Received December 30, 2021, accepted January 12, 2022, date of publication January 18, 2022, date of current version February 8, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3144035

An Improved Action Recognition Network With Temporal Extraction and Feature Enhancement

JIE JIANG AND YI ZHANG
Department of Computer Science, Sichuan University, Chengdu 610065, China
Corresponding author: Yi Zhang (yi.zhang@scu.edu.cn)

This work was supported by the Program of NSFC under Grant U20A20161.

ABSTRACT Image classification and action recognition are both active research topics in the field of computer vision. However, the development of action recognition is rather slow compared with image classification, due to the difficulties in spatial-temporal information modeling. In this paper, we present TEFE, a deep structure combining temporal extraction with feature enhancement to explore the spatial coherence across temporal dimension. The temporal extraction (TE) module is used to capture the short-term and long-term temporal features. Considering the spatial and temporal cues are fine-grained, we believe such cues (if encoded by well-designed bilinear models) could enhance the representation of actions in videos. The feature enhancement (FE) module approximates bilinear pooling operations, which greatly reduce the amount of parameters exist in original bilinear pooling. Extensive experiments have been conducted on mainstream datasets of human action (Something - Something V1 & V2, and Jester). Experimental results show that our model achieves competitive performances than some existing methods.

INDEX TERMS Action recognition, spatial-temporal information modelling, feature enhancement.

I. INTRODUCTION

Human action recognition from videos has attracted wide attention in recent years, due to its potential applications in video surveillance, behavior analysis and virtual reality etc. [1]–[4]. Although astonishing improvement has been witnessed in image classification [5], an equivalent achievement has not been made in video action recognition. The reason is that action recognition requires not only spatial features (2D appearances), but also temporal features (3D motion). However, it is a challenging task in developing an efficient model to explore and represent these features.

Under such circumstances, several datasets [1], [6], [7] have been released to cater for the ever-growing need for action recognition in recent years. For instance, Fig. 1 shows an example from the Something-Something dataset [7], displaying the action of pushing something towards a certain direction. Apparently, it is insufficient to determine the direction of the pushing action merely based on a single image, it has to be inferred from a sequence. Considering the powerful feature extraction ability of CNNs, 3 types of spatial-temporal modeling methods become prevailing:

(1) Two-stream architecture [2], [8]–[11]: One stream processes original RGB input (spatial stream), and the other stream extracts temporal features from the optical flow. However, optical flow is quite computational expensive;

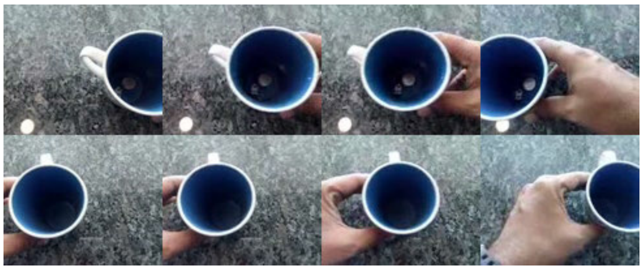
FIGURE 1. Example sequences from the Something-Something dataset [13]. First row: Pushing something from the right to the left; second row: Pushing something from the left to the right.

(2) 2D CNNS [12]–[17]: 2D CNNs extract spatial cues based on traditional 2D convolutional architectures, while temporal modules are also developed and attached to the 2D backbone to capture temporal information. Moreover, 2D CNNs require less computation than optical flow.

(3) 3D CNNs [18]–[25]: 3D CNNs expand the kernels of 2D CNNs along the temporal dimension, which achieve excellent performance by capturing the spatial and temporal features simultaneously simply based on RGB frames. As reported by [3], 3D CNNs exhibit strong temporal feature

The associate editor coordinating the review of this manuscript and approving it for publication was Yongjie Li.

modelling abilities. However, it inevitably requires huge training data to reach a good performance.

Partially inspired by (2) and (3), we develop an efficient spatial-temporal modeling network (dubbed TEFE) using ResNet-50 as the backbone. Our network consists of 2 core modules: The temporal extraction (TE) module and the feature enhancement (FE) module. Some existing methods [12], [16], [20], [21] acquired the long-term temporal information via stacking local temporal windows. Since the information needs to be propagated through long way through multiple layers, repeating the stacking operations would cause difficulty in optimization process [26]. To solve this problem, we present a TE module.It has 2 branches: a short-term branch and a long-term branch. The former one utilizes temporal convolutions to capture short-term motion sensitive information, while the latter one produces a kernel for temporal aggregation. On the other hand, FE module adopts an approximation of bilinear pooling operation to obtain more fine-grained spatial and temporal cues to enhance the features extracted by TE module. Although our FE module has 3D convolution kernels, it does not need to be pre-trained on the large scale action recognition datasets (e.g. Kinetics [1]). Instead, we only pre-train our network on ImageNet [27]. As a result, the integration of the 2 modules makes powerful representation ability in video action recognition tasks. The main contributions of our method are summarized as follows:

(1) Temporal extraction (TE) module is developed to capture both short-term and long-term temporal information together with two corresponding branches.

(2) Feature enhancement (FE) module is presented to obtain more fine-grained spatial and temporal cues through an approximation of bilinear pooling operation. Meanwhile, an initialization scheme is developed for FE module so as to avoid heavy pre-training on a very large video dataset.

(3) Extensive experiment have been conducted on mainstream datasets (Something - Something V1 & V2, Jester). The results prove that our model exceeds other existing models without pre-training on large video datasets.

The rest of the paper is organized as follows: related works are described in Section II. Details of algorithm implementation are depicted in Section III. The experimental results with ablation studies are shown in Section IV with thorough analysis. A final conclusion is drawn in Section V.

II. RELATED WORKS

Spatial-temporal modeling. Human actions include simple body movements,human-human interactions and human-object interactions.Action recognition has attracted lots of research attention in recent years,owing to its applications in many areas such as smart surveillance, video retrieval and audio-visual speech recognition system. For example, viseme classification is widely used in audio-visual speech recognition system [28].Using clustering method for viseme classification has achieved good results [29]. However,this method based on still images can not deal with the coarticulation effects well. The action recognition model can make use of temporal clues in the video to get the context information, which is helpful for viseme classification.

Motivated by the successful applications of deep convolutional architectures in image recognition, a variety of deep learning paradigms have pushed the limits of action recognition, leading to a surge of deep models. [22], [25], [30]–[32]. In early years, they were committed to 2D structures. Among them, Karpathy *et al.* [32] elaborated a deep 2D network, which was trained on a very large dataset (sports-1M) using individual RGB frames as its input. Although it integrated temporal information into its core modules, it failed to capture instantaneous motion information as short-term temporal features. In order to make up for this shortcoming, Simonyan *et al.* [33] advocated a two-stream ConvNet architecture, in which one stream captures the static spatial features from RGB frames and the other stream models the temporal information from optical flow separately. Then a weighted average result is generated from the 2 streams. Unfortunately, it did not gain a strong spatiotemporal representation ability in action classification. In this light, Feichtenhofer *et al.* [34], [35] placed a novel fusion module in the two-stream network so as to boost the spatiotemporal features. TSN [9] suggested a sparse sampling scheme for two-stream architecture to capture the different temporal scale features. However, it suffers two inherent deficiencies: Firstly, it relies heavily on pre-computed optical flow, which leads to additional computational cost and memory footprint; secondly, it may not work properly under complicated scenarios based on its simple fusion strategy.

In order to solve the aforementioned problems, powerful models are required to integrate the spatiotemporal features without the dependence on pre-computed optical flow. Du *et al.* [25] laid a solid foundation for 3D CNNs by outlining a 3D architecture for end-to-end action recognition, which inspired more efficient 3D models [3], [18], [19], [22] in the next few years. T3D [21] aimed to obtain different scale temporal information with 3D convolutions. In Slowfast [3], a slow path is created to capture spatial semantics while a fast path is used to distill motion information at a finer resolution. However, Slowfast has huge number of parameters, which requires extensive pre-training. Besides, it also has a slow rate of convergence in training. In recent years,some novel work has explored using skeleton data as input for action recognition [36], [37], [38]. These methods are robust to illumination and complex background.But due to the lack of appearance information, these methods can not effectively capture some dependencies that can be easily obtained from RGB images, such as some actions of human interaction with objects.

$(2+1)$ d method [19], [30] simplifies the 3D convolution structure by decoupling the 3D convolution kernel into 2D convolution kernel and 1D convolution kernel for extracting spatial and temporal features Respectively. Motion Feature Network [39] presented a motion block which using a shift operation for imitating optical flow to obtain the temporal features. TSM [15] built a temporal shift module, which achieved a satisfactory performance with relatively small network scales. STM [16] extracted spatiotemporal features by 1D convolution structure and 2D convolution structure, and used feature differencing operation to encode motion features. The extracted motion features can supplement the spatiotemporal features and improve the accuracy of recognition.TRN [17] attempted to combine multiple MLPs to reason temporal information.

Although various strategies have been presented to capture short-term temporal information in different ways, they lack aggregation abilities. To remedy this drawback, our TE module aims at instantaneous temporal extraction, which aggregates temporal features through long-term branch. Besides, the FE module is used to boost the fine-grained spatiotemporal information.

Bilinear models. Feature fusion includes max, sum and average pooling etc. Tsung *et al.* [40] introduced bilinear pooling to fuse fine-grained features in image classification task for the first time with satisfactory results. However, bilinear pooling is an operation of outer product, which involves high dimensional features and expensive computations. To address the problem, different solutions were proposed [41]–[43]. Among them, compact bilinear pooling [43] leveraged a tensor sketch algorithm to project the features from high-dimensional space to low dimensional space to reduce computations. Row rank bilinear pooling [42] utilized eigenvalue decomposition to avoid the original bilinear pooling. Multimodal compact linear pooling [41] used Hadamard product to approximate bilinear pooling. Diba *et al.* [14] developed a temporal linear encoder method (TLE) which applied compact bilinear pooling to action recognition tasks. Considering that TSN [9] lacks the aggregation of temporal information across frames, TLE [14] published a scheme to integrate temporal features from multiple dimensions, which performed feature fusion before the classification layers. In contrast, we implant FE module into multiple convolutional layers in the middle of the network to facilitate feature aggregation process. Moreover, Girdhar *et al.* [44] proposed an attention mechanism with bilinear pooling, forcing the network to focus on the fine-grained action interactions. Wang *et al. al.* [46] used compact bilinear pooling to fuse motion and spatial information from optical flow, which undoubtedly increased the complexity of the model. As can be seen, an efficient and powerful model is required for the fusion of spatial-temporal features to enhance feature representation. Through a thorough investigation of both bilinear models and spatial-temporal models, we find that they could be complementary to each other. Considering this, our TE module extracts both short-term and long-term temporal features, while the FE module reasons the fine-grained temporal cues (captured by TE) via an approximation of bilinear pooling. Finally, TEFE realizes motion modeling based on RGB frames as input (without calculation of optical flow), and obtains a powerful spatial-temporal feature representation through feature fusion.

We name our network TEFE by combination of TE and FE. The details of our algorithm are described in section 3 below.

III. ALGORITHM OVERVIEW

The overall architecture of TEFE is illustrated in Fig. 2 above. The input videos with different lengths are sampled using a sparse temporal sampling method [9]. First, the videos are divided into T segments with equal length. Then we randomly sample 1 frame from each segment to form the input sequence with T frames in total. Finally, we use TE module to explore motion cues and a FE module to reason the spatial-temporal features respectively.

A. TEMPORAL EXTRACTION MODULE (TE MODULE)

The TE module is designed for efficient temporal modeling, the architecture of TE module is illustrated in Fig.3.

Assume the shape of input feature X is characterized as: $[N, C, T, H, W]$, where N is the batch number, C is the feature channels and T is the temporal dimension, H and W are the spatial resolution. We employ a global spatial average pooling to highlight the motion-sensitive information, while ignoring some of the spatial layouts:

$$X^S = Pool(X), X^S \in R^{N\times C\times T\times 1\times 1} \tag{1}$$

Recall that TE embodies 2 branches. The feature X^S from (1) is then fed to both the short-term branch and the long-term branch individually.

1) SHORT-TERM BRANCH

The short-term branch aims to capture sensitive information of local motion. To restrain the computational load, we employ a 1D convolutional layer to reduce the channel dimensions by a factor of r ($r = 4$):

$$X^r = Conv * X^S, X^r \in R^{N\times \frac{C}{r}\times T\times 1\times 1} \tag{2}$$

The size of the 1D convolution kernel is set as 3, so as to learn the adjacent temporal information. Another 1D convolutional layer (called $\mathrm{Conv}_{\mathrm{exp}}$) is appended to expand the dimension of motion features to C, then motion-attentive weights M is defined in the form of a Sigmoid function:

$$M = Sigmoid(Conv_{\mathrm{exp}} * X^r), M \in R^{N\times C\times T\times 1\times 1} \tag{3}$$

To explore the motion-sensitive information, a channel-wise multiplication of the input feature and motion-attentive weights is implemented as follows:

$$X^0 = M \odot X \tag{4}$$

Where X^0 is the output of the short-term branch. Operator $\odot$ denotes a channel-wise multiplication. Through the above steps, the motion pattern has been fully explored and highlighted.

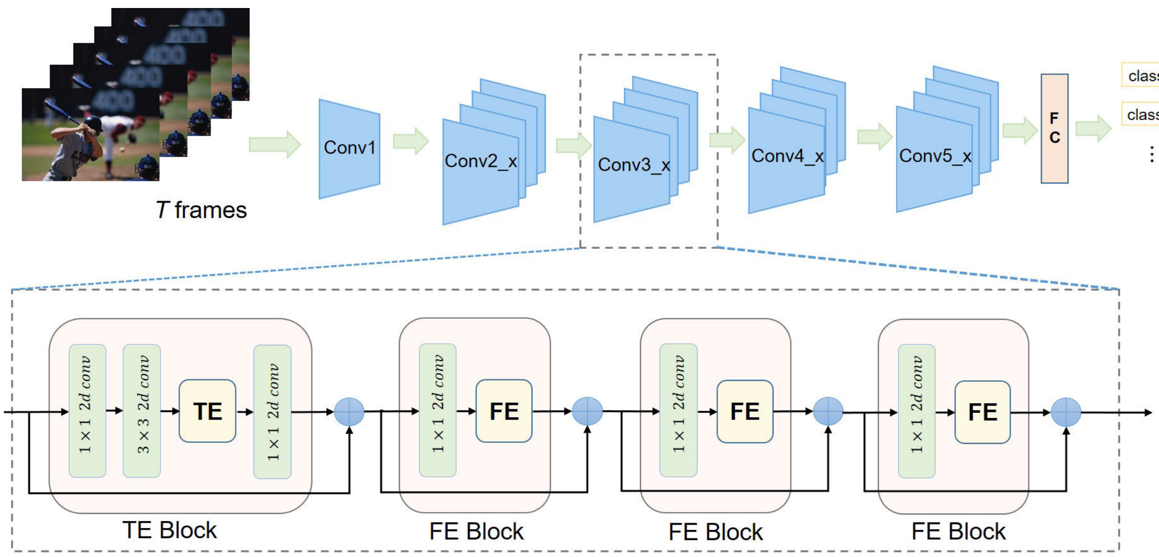

FIGURE 2. The overall architecture of TEFE network. We adopt Resnet-50 as the backbone, and implant TE and FE modules into residual blocks.

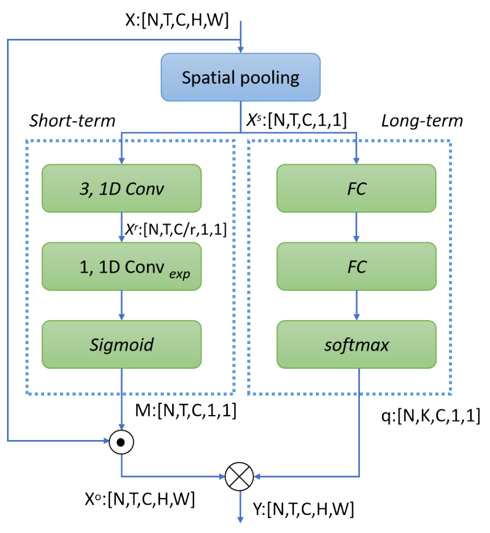

FIGURE 3. The architecture of TE module.

2) LONG-TERM BRANCH

Compared with the short-term branch, the long-term branch leverages a convolution kernel to integrate temporal information, without calculating the correlation among channels.

As shown in Fig. 3, we reap the benefits of stacked two fully connected (FC) layers to learn the temporal relations. After the 2nd FC layer, we add a softmax function to normalize aggregation weights, which is written as:

$$q_c = softmax(F(W_2, F(W_1, X_C^S)) \tag{5}$$

Where $q_c \in R^{N\times K\times 1\times 1\times 1}$ is a calculated weight for the c^{th} channel, and $K = 3$ is the kernel size.

The learned weights $q = \{q_1, q_2, \ldots, q_c\}$ is employed to aggregate the short-term temporal information. Formally,

$$Y = q \otimes X^0 \tag{6}$$

Where $\otimes$ represents a channel-wise convolutional operator, which ensures that the information among different channels would not be confused. $Y \in R^{N\times C\times T\times H\times W}$ is the output feature map.

Through the above steps, both local motion information and global temporal aggregation information are acquired by TE module with the aid of short-term and long-term branches.

B. FEATURE ENHANCEMENT MODULE (FE MODULE)

Some similar actions will increase the difficulty of recognition, which requires more powerful feature representation. Modhej et al. [47] studied how to separate similar features and achieved good results.Fine-grained recognition refers to the differentiation of inter-class features, which belong to the objects of the same category. As mentioned by [40], bilinear CNN models represent an image as a pooled outer product of CNN features in fine-grained recognition tasks. For temporal modeling in action recognition tasks, we find that those popular temporal feature extraction methods (e.g. optical flow or dense trajectory based methods) are easily affected by subtle local changes. In view of this, we believe that bilinear method could improve the action recognition results. Hence, our FE module adopts an approximation of bilinear operation to advance feature representation (extracted by TE). The bilinear models are discussed below, followed by the implementation details of FE module.

1) BILINEAR MODELS

Without loss of generality, consider two input features $x \in R^M$ and $x' \in R^N$, a bilinear pooling is performed as an outer product of x and x':

$$z = xx'^T, z \in R^{M\times N} \tag{7}$$

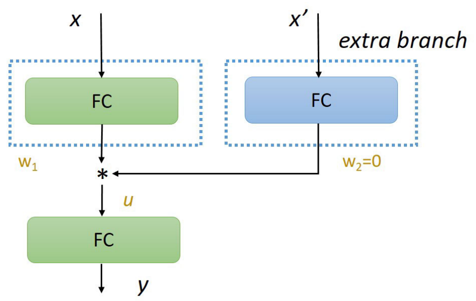

FIGURE 4. An approximation of bilinear pooling based on a 2-layer MLP with an extra branch. ∗ indicates element-wise multiplication.

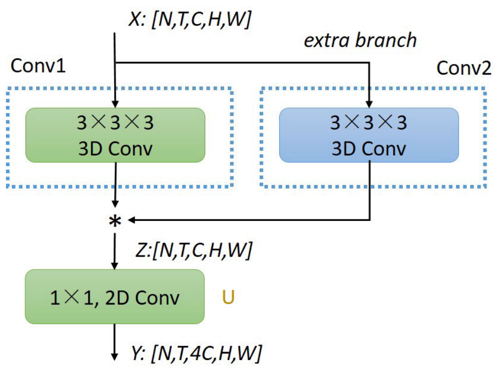

FIGURE 5. Architecture of FE module. ∗ indicates element-wise multiplication.

Where z is a feature fusion result. Unfortunately, as an input to the next layer, z would bring in a large number of parameters:

$$y = Wvec(z), \quad y \in R^D \tag{8}$$

Where $W \in R^{D\times M\times N}$, $vec()$ transforms a $M\times N$ matrix into an one dimensional vector with length MN. As discussed in [41], a hadamard low-rank bilinear pooling reduces the amount of parameters significantly. Therefore, we devise an approximation of bilinear pooling to replace the original operation. Firstly, we use 3 matrices $w_1 \in R^{R\times M}$, $w_2 \in R^{R\times N}$ and $u \in R^{D\times R}$ to approximate $W \in R^{D\times M\times N}$. The output y in (8) is computed as:

$$y = u(w_1 x * w_2 x') \tag{9}$$

Where $*$ denotes element-wise product. The matrices w_1, w_2 and u in (9) all have much smaller size than W, hence reduces the computational load.

Secondly, when we fix $w_2x' = 1$, the structure of the (9) can be regarded as a 2-layer MLP, which approximates a bilinear pooling. As shown in Fig. 4 below:

The structure can be built upon the basis of a 2-layer MLP by adding an extra branch, where the input is x' and the output is 1. This design endows a traditional 2-layer MLP more discrimination ability, which is also the basis for constructing FE module with pre-trained weights.

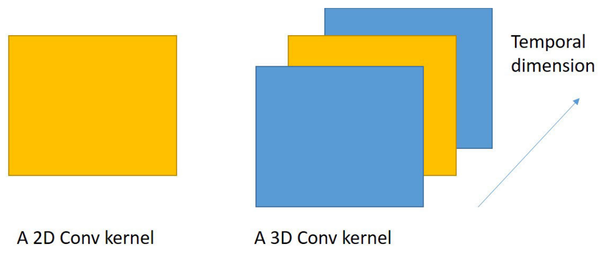

FIGURE 6. Illustration of pre-training process.

2) IMPLEMENTATION DETAILS OF FE MODULE

Suppose feature X is fed to the FE module, the corresponding output is calculated as:

$$Y = U(Conv1(X) * Conv2(X)) \tag{10}$$

We replace the original FC layers (shown in Fig. 4) with 3D convolution modules, considering their efficiencies in spatial-temporal feature extraction. Then a FE module is constructed in Fig. 5. As shown in Fig. 5 above, our FE module has two 3D convolution kernels. Conv1 is a 3D convolution kernel, which is stacked by three 2D convolution kernels along the temporal dimensions. To initialize Conv1, we load pre-training weights of the 2D convolution kernel stacked in the middle (yellow), and set the weights of the other 2D kernels (blue) as 0 (as shown in Fig. 6 below).

Conv2 is constructed in a similar way as Conv1. We initialize all the weights of Conv2 to 0, and bias to 1. In that case, the output of Conv2 would always be 1, regardless of the input. Then we have:

$$Conv1(X) * Conv2(X) = Conv1(X) \tag{11}$$

Through the above steps, we well preserve the pre-trained parameters on ImageNet with learned spatial features.

C. THE OVERALL STRUCTURE OF TEFE

Since TE and FE modules are compatible to ResNet architecture, we implant them into ResNet-50 in a flexible way and replace them with existing residual blocks to form the final structure. The overall architecture is illustrated in Fig. 2 in Section 3. In both TE and FE blocks, the first 1 × 1.2 D convolutional layer is used to reduce channel dimensions, and the second 3 × 3.2 D convolutional layer in TE block is used for extraction of spatial information. Then the feature maps are then propagated through TE and FE modules to gather temporal and fine-grained features, respectively. To ensure the effectiveness of the proposed structure, we have also tried different ways of placing the 2 blocks in different stages (from the 2nd to the 5th stage) of ResNet-50 to search for the optimal structure (as discussed in ablation study).

IV. EXPERIMENT

In this section, we firstly describe the open datasets (Something - Something V1 and V2, and Jester) we use

TABLE 1. Comparison with state-of-the-art methods on something-something V1. '-' indicates this parameter is not provided.

Method	Backbone	Frames	GFlops	pre-trained	top-1(%)	top-5(%)
TSN[9]	Resnet-50	8	33	Kinetics-400	19.7	46.6
TSN[9]	Resnet-50	16	66	Kinetics-400	19.9	47.3
I3D[18]	Resnet-50	32×6	918	ImageNet + Kinetics-400	41.6	72.2
NL I3D[18]	Resnