Neurocomputing 523 (2023) 58–68

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Improving dynamic gesture recognition in untrimmed videos by an online lightweight framework and a new gesture dataset ZJUGesture

Chao Xu [a,1], Xia Wu [a,1], Mengmeng Wang [a,1], Feng Qiu [a], Yong Liu [a,*], Jun Ren [b]

[a] State Key Laboratory of Industrial Control Technology and Institute of Cyber-systems and Control, Zhejiang University, China
[b] Beijing Institute of Mechanical and Electrical Engineering, China

ARTICLE INFO

Article history:
Received 22 June 2022
Revised 1 December 2022
Accepted 11 December 2022
Available online 15 December 2022
Communicated by Zidong Wang

Keywords:
Gesture recognition
Gesture dataset
Human–computer interaction
Temporal relation
Video recognition

ABSTRACT

Human–computer interaction technology brings great convenience to people, and dynamic gesture recognition makes it possible for a man to interact naturally with a machine. However, recognizing gestures quickly and precisely in untrimmed videos remains a challenge in real-world systems since: (1) It is challenging to locate the temporal boundaries of performing gestures; (2) There are significant differences in performing gestures among different people, resulting in a variety of gestures; (3) There must be a trade-off between the accuracy and the computational consumption. In this work, we propose an online lightweight two-stage framework, including a detection module and a gesture recognition module, to precisely detect and classify dynamic gestures in untrimmed videos. Specifically, we first design a low-power detection module to locate gestures in time series, then a temporal relational reasoning module is employed for gesture recognition. Moreover, we present a new dynamic gesture dataset named ZJUGesture, which contains nine classes of common gestures in various scenarios. Extensive experiments on the proposed ZJUGesture and 20-bn-Jester dataset demonstrate the attractive performance of our method with high accuracy and a low computational cost.

1. Introduction

With the significant progress of science and technology, people are surrounded by many electronic devices, such as computers, phones, and smartwatches. Thus it raises an increasing demand for flexible and effective human-computer interaction. Compared with the traditional forms of keyboards and touchscreen, the dynamic gesture is a more intuitive and natural way to interact. For example, in a noisy environment where speech recognition systems do not work, gesture interaction is an effective touch-free way. Besides, using in-air gestures in human-computer interaction can be safer and more convenient when driving a car.

Dynamic gesture recognition is the task to identifying the specific category of gestures in the videos. Many works have achieved excellent performance, including wearable hardware-based methods [1–4], ultrasound-based method [5], and computer vision-based methods [6–13]. The first method is based on information captured from special sensors, such as the motion profile and the position. Although they meet the performance requirements, the attached devices would be inconvenient in the application. The second one utilizes the Doppler ultrasound, which is easily affected by environmental noise, and suffering from difficulties when deploying. While the vision-based method does not need additional equipment except for the image acquisition sensors, and the development of deep convolutional neural networks brings significant improvements for gesture recognition.

Although deep networks perform well in this task, they fail to meet the requirement of real-time gesture recognition systems. In these systems, dynamic gesture recognition faces many open challenges. Firstly, different from the trimmed video clip that only contains a single gesture, untrimmed videos usually comprise an unknown number of gestures, which have the diverse appearance and no indication of the start and end timestamps. Secondly, the response delay of the real-time system must be short enough to provide immediate feedback. Therefore, the lightweight model is expected in the whole system. Thirdly, we need a robust system to deal with the challenges of illumination, complicated backgrounds, and intra-class variations in practical application. Notably, a complete dynamic gesture can be divided into three parts: preparation, the core of the gesture, and retraction, of which the second part is the most important to discriminate inter-class differences. While the other two parts are very similar among different categories, which is easy to cause misrecognition.

* Corresponding author.
E-mail address: yongliu@iipc.zju.edu.cn (Y. Liu).
[1] Chao Xu, Xia Wu, and Mengmeng Wang contribute equally to this work.

https://doi.org/10.1016/j.neucom.2022.12.022
0925-2312/

For CNN-based methods, the size and quality of datasets seriously affect the final performance, so many datasets have emerged in the field of gesture recognition. Unlike HDMB [14], UCF101 [15], and other action datasets, gesture datasets emphasize the importance of temporal relational reasoning. There are common gesture datasets, such as EGOGesture [16], NvDataset [17], 20-bn-Jester [18]. However, there are still some defects in these datasets: 1) The transition and core actions are not clearly divided. 2) The collection distance is far, which leads to lots of noise, *e.g.*, arm actions. 3) Most of the datasets have certain standards for the length of the video, which means that the gestures in these datasets have similar action speeds and cannot adapt to different operating habits among different users in the real world. We urgently need a dynamic gesture dataset that is more focused on one-handed human–computer interaction and should have the following characteristics: 1) The preparation and retraction in the complete gesture are classified into the *no gesture* category, and only the most distinct parts are divided into the defined gesture category. In this way, we can obtain more accurate response in the untrimmed videos without being affected by the transition parts that often cause error responses. 2) Images in the video clips need to pay more attention to hand movements to be closer to the human–computer interaction needs of handheld devices, such as the most common mobile phones, which focus on hand movements and exclude other unrelated noise. 3) The gestures in the dataset should have various moving speeds. Even for the same category of action, the length of each sample needs to be diverse to simulate different users.

In this paper, we propose an online lightweight two-stage framework to precisely detect and classify dynamic gestures for the raw video streams in real-world systems with a single RGB camera. We design an efficient detection module that consists of a MotionNet and the post process to distinguish whether there are gestures. Then the cleaned video sequence is sent to the following gesture recognition module, which consists of a temporal relational reasoning network (GestureNet) and a single-time filter to identify the specific gesture category. Our method benefit from the two-stage phase is three folds: 1) The gesture recognition module only turn on if there is a gesture in the video clip. Since the computation of the detection module is lower than that of the gesture recognition module, thus it dramatically reduces the power consumption of the overall system. 2) The detection module reduces the noise caused by unknown hand movements. 3) Two modules are highly related, and the role of the gesture recognition module is used to refine the output of the detection module to ensure that the whole system is more reliable than the single-stage system.

Moreover, we propose a ZJUGesture dataset to meet the practical application requirements. This dataset distinguishes transitional movements, focuses on hand movements, and contains different action speeds. We define nine common categories of gestures, including no gesture, swipe left, swipe right, push, turn clockwise, turn counterclockwise, palm to fist, fist to palm and fold up. As mentioned earlier, a complete gesture should be divided into three parts. For example, the action of changing a palm into a fist can be divided into: the preparation action of the hand into the camera field of view to form the palm, the core action of the palm into the fist, and the exit action of leaving the field of view of the camera after turning into the fist. Fig. 1 shows an example of the partition for the whole gesture. With this in mind, we clean every gesture in the ZJUGesture dataset into preparation, core action, and retraction. To the best of our knowledge, we are the first to clean a complete gesture into three partitions which is convenient for algorithm design. The preparation and retraction are classified as *no gesture* for training, effectively reducing the misrecognition of unknown gestures. Our dataset is captured under different speeds and complex scenarios, *e.g.*, diverse illuminations and backgrounds, to cover most circumstances and users.

The contributions of this paper are summarized as follows:

- We propose an online lightweight two-stage framework for dynamic gesture recognition in raw video streams, which can handle untrimmed videos and achieve high precision with a low computational cost.
- A detection network that combines the texture and motion features in the untrimmed video stream through RGB images and differential images is introduced to locate gestures in time series. And a classification network is employed to deduce temporal relationships at multiple time scales. Both networks are highly efficient and effective.
- We present a new gesture dataset, termed ZJUGesture, which focuses on solving one-handed operation scenarios in practice. This dataset aims to improve the diversity of gestures, like different speeds, illumination changes, and lots of scenarios. Besides, we employ more fine-grained annotation to reduce frequent error responses in real systems.

We organize this paper as follows: Section 2 reviews related research works about motion detection, gesture recognition, and gesture recognition dataset. Section 3 introduces the framework and details of the proposed method for dynamic gesture recognition in untrimmed videos. Section 4 introduces our ZJUGesture dataset in detail. In Section 5, comprehensive experiments are illustrated, and the effectiveness of the proposed method is evaluated. Section 6 presents the conclusion.

2. Related Work

2.1. Motion Detection Methods

Motion detection is the process of obtaining motion information from the video. Christopher Richard Wren et al. [19] establish a Gaussian background model to segment moving foreground object. Although these approaches are simple, they lack robustness in practical application because the influence of the changed illumination and background. The method [20] computes the change of pixels in the time domain and the correlation between adjacent frames to get the motion information of the object in both camera stationary and moving situations. Limited to an enormous amount of calculation, these optical flow estimation methods are too hard to be applied in real-time systems.

With the development of deep learning, more and more algorithms [21,22] track moving objects through the features extracted by the networks. The sophisticated methods can maintain high accuracy in variable scenarios but come with heavy consumption. Therefore, a lightweight network with excellent performance is more suitable for edge computing scenarios.

2.2. Gesture Recognition Methods

Gesture recognition has been widely investigated. Early efforts [23–25] adopt low-level features, such as skin color and motion features, to track the hand and then identify the specific gesture category by the classifier. Recently, deep-learning-based methods have achieved tremendous success in many computer vision tasks. Similar to action recognition, it is critical to extract spatial and temporal features of the dynamic gestures. Specifically, some works design the two-stream networks, which rely on isolated texture encoder and motion representation, *e.g.*, optical flow [26–28], and motion vectors [29]. Other methods use recurrent neural networks to model temporal information. For example, [30] extracts

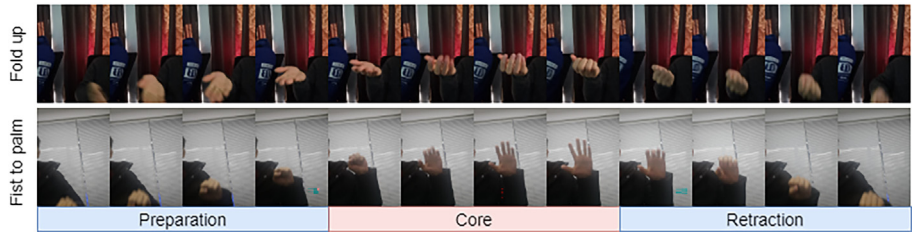

Fig. 1. Two examples of a complete gesture partition.The gesture types for the two rows are *fold up* and *fist to palm*, respectively. In this paper, each complete gesture is divided into three sub-parts.

the features of each frame of the video by 2D CNNs, then models temporal information through LSTM [31], finally get video-level predictions. Furthermore, to better model spatial and temporal at the same time, 3D CNNs have been proposed. C3D [32], I3D [26], and P3D [33] utilize 3D CNNs to improve the situation that 2D CNNs severely compress the inter-frame motion information in the temporal dimension. Consequently, many researchers [26,34–40] explore 3D convolutions, which extract spatial and temporal features simultaneously. However, the aforementioned methods only support the trimmed video clips that contain a single gesture, which prevent their real-world applications. This paper focuses on untrimmed video streams obtained directly from the camera. The common solution [17] is adding an extra no gesture class and processing it together with the gesture classes. Some works have the same spirit as ours. [34] first trains a binary classifier to detect whether gestures appear. A followed predictor outputs the specific category if there is a gesture in the video clip. Compared with it, our method takes temporal features into the first stage and effectively models the multi-scale short- and long-term temporal cues in the second stage.

2.3. Real-time Recognition

Although existing methods have achieved significant improvements, they still suffer inconvenience in real-time deployment due to the heavy parameters. To address this challenge, Zhang et al. [29] design the enhanced motion vectors as motion features instead of optical flows to boost the speed of the network. Sung et al. [41] present an on-device real-time gesture recognition system based on RGB frames. Recently, some works [42,34] propose a deep 3D CNN for gesture classification and a lightweight gesture detector. They still employ 3D CNN as backbones, which will inevitably increase the amount of calculation. In contrast, our method only adopts 2D network and designs some structures to supplement the temporal information.

2.4. Gesture Recognition Datasets

Some gesture recognition datasets are collected for human–computer interaction or sign language understanding. For example, NVIDIA Gesture [17] focuses on the interactions while driving, so the clips are recorded inside a car simulator. EgoGesture dataset [16] is a multi-modal large-scale dataset for egocentric hand gesture recognition, which provides both RGB and depth videos. ChaLearn ConGD [43] is also captured by the Kinect devices. But this dataset requires the hand gesture and camera to be as close as possible. For Jester [18], it adopts the computer camera or laptop to record gestures, which are used to device-freely control the computer. In this paper, we directly utilize the front camera of the mobile phone to collect the diverse gesture data, exhibiting changed speeds, different illumination conditions, and various scenes. Besides, we provide more fine-grained annotation to benefit algorithm design.

3. Methodology

In this section, we will describe the proposed online lightweight two-stage framework for detecting and classifying dynamic gestures in untrimmed videos in detail. The flowchart of this framework is shown in Fig. 2. After obtaining the video stream from the camera, the whole analysis procedure can be summarized as follows:

- Detection module: We first introduce a motion detection network, *i.e.*, MotionNet, to determine whether there is a gesture at present in the raw video stream. Furthermore, to improve the reliability of the system, we propose a post-process that smooths and filters the output of the MotionNet. After that, the gesture clips will be sent to the cascade gesture recognition network.
- Gesture recognition module: We employ a temporal relational reasoning network, *i.e.*, GestureNet to process the cleaned gesture sequences and identify the specific gesture type. Besides, we design a state machine to process the results from the gesture recognition network so that each complete gesture corresponds to a single-time prediction.

3.1. Detection Module

For real-time gesture recognition on untrimmed videos, the basic idea of these methods is to slide over a video sequence with a certain step and window size and perform gesture classification on these window segments. However, clips at the beginning and end of the action are prone to errors. It also causes heavy resource consumption due to the long-time and high-load operation. Therefore, we design a low-power motion detection module as the state converter of the gesture recognition module. Thus the gesture recognition module will only be enabled if a gesture is detected. The detection module consists of the MotionNet and the post-process to distinguish whether there are gestures in the raw video stream.

As a real-time system, we expect the MotionNet: 1) has acceptable accuracy to ensure the performance of the entire system; 2) is a lightweight network to ensure that it can run for a long time with low power consumption; 3) has the fast speed to avoid interaction delay. Although the optical flows and 3D CNNs perform well in capturing motion information, the former is complex and slow to compute, and the latter has vast parameters and a large model. Hence, we propose a lightweight network based on 2D CNNs for gesture motion detection. The original video data is RGB image sequences at 30 FPS captured from the camera, which are then sampled at 15 FPS. A raw sequence is obtained by using a sliding window with length 4 and stride 2 on the input video. To encoder appearance and motion features simultaneously, we convert the four frames of RGB images into a combination of RGB images and differential images, which are then sent into the MotionNet.

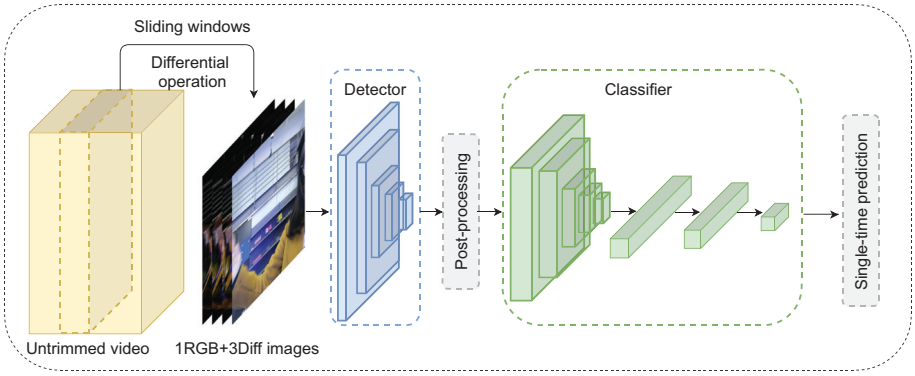

Fig. 2. Flowchart of our proposed framework. The differential operation means we preprocess the ordered frames obtained by the sliding window instead of directly processing the original frames.

In our method, we regard the gesture motion detection task as a binary classification task. The difference is that the input of a general classification task is a single image, but a sequence is processed here. There are many excellent image classification networks, such as VGG [44], ResNet [45], and the family of Inception [46–49]. However, these methods are unsuitable for real-time systems because of heavy computation. We select Mobilenetv2 [50] for its depthwise separable convolution design and change it to make the network more lightweight. Specifically, as shown in Fig. 3, the standard convolution operation is performed simultaneously in the two dimensions of space and channel, but the depthwise separable convolution splits the convolution operation into two layers. Depthwise convolution uses a single convolution filter to compute per input channel, while pointwise convolution computes linear combinations of the input channels. The computation amount of the depthwise separable convolution is reduced by about K^2 times than the standard one when kernel size is set K. The architecture of the MotionNet used in this work is shown in Table 1. (See Fig. 4).

In practice, we observe some critical issues. First, the uncertainty of obtaining dynamic gestures in untrimmed videos will lead to the loss of some frames of a complete gesture. Second, for the existence of the intermediate transition action, the detector usually makes a misjudgment. Third, it may repeatedly detect a specific gesture if the length of the sliding window does not cover the entire gesture.

To address the above three problems, we propose to filter the gesture motion detection results in pursuit of the reliability of the entire system. During the online detection, the results of the MotionNet are further processed. Specifically, a double-ended queue is set up to save the four historical states from t_1 to t_8 at the current time. We apply this queue to correct the current result. The specific rules are as follows: 1) The state of the first position in the queue must be gestures; 2) Among all states in the current queue, at least one result is judged to be a gesture; 3) In the final queue obtained by filtering, it is guaranteed that no two consecutive results are judged as gestures. If the above rules are satisfied simultaneously, it is determined that there is a dynamic gesture in the new motion detection result. This process is the key to ensuring the accuracy of the whole real-time system.

3.2. Gesture Recognition Module

When the motion detection module detects a gesture, it will activate the gesture recognition module. The gesture recognition module consists of the GestureNet and the single-time filter. Generally, to obtain temporal relations from video sequences, most recent methods adopt 3D CNNs, but they struggle to model the long-term temporal cues in untrimmed videos. Besides, the successive frames usually show redundant information in spatial and temporal dimension. Inspired by the Temporal Relation Network (TRN) [51] that uses a sparse sampling strategy and deduces the short- and long-term temporal relations, we propose a lightweight relation reasoning network, termed GestureNet, to identify the specific type of gestures from video clips. The differences between our GestureNet and TRN are: (1) The feature extraction layer of TRN is replaced with the Mobilenet-v2 to reduce the parameters. (2) TRN calculates frame relationships, while we only use 3-frames and 7-frames relationships to reduce the amount of computation in our system. Our design makes it possible to reason temporal relationships at multiple time scales and satisfies the real-time application. The architecture of the basic model in GestureNet is shown in Table 2. Formally, the temporal relationship between the two frames is defined as below:

$$TR_2(V) = FC_\beta\left(\sum_{i<j} FC_\theta(f_i, f_j)\right) \tag{1}$$

where the input video $V = \{f_1, f_2, f_3, \cdots, f_n\}$contains n sampled frames, where f_i represents the features of the i-th original video frame extracted by the backbone layer. The function FC_θ is a two-layer MLP with 256 units per layer, while FC_β is a one-layer MLP with the unit number matching the class number. These two layers are shared within each scale. They are used to fuse the relationships of different ordered frames. Similarly, the temporal relationship between the three frames is defined below:

$$TR_3(V) = FC'_\beta\left(\sum_{i<j<k} FC'_\theta(f_i, f_j, f_k)\right) \tag{2}$$

According to the formula defined above, we can easily get higher frame relationships. In order to obtain the temporal relationships at multiple time scales, the fusion function is defined below:

$$MTR_N(V) = TR_2(V) + TR_3(V) \ldots + TR_N(V) \tag{3}$$

Specifically, the GestureNet is fed by the video sequences filtered through the detection module. We sample 8 frames to calculate temporal relationships at multiple time scales, *i.e.*, from 2-frames up to 8-Frames. For example, in the 3-frames temporal relationships, there are 56 combinations. If all ordered pairs are analyzed, the amount of computation will be too heavy to be accepted by our framework. In this paper, we choose 3-frames and 7-frames to balance accuracy and computation.

In practice, we expect that a complete gesture corresponds to a single-time prediction in the real-time gesture recognition system. Besides, different types of gestures may contain similar preparation and retraction, which can easily lead to ambiguity at the

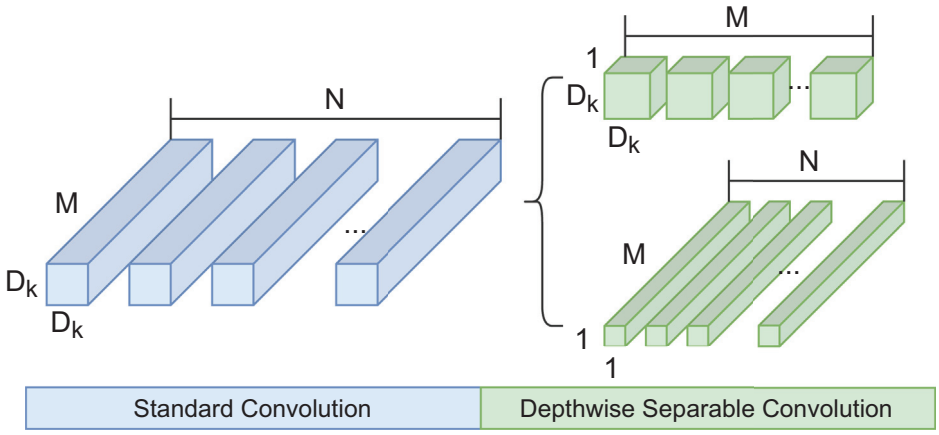

Fig. 3. Comparison of standard convolution and depthwise separable convolutions, where M, N are the input and output channels, D_k is the kernel size. Splitting the standard convolution into two sub-parts is helpful in reducing the amount of calculation.

Table 1
Model architecture for MotionNet. Unless otherwise stated, all spatial convolutions use 3×3 kernels. The expansion ratio is the ratio between the size of the input bottleneck and the latent size. The channel is the number of filters. The bottleneck is shown in Fig. 4.

Input	Layer	Channel	Stride	Expansion ratio
56×56×12	conv2d	32	2	-
28×28×32	bottleneck	16	1	1
28×28×16	bottleneck×2	24	2/1	1.5
14×14×24	bottleneck×3	32	2/1/1	1.5
7×7×32	bottleneck×4	64	1/1/1/1	1.5
7×7×64	bottleneck×3	96	2/1/1	1.5
4×4×96	bottleneck×3	160	2/1/1	1.5
2×2×160	bottleneck	320	1	1.5
2×2×320	conv2d 1×1	512	1	1.5
2×2×512	avgpool 2×2	-	-	-

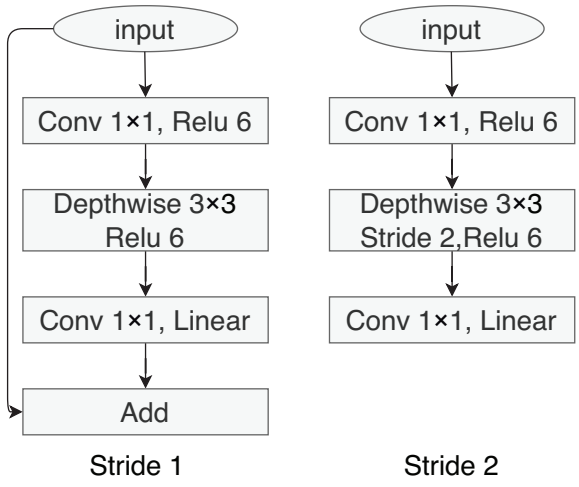

Fig. 4. The details of the bottleneck in Table 1.

beginning and end of the gesture. Meanwhile, we also do not want to receive multiple unstable outputs during a long-term gesture performing. Considering the above issues, we set up a state machine to process the results of the gesture recognition network, which judges the next state based on the current and historical states to return the final recognition result. Our method initializes a queue with a fixed length of 3 for storing and updating the result, confidence, and timestamp of the classifier's output. We update the result R, confidence $Conf$, and timestamp T_s to the end of the queue. If the gap between the timestamp of the previous state and the current is greater than 1 s, the result is regarded as the current state $(R^*, Conf^*, T_s^*)$, otherwise, we will traverse all elements of the queue, the highest confidence is regarded as the current state. After obtaining the current state, its confidence is greater than the predefined threshold and is different from the previous state, then R^* will be a final result; otherwise, it will not be output.

4. ZJUGesture Dataset

This section provides the details of the ZJUGesture dataset. This dataset mainly focuses more on solving one-handed operation scenarios in practice, such as mobile phones and tablets. Our dataset defines nine common categories of gestures that are easy for users to operate: *no gesture*, *swipe left*, *swipe right*, *push*, *turn clockwise*, *turn counterclockwise*, *palm to fist*, *fist to palm*, and *fold up*. In order to ensure the diversity of data samples, we collected each action in 12 sub-scenes, including different backgrounds and light intensity. Fig. 5 shows some of the data in the ZJUGesture dataset. The video data is collected by 60 identities at a resolution of 1280×720, and 30 FPS.

According to the training requirements of the model, there are two steps to label the gesture dataset. The first step is to label all videos frame by frame whether they contain gestures or not. Secondly, the video clips with gestures are labeled with a specific gesture category. Nine types of gesture data correspond to the label 0–8. To ensure the effectiveness of the experiment, our training set and test set are strictly divided according to the principle that the same person cannot appear in both subsets simultaneously. In order to ensure the quality of the data, we hire 20 professional employees to label each original video frame by frame. During the labeling process, 1 or 2 frames will be skipped between each adjacent action.

Compared with the existing datasets, ZJUGesture maintains the same collection conditions, such as distinct subjects, complex scenarios, and different illumination, but embodies three different properties, *i.e.*, considering the transitional movements, focusing on hand movements, and containing different action speeds. Specifically, we found that the transition actions of each gesture are close. If the complete gestures are retained in the dataset without distinction, it will cause frequent error responses when processing untrimmed videos in real systems. Consequently, we divide a gesture into three parts: preparation, core action and retraction. Second, the collection distance is far in current datasets, resulting in the noise of arm and body actions, which does not sat-

Table 2
Basic model architecture for gesture recognition. The meaning of the parameters is the same as described in Table 1.

Input	Layer	Channel	Stride	Expansion ratio
224×224×24	conv2d	32	2	-
112×112×32	bottleneck	16	1	1
112×112×16	bottleneck×2	24	2/1	1.5
56×56×24	bottleneck×2	32	2/1	1.5
28×28×32	bottleneck×3	64	1/1/1	1.5
28×28×64	bottleneck×2	96	2/1	1.5
14×14×96	bottleneck×2	160	2/1	1.5
7×7×160	bottleneck	320	1	1.5
7×7×320	conv2d 1×1	512	1	1.5
7×7×512	avgpool 7×7	-	-	-

isfy the human–computer interaction requirements of handheld devices with relatively small view fields. In contrast, we choose to directly use the front camera of the mobile phone to collect gesture data and pay more attention to hand movements. Third, the speeds of the gestures in the dataset must be different, even for the same action category. In ZJUGesture dataset, 60 persons are asked to perform each gesture according to their habits in three levels, *i.e.*, fast, normal, and slow. In summary, our dataset has dramatically diversified properties.

5. Experiments and Discussions

5.1. Details of Data and Experiments

In this paper, we evaluate our framework on Jester[18] and ZJUGesture. Jester is the standard publicly available dataset for gesture recognition, which is collected by crowd workers performing 27 kinds of gestures in front of a laptop camera or webcam, consisting of three parts, 118,562 for training, 14,787 for validation, and 14,743 for testing. The images in Jester are extracted from the videos at 12 FPS. For ZJUGesture, we first sample 30 original videos that contain multiple gestures within a continuous sequence for the detection module experiments. We label videos frame by frame into *gestures* and *no gestures*, obtaining the MotionData. The sample numbers of the training and test sets are given in Table 3. For gesture recognition experiments, we carefully annotate the original collected videos into nine categories. The statistics of gesture instances in the training and test sets are given in Table 4. We also report the gesture duration, which is described in average, maximum, and minimum. It can be seen that the duration of gestures both in inter- and intra-class are significantly different. The difference between the longest and shortest clips can reach 103 frames, or 91 frames for the same gesture class, which brings great challenges to gesture recognition. Besides, the number of frames in the preparation and retraction stages are relatively close.

5.2. Experiments for Detection Module

Since the motion detection module is always running in our framework, we need to consider the trade-off between accuracy and the consumption of time and space. We first compare the performance of our MotionNet with other popular approaches on the MotionData. In this experiment, we first resize the video frames from 1280×720 resolution to 56×56 to avoid the heavy computational burden. Here we stack the continuous four frames covered by the sliding window, obtaining the input with 56×56×12. In our MotionNet, we convert the input of four RGB frames into one RGB image and three differential images to provide appearance and motion information. The models are trained from scratch with Kaiming initialization. Table 5 shows the results of the MotionData test set. We find that the MotionNet performs better than VGG16 [44], Resnet50 [45], and TRN [51] with 4-frames relationships on metrics of accuracy, model size, and counting FLOPs. The third metric means floating point operations, which are used to measure the complexity of the algorithm. The FLOPs and model size of our detector are much smaller than other models to ensure that it is lightweight. Besides, our method with differential images has higher accuracy than the original Mobilenet-v2 without additional computing resources, which shows the effectiveness of our method.

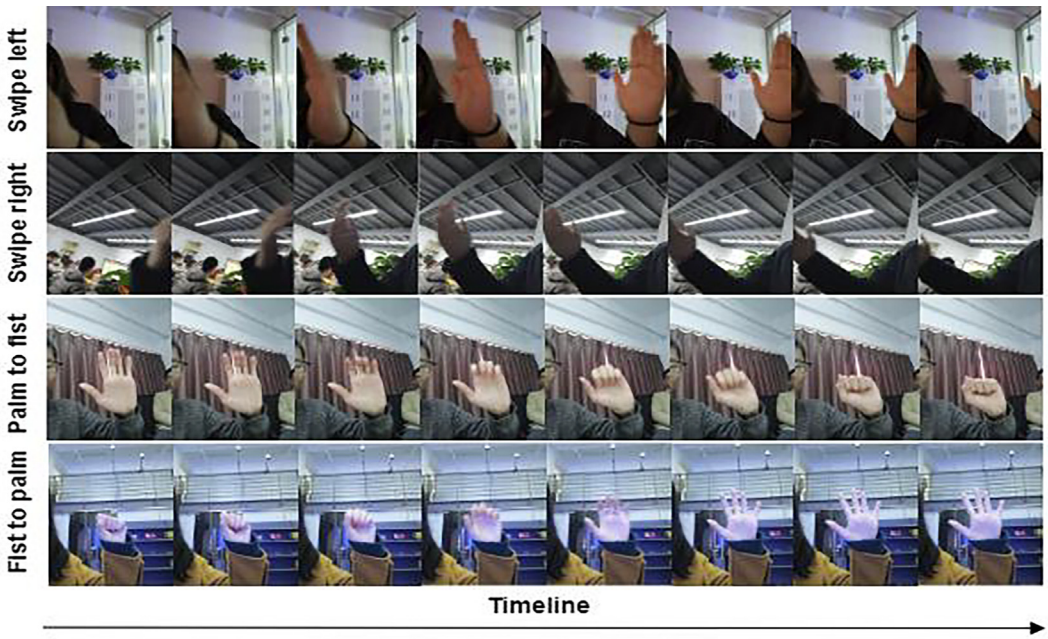

Fig. 5. Some data examples in the ZJUGesture dataset. The categories of gestures from top to bottom are *swipe left*, *swipe right*, *palm to fist*, and *fist to palm*.

Table 3
Statistics for MotionData in detection module experiments.

Category	Instances		
	Total	Train	Test
gesture	2920	2000	920
no gesture	2255	1825	430

In order to further analyze the MotionNet, we compare MotionNet with other lightweight networks on the Jester subset. Specifically, the experiment utilizes 17,664 videos in the Jester data set, of which 15,706 were used as training samples, and 1,958 were used as test samples. The results are shown in Table 6. We observe that our method exhibits superior performance over VGG16, Mobilenet-v2, and Shufflenet [52] both in model accuracy and the amount of computation. Thus, we can draw the conclusion that our detector is lightweight and high accuracy, satisfying our requirements for deployment in real-time systems.

Then, we study the performance of the MotionNet under different input forms and sizes to achieve the trade-off between accuracy and resource consumption. The performance comparison is shown in Table 7. Although increasing the resolution of the input image can bring a little performance improvement, the amount of computation has increased a lot. Experimentally, we choose the input with a size of 56×56. Besides, we present four variants, four RGB frames, three RGB frames and 1 differential image calculated from pair 1–2 frames, two RGB frames and 2 differential images calculated from pairs 1–2 and 1–3 frames, one RGB frame and three differential images calculated from pairs 1–2, 1–3, and 1–4 frames. From Table 7, we conclude that the concatenation of differential images helps capture motion information, and combining low- and high-ordered motion cues will further improve performance.

Thirdly, we design an experiment to study the effect of the intervals of differential images on performance. We fix the input size as 56 and choose the format of three RGB images and one differential image. The distance to calculate the differential image is 1, 2, 3. Table 8 illustrates that the differential image with a large distance shows the best performance, which reveals that adjacent frames contain less important motion information than that with larger intervals.

5.3. Experiments for Gesture Recognition Module

We evaluate the gesture recognition models on ZJUGesture and Jester in terms of accuracy and computation, *i.e.*, top-1 accuracy, top-5 accuracy, total parameters of the model, and counting FLOPs, to prove that our model is applicable in real-time systems.

First of all, we compare our GestureNet with SOTA methods in the field of action recognition on the test set of the ZJUGesture dataset. Here, all models are trained from scratch with Kaiming initialization. We sample 8 fixed-length RGB images from the video and resize the resolution of the images to 224×224, which are then stacked along the channel dimension, obtaining the input size of 224×224×24. Notably, the duration of each gesture is different, resulting in an uncertain number of video frames per sample during the training, we propose a time scale normalization method to enhance the adaptability of the network to the inconsistency of different time scales. Specifically, we set the length of the window according to different video lengths and then randomly generate offsets within the video range. The window location plus the offset is the final sampling result. As shown in the upper part of the Table 9, our method achieves competitive performance in accuracy and lower computation consumption than common action recognition methods, C3D [32], P3D [33], MSTRN [51], and TSM [53]. We further compare the proposed method with a recent real-time gesture recognition method with officially released codes. From the bottom part of the Table 9, our method shows a comparable amount of calculation with RGDC [34] but outperforms a large margin on recognition accuracy than it. Therefore, it is more suitable for our method to be applied to the actual gesture recognition system than the existing methods.

We further evaluate the gesture recognition models on the test and valid set of Jester. The results are listed in Table 10 and Table 11, respectively. Similar to the results of the ZJUGesture dataset, common action recognition methods perform pretty well in accuracy, but require the complex model and heavy calculation. Another real-time method RGDC has low GFLOPs and model size but suffer from the sharp drop in accuracy. In contrast, our method exhibits trade-off performance on accuracy and computation,

Table 5
Comparison of our MotionNet to the state-of-the-art methods on the test set of MotionData. **Bold** and underline represent optimal and suboptimal results. The up arrow indicates that the larger the value, the better the model performance, and vice versa.

Model	Top-1 acc(%) ↑	Size(M) ↓	GFLOPs ↓
VGG16	60.3	285.7	1.381
ResNet-50	89.9	297.3	0.660
TRN (4 frames)	88.2	40.1	0.084
Mobilenet-v2	90.2	17.6	0.027
Ours	**91.7**	**17.6**	**0.027**

Table 6
Comparison of our MotionNet to the state-of-the-art methods on the Jester dataset.

Model	Top-1 acc(%) ↑	GFLOPs ↓
VGG16	95.43	1.381
Mobilenet-v2	95.85	0.027
ShuffleNet (x2)	95.64	0.398
ShuffleNet (1.5)	94.13	0.231
Ours	**96.30**	**0.027**

Table 4
Statistics for ZJUGesture dataset in gesture recognition experiments. AP. and AR. mean the average number of preparation and retraction stages.

Category	Instances			Duration			AP.	AR.
	Total	Train	Test	Avg.	Max.	Min.		
no gesture	2255	1965	290	17	48	9	-	-
swipe left	1110	934	176	21	52	10	3	4
swipe right	1031	867	164	22	56	12	3	5
push	807	648	159	20	67	9	5	6
clockwise	1061	864	197	39	108	17	3	7
counterclockwise	1089	878	211	41	104	20	4	6
palm to fist	862	754	108	13	42	5	3	5
fist to palm	918	790	128	16	70	6	5	6
fold up	759	590	169	36	42	19	6	4

Table 7
Comparison of the MotionNet performance of different input forms and sizes on the test set of MotionData. Diff means differential images.

Input form	Input size	Top-1 acc(%) ↑	Size(M) ↓	GFLOPs ↓
4 RGB	224×224	92.3	17.6	0.442
3 RGB + 1 Diff		92.8		
2 RGB + 2 Diff		93.2		
1 RGB + 3 Diff		**93.4**		
4 RGB	56×56	90.2	17.6	0.027
3 RGB + 1 Diff		91.3		
2 RGB + 2 Diff		91.6		
1 RGB + 3 Diff		**91.7**		

Table 8
Comparison of the MotionNet performance of different distances on the test set of MotionData.

Input form	Distance	Top-1 acc(%) ↑
3 RGB + 1 Diff	1	91.3
	2	91.4
	3	**91.7**

which is more suitable and compact for real-time application. Notably, GestureNet has a significant performance advantage on the ZJUGesture. For example, the accuracy improvements compared with P3D63 are 9.04 and 0.67 on ZJUGesture and Jester, respectively. The reason is that methods like P3D63, which struggle to capture long-term temporal relationships, usually fail to distinguish the same parts of different gestures, such as swipe left, swipe right, and turn clockwise have many similar parts. Our dataset only contains high-frequency daily-used gestures and magnifies the shortcomings of these methods, resulting in performance degradation.

Finally, we evaluate our method with various frame relation modules on ZJUGesture. As shown in Table 12, more additional frames included in the relation bring an obvious boost. To consider the low- and high-ordered relation, we combine two relation modules, *e.g.*, 2-frames and 8-frames, and achieves further improvements. Our method with all scale relations shows the best performance. As for the real-time system, we adopt 3-frames and 7-frames relationship to balance the performance and computation.

5.4. Experiments for Untrimmed Videos Online

In this experiment, we adopt the average Levenshtein accuracy metric [34] to further evaluate the system's misclassification, multiple detection, and missing detection online. Levenshtein distance measures the distance between sequences by counting the number of item-level operations (inserts, deletions, or replacements) that convert one sequence to another. Formally:

$$Levenshtein\ accuracy = \left(1 - \frac{Levenshtein\ distance}{Number\ of\ categories}\right) \tag{4}$$

Unlike the offline test, this experiment simulates the actual system processing untrimmed videos directly from cameras online. Here, we select 153 untrimmed videos from ZJUGesture, 15 gestures per video on average. The sliding window size of the detector is 8 and 16 for the classifier. The clips are down-sampled to 4 and 8 frames for networks. The experimental results are shown in Table 13. The first row indicates that if the classifier directly processes the video streams without motion detector, the results are prone to be *no gesture*, which severely affects gesture identification. Besides, the computational cost of the classifier is much higher than that of the detector, which will greatly reduce the speed when performing uninterruptedly. The second row illustrates that if there is no post-processing, although the speed is improving significantly, the

Table 9
Comparison of our GestureNet to the state-of-the-art methods on the test set of ZJUGesture dataset.

Model	Top-1 acc(%) ↑	Top-5 acc ↑	GFLOPs ↓	Parameters(M) ↓
C3D	79.62	97.69	129.9	99
P3D199	75.83	97.88	35.5	66
P3D131	81.54	97.88	24.6	47
P3D63	81.35	97.69	15.9	25
MSTRN	**92.15**	**99.75**	16.4	13
TSM	91.02	99.70	23.7	24
RGDC	83.71	98.89	2.3	**6**
Ours	90.39	99.74	**2.0**	10

Table 10
Comparison of our GestureNet to the state-of-the-art methods on the test set of Jester dataset.

Model	Top-1 acc(%) ↑	GFLOPs ↓	Parameters(M) ↓
MSTRN	**94.78**	16.4	13
P3D63	91.19	15.9	25
TSM	92.29	23.7	24
RGDC	82.30	2.3	**6**
Ours	91.22	**2.0**	10

Table 11
Comparison of our GestureNet to the state-of-the-art methods on the valid set of Jester dataset.

Model	Top-1 acc(%) ↑	Top-5 acc ↑	GFLOPs ↓	Parameters(M) ↓
MSTRN	**93.70**	99.59	16.4	13
P3D63	91.13	98.99	15.9	25
TSM	93.58	**99.68**	23.7	24
RGDC	83.92	97.64	2.3	**6**
Ours	91.80	99.06	**2.0**	10

Table 12
The results of different frame-relation on ZJUGesture dataset. R-N means the relationship between N frames.

Model	Top-1 acc(%) ↑	GFLOPs ↓	Model	Top-1 acc ↑	GFLOPs↓
R-2	82.79	0.6	R-7	88.62	1.4
R-3	84.61	0.8	R-2–8	90.36	1.9
R-4	86.58	1.2	R-3–7	90.39	2.0
R-5	87.17	1.3	R-4–6	90.41	2.6
R-6	88.09	1.4	R-All	92.10	12.1

Table 13
Online results in untrimmed videos selected frome ZJUGesture dataset.

Method	Levenshtein accuracy(%) ↑	Speed(seg/s) ↑
No MotionNet	87.2	55
No post-processing	70.9	**186**
No single-time filter	80.2	176
Ours	**92.4**	173

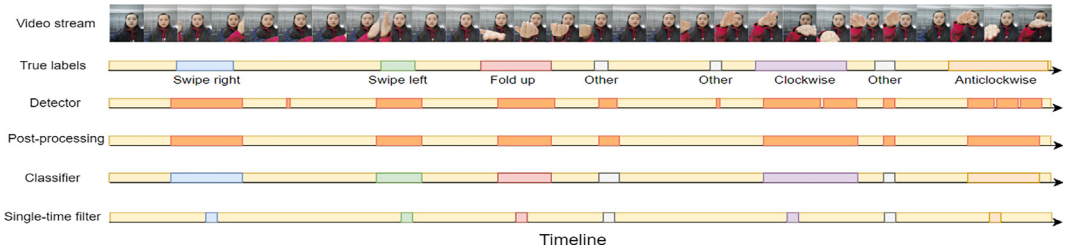

Fig. 6. Visualize the results of each step of the entire framework. *Other* represents interference gestures. Our method locates the moment when the user performs the gesture in the video stream and achieves a single-time prediction. And the activation often occurs in the first half of the gesture, reducing the response time.

Table 14
Comparison of our framework to the state-of-the-art methods with MobileNet-v2 backbone in untrimmed videos.

Model	Levenshtein accuracy(%) ↑	Speed(seg/s) ↑
C3D	76.1	28
P3D	78.2	57
MSTRN	72.4	145
Ours	**92.4**	**173**

Levenshtein accuracy decreases obviously. This phenomenon attributes to the misrecognition occur in the neural network and the loss of field of view in the real-time system, thus the post-processing is critical to correct the results. Furthermore, comparing rows 3 and 4, it is necessary to filter the recognition results when processing long-term gestures. The results of each step of our framework are visualized in Fig. 6. Moreover, in Table 14, we change the backbone of C3D, P3D, and MSTRN to MobileNet-v2. It is obvious that directly sending the frames covered by the sliding window to these methods will result in a loss of accuracy and efficiency, demonstrating the effectiveness of the combination of detector and classifier. In summary, our framework is both lightweight and efficient when dealing with untrimmed videos directly from cameras online.

6. Conclusions

In this paper, we proposed an online lightweight two-stage framework to detect and classify dynamic gestures in untrimmed video streams. The proposed framework can quickly return a single video-level prediction for a complete gesture. Specifically, we first design a lightweight detection module that introduces extra motion information by using differential images and a post-process to reduce misrecognition. Then, a followed lightweight gesture recognition module predicts gesture categories from the RGB images with gestures detected by the detection module and a filter to ensure a single-time response. Furthermore, we propose a more complex and fine-grained annotated gesture recognition dataset ZJUGesture. The experiments conducted on Jester and ZJUGesture datasets demonstrate our lightweight framework with competitive accuracy using only images. Our approach has been deployed to the mobile for use online, which verifies the effectiveness of the whole framework.

Data availability

Data will be made available on request.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work is supported by the National Key R&D Program of China under Grant (2019YFB1311600) and the Key R&D Program Project of Zhejiang Province (2021C01035).

References

[1] A. Pandit, D. Dand, S. Mehta, S. Sabesan, A. Daftery, A simple wearable hand gesture recognition device using imems, in: 2009 International Conference of Soft Computing and Pattern Recognition, IEEE, 2009, pp. 592–597.
[2] K.S. Abhishek, L.C.K. Qubeley, D. Ho, Glove-based hand gesture recognition sign language translator using capacitive touch sensor, in: 2016 IEEE International Conference on Electron Devices and Solid-State Circuits (EDSSC), IEEE, 2016, pp. 334–337.
[3] P. Kumar, J. Verma, S. Prasad, Hand data glove: a wearable real-time device for human-computer interaction, International Journal of Advanced Science and Technology 43.
[4] H. Kenn, F. Van Megen, R. Sugar, A glove-based gesture interface for wearable computing applications, in: 4th International Forum on Applied Wearable Computing 2007, VDE, 2007, pp. 1–10.
[5] K. Kalgaonkar, B. Raj, One-handed gesture recognition using ultrasonic doppler sonar, in: 2009 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2009, pp. 1889–1892.
[6] Y. Li, Hand gesture recognition using kinect, in: 2012 IEEE International Conference on Computer Science and Automation Engineering, IEEE, 2012, pp. 196–199.
[7] P. Molchanov, S. Gupta, K. Kim, J. Kautz, Hand gesture recognition with 3d convolutional neural networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2015, pp. 1–7.
[8] N. Neverova, C. Wolf, G.W. Taylor, F. Nebout, Multi-scale deep learning for gesture detection and localization, in: European Conference on Computer Vision, Springer, 2014, pp. 474–490.
[9] P. Molchanov, S. Gupta, K. Kim, K. Pulli, Multi-sensor system for driver's hand-gesture recognition, 2015 11th IEEE international conference and workshops on automatic face and gesture recognition (FG), Vol. 1, IEEE, 2015, pp. 1–8.
[10] E. Ohn-Bar, M.M. Trivedi, Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations, IEEE transactions on intelligent transportation systems 15 (6) (2014) 2368–2377.
[11] H. Tang, H. Liu, W. Xiao, N. Sebe, Fast and robust dynamic hand gesture recognition via key frames extraction and feature fusion, Neurocomputing 331 (2019) 424–433.
[12] Z. Hu, Y. Hu, J. Liu, B. Wu, D. Han, T. Kurfess, 3d separable convolutional neural network for dynamic hand gesture recognition, Neurocomputing 318 (2018) 151–161.
[13] Z. Cao, X. Xu, B. Hu, M. Zhou, Q. Li, Real-time gesture recognition based on feature recalibration network with multi-scale information, Neurocomputing 347 (2019) 119–130.
[14] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, T. Serre, Hmdb: a large video database for human motion recognition, in: 2011 International Conference on Computer Vision, IEEE, 2011, pp. 2556–2563.
[15] K. Soomro, A.R. Zamir, M. Shah, Ucf101: A dataset of 101 human actions classes from videos in the wild, arXiv preprint arXiv:1212.0402.
[16] Y. Zhang, C. Cao, J. Cheng, H. Lu, Egogesture: a new dataset and benchmark for egocentric hand gesture recognition, IEEE Transactions on Multimedia 20 (5) (2018) 1038–1050.
[17] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, J. Kautz, Online detection and classification of dynamic hand gestures with recurrent 3d convolutional neural network, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 4207–4215.
[18] J. Materzynska, G. Berger, I. Bax, R. Memisevic, The jester dataset: A large-scale video dataset of human gestures, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2019, pp. 0–0.
[19] C.R. Wren, A. Azarbayejani, T. Darrell, A.P. Pentland, Pfinder: Real-time tracking of the human body, IEEE Transactions on pattern analysis and machine intelligence 19 (7) (1997) 780–785.

[20] D. Fleet, Y. Weiss, Optical flow estimation, in: Handbook of mathematical models in computer vision, Springer, 2006, pp. 237–257.
[21] M. Danelljan, G. Bhat, F. Shahbaz Khan, M. Felsberg, Eco: Efficient convolution operators for tracking, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 6638–6646.
[22] M. Danelljan, A. Robinson, F.S. Khan, M. Felsberg, Beyond correlation filters: Learning continuous convolution operators for visual tracking, in: European conference on computer vision, Springer, 2016, pp. 472–488.
[23] B. Kaufmann, J. Louchet, E. Lutton, Hand posture recognition using real-time artificial evolution, in: European Conference on the Applications of Evolutionary Computation, Springer, 2010, pp. 251–260.
[24] C. Weng, Y. Li, M. Zhang, K. Guo, X. Tang, Z. Pan, Robust hand posture recognition integrating multi-cue hand tracking, in: International Conference on Technologies for E-learning and Digital Entertainment, Springer, 2010, pp. 497–508.
[25] M. Flasiński, S. Myśliński, On the use of graph parsing for recognition of isolated hand postures of polish sign language, Pattern Recognition 43 (6) (2010) 2249–2264.
[26] J. Carreira, A. Zisserman, Quo vadis, action recognition? a new model and the kinetics dataset, in: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6299–6308.
[27] P. Narayana, R. Beveridge, B.A. Draper, Gesture recognition: Focus on the hands, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 5235–5244.
[28] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: Advances in neural information processing systems, 2014, pp. 568–576.
[29] B. Zhang, L. Wang, Z. Wang, Y. Qiao, H. Wang, Real-time action recognition with enhanced motion vector cnns, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2718–2726.
[30] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, T. Darrell, Long-term recurrent convolutional networks for visual recognition and description, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 2625–2634.
[31] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.
[32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, M. Paluri, Learning spatiotemporal features with 3d convolutional networks, in: Proceedings of the IEEE international conference on computer vision, 2015, pp. 4489–4497.
[33] Z. Qiu, T. Yao, T. Mei, Learning spatio-temporal representation with pseudo-3d residual networks, in: proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 5533–5541.
[34] O. Köpüklü, A. Gunduz, N. Kose, G. Rigoll, Real-time hand gesture detection and classification using convolutional neural networks, in: 2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019), IEEE, 2019, pp. 1–8.
[35] Q. Miao, Y. Li, W. Ouyang, Z. Ma, X. Xu, W. Shi, X. Cao, Multimodal gesture recognition based on the resc3d network, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 3047–3055.
[36] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, M. Paluri, A closer look at spatiotemporal convolutions for action recognition, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2018, pp. 6450–6459.
[37] C. Xu, X. Wu, Y. Li, Y. Jin, M. Wang, Y. Liu, Cross-modality online distillation for multi-view action recognition, Neurocomputing 456 (2021) 384–393.
[38] C. Yang, Y. Xu, J. Shi, B. Dai, B. Zhou, Temporal pyramid network for action recognition, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 591–600.
[39] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, L. Van Gool, Temporal segment networks: Towards good practices for deep action recognition, in: European conference on computer vision, Springer, 2016, pp. 20–36.
[40] Y. Zhang, L. Shi, Y. Wu, K. Cheng, J. Cheng, H. Lu, Gesture recognition based on deep deformable 3d convolutional neural networks, Pattern Recognition 107 (2020).
[41] G. Sung, K. Sokal, E. Uboweja, V. Bazarevsky, J. Baccash, E.G. Bazavan, C.-L. Chang, M. Grundmann, On-device real-time hand gesture recognition, arXiv preprint arXiv:2111.00038.
[42] G. Benitez-Garcia, J. Olivares-Mercado, G. Sanchez-Perez, K. Yanai, Ipn hand: A video dataset and benchmark for real-time continuous hand gesture recognition, in: 2020 25th International Conference on Pattern Recognition (ICPR), IEEE, 2021, pp. 4340–4347.
[43] J. Wan, Y. Zhao, S. Zhou, I. Guyon, S. Escalera, S.Z. Li, Chalearn looking at people rgb-d isolated and continuous datasets for gesture recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2016, pp. 56–64.
[44] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv preprint arXiv:1409.1556.
[45] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
[46] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1–9.
[47] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, arXiv preprint arXiv:1502.03167.
[48] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, Rethinking the inception architecture for computer vision, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2818–2826.
[49] C. Szegedy, S. Ioffe, V. Vanhoucke, A.A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Thirty-first AAAI conference on artificial intelligence, 2017.
[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, L.-C. Chen, Mobilenetv 2: Inverted residuals and linear bottlenecks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 4510–4520.
[51] B. Zhou, A. Andonian, A. Oliva, A. Torralba, Temporal relational reasoning in videos, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 803–818.
[52] X. Zhang, X. Zhou, M. Lin, J. Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices.
[53] J. Lin, C. Gan, S. Han, Tsm: Temporal shift module for efficient video understanding, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 7083–7093.

Chao Xu received the B.S. degree in electrical engineering and its automation from Nanchang University, Nanchang, China, in 2018.
He is currently working toward the doctor degree in control science and engineering with the School of Control Science and Engineering, Zhejiang University, Hangzhou, China.
His major research interests include video understanding and video generation.

Xia Wu received the B.S. degree in electronic information from China Agricultural University,Beijing, China, in 2018. She received the M.S. degree in control science and engineering with the School of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2021.
Her major research interests include computer vision and temporal action recognition.

Mengmeng Wang received the B.S. degree and M.S. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2015 and 2018.
She is currently working toward the doctor degree in control science and engineering with the School of Control Science and Engineering, Zhejiang University, Hangzhou, China.
Her major research interests include computer vision and deep learning.

Feng Qiu received the B.S. degree from Harbin Institute of Technology, Harbin, China, in 2017. he received the M.S. degree in control science and engineering with the School of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2020.
His major research interests include computer vision and temporal action recognition.

Yong Liu received the B.S. degree in computer science and engineering and the Ph.D degree in computer science from Zhejiang University, Zhejiang, China, in 2001 and 2007, respectively. He is currently a professor of Institute of Cyber-Systems and Control at Zhejiang University.

His main research interests include: intelligent robot systems, robot perception and vision, deep learning, big data analysis, and multi-sensor fusion. He has published over 30 research papers on machine learning, computer vision, information fusion, and robotics.

Jun Ren received the M.S. degree from Beijing Institute of Technology, Beijing, China, in 2013. Her main research interests include artificial Intelligence and computer vision, with particular interests in object detection.