the method we proposed in this article is mainly suitable for single-person action recognition in fixed scenes; however, it is not yet possible to deal with multi-person action recognition in changing scenes. In this article, we mainly focus on the TAD problem for one person, whereas for multi-person TAD problem, our method is not applicable. In addition, when dealing with changing background or dynamic scenes, our method is also not suitable. In the near future, we will investigate some of these issues to improve the performance of our method.

Declaration of conflicting interests

I would like to declare on behalf of my coauthors that any figure or text taken from another article is clearly indicated with the full source and permission of the authors of the said source. Besides, our manuscript has not been published previously and is not under consideration for publication elsewhere in whole or in part. All the authors listed have approved the present submitted version.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported in part by the National Key Research and Development Program of China (2019YFB1310004) and the Self-Planned Task (NO. SKLRS202111B) of the State Key Laboratory of Robotics and System (HIT).

ORCID iD

Ke Wang https://orcid.org/0000-0002-5615-0847

Supplemental material

Supplemental material for this article is available online.

References

1. Fox JA and Fox MA. Falls in older people—an overview for the acute physician. Acute Med 2011; 10(2): 99–102.
2. Noury N, Rumeau P, Bourke AK, et al. A proposal for the classification and evaluation of fall detectors. IRBM 2008; 29(6): 340–349.
3. Krizhevsky A, Sutskever I, and Hinton G. ImageNet classification with deep convolutional neural networks. Adv Neural Inform Process Syst 2012; 25(2): 84–90.
4. Sevilla-Lara L, Liao Y, Güney F, et al. On the integration of optical flow and action recognition. In: 40th German conference on pattern recognition, Stuttgart, Germany, 9–12 October 2018, pp. 1–10.
5. Zhu J, Zhu Z, and Zou W. End-to-end video-level representation learning for action recognition. In: 2018 24th International Conference on Pattern Recognition (ICPR), Beijing, China, 20–24 August 2018, pp. 645–650.
6. Mettes P, Gemert J, Cappallo S, et al. Bag-of-fragments: selecting and encoding video fragments for event detection and recounting. In: 5th ACM on international conference on multimedia retrieval, Shanghai China, 23–26 June 2015, pp. 1–8.
7. Shou Z, Wang D, and Chang SF. Temporal action localization in untrimmed videos via multi-stage cnns. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016, pp. 1049–1058.
8. Yeung S, Russakovsky O, Mori G, et al. End-to-end learning of action detection from frame glimpses in videos. In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016. pp. 1–10.
9. Simonyan K and Zisserman A. Two-stream convolutional networks for action recognition in videos. In: 27th international conference on neural information processing systems, Montreal, Canada, 8–13 December 2014, pp. 568–576.
10. Zhao Y, Xiong Y, Wang L, et al. Temporal action detection with structured segment networks. In: 2017 IEEE international conference on computer vision, Venice, Italy, 22–29 October 2017, pp. 2914–2923.
11. Shou Z, Chan J, Zareian A, et al. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In: IEEE conference on computer vision and pattern recognition, Honolulu, HI, USA, 21–26 July 2017, pp. 5734–5743.
12. Xiong Y, Zhao Y, Wang L, et al. A pursuit of temporal accuracy in general activity detection, https://arxiv.org/abs/1703.02716v1 (2017, accessed 20 January 2020).
13. Gao J, Yang Z, Sun C, et al. Turn tap: temporal unit regression network for temporal action proposals. In: 2017 IEEE international conference on computer vision (ICCV), Venice, Italy, 22–29 October 2017, pp. 3648–3656.
14. Chao YW, Vijayanarasimhan S, Seybold B, et al. Rethinking the faster r-cnn architecture for temporal action localization. In: 2018 IEEE/CVF conference on computer vision and pattern recognition, Salt Lake City, UT, USA, 18–23 June 2018, pp. 1130–1139.
15. Zhang Z. Microsoft Kinect sensor and its effect. *IEEE Multimedia* 2012; 19(2): 4–10.
16. Li Y, Lan C, Xing J, et al. Online human action detection using joint classification-regression recurrent neural networks. In: European conference on computer vision, Amsterdam, The Netherlands, 11–14 October 2016, pp. 203–220. Cham: Springer.
17. Tran D, Bourdev L, Fergus R, et al. Learning spatiotemporal features with 3D convolutional networks. In: 2015 IEEE international conference on computer vision (ICCV), Santiago, Chile, 7–13 December 2015, pp. 4489–4497.
18. Du Y, Fu Y, and Wang L. Skeleton based action recognition with convolutional neural network. In: 2015 3rd IAPR Asian conference on pattern recognition (ACPR). Kuala Lumpur, Malaysia, 3–6 November 2015, pp. 1–4.
19. Zhang TY and Zhang JS. Survey and analysis on the functional needs of elderly companion robot. *Soc Welf* 2019; 8: 52–57.
20. Kostavels I, Vasileiadis M, Skartados E, et al. Understanding of human behavior with a robotic agent through daily activity analysis. *Int J Soc Robot* 2019; 11: 1–26.
21. Ramirez-Amaro K, Yang Y, and Cheng G. A survey on semantic-based methods for the understanding of human movements. *Robot Auton Syst* 2019; 119: 31–50.
22. Yang X and Tian YL. Super normal vector for activity recognition using depth sequences. In: 2014 IEEE conference on computer vision and pattern recognition (CVPR). Columbus, OH, USA, 23–28 June 2014, pp. 804–811.
23. Du Y, Wang W, and Wang L. Hierarchical recurrent neural network for skeleton based action recognition. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), Boston, MA, USA, 7–12 June 2015, pp. 1110–1118.
24. Wang L, Xiong Y, Wang Z, et al. Temporal segment networks: towards good practices for deep action recognition. In: 14th european conference on computer vision, Amsterdam, The Netherlands, 11–14 October 2016, Part VIII, pp. 1–16.
25. Cheng G, Wan Y, Saudagar AN, et al. Advances in human action recognition: a survey, https://arxiv.org/abs/1501.05964 (accessed 8 May 2020).
26. Lan Z, Lin M, Li X, et al. Beyond Gaussian pyramid: multiscale feature stacking for action recognition. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR). Boston, MA, USA, 7–12 June 2015, pp. 204–212.
27. Xu Z, Yang Y, and Hauptmann AG. A discriminative cnn video representation for event detection. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), Boston, MA, USA, 7–12 June 2015, pp. 1798–1807.
42. Sun C, Shetty S, Sukthankar R, et al. Temporal localization of fine-grained actions in videos by domain transfer from web images. In: 23rd ACM international conference on Multimedia, Brisbane, Australia, 26–30 October 2015, pp. 371–380.
43. Szegedy C, Toshev A, and Erhan D. Deep neural networks for object detection. In: Proceedings of the 26th international conference on neural information processing systems, Lake Tahoe, Nevada, 5–10 December 2013, pp. 2553–2561.
44. Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans Pattern Anal Mach Intell 2017; 39(6): 1137–1149.
45. Redmon J and Farhadi A. Yolo9000: better, faster, stronger. In: 2017 IEEE conference on computer vision and pattern recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017, pp. 6517–6525.
46. Redmon J and Farhadi A. Yolov3: an incremental improvement, https://arxiv.org/abs/1804.02767 (accessed 30 August 2020).
47. Liu W, Anguelov D, Erhan D, et al. Ssd: Single shot multibox detector. In: 14th European conference on computer vision, Amsterdam, Netherlands, 11–14 October 2016, Part I, pp. 21–37.
48. Raptis M, Kirovski D, and Hoppe H. Real-time classification of dance gestures from skeleton animation. In: Proceedings of the 2011 Eurographics/ACM SIGGRAPH symposium on computer animation, SCA 2011, Vancouver, BC, Canada, 5–7 August 2011, pp. 1–11.
49. Jiang YG, Liu J, Roshan Zamir A, et al. THUMOS challenge: action recognition with a large number of classes, http://crcv.ucf.edu/THUMOS14/ (accessed 3 March 2020).
50. Stoian A, Ferecatu M, Benois-Pineau J, et al. Fast action localization in large-scale video archives. IEEE Trans Circuits Syst Video Technol 2016; 26(10): 1917–1930.
51. Chunhui L, Yueyu H, Yanghao L, et al. PKU-MMD: a large scale benchmark for continuous multi-modal human action understanding, https://arxiv.org/abs/1703.07475 (accessed 1 May 2021).
52. Zhu Y and Newsam S. Efficient action detection in untrimmed videos via multi-task learning. In: 2017 IEEE winter conference on applications of computer vision, Santa Rosa, CA, USA, 24–31 March 2017, pp. 197–206.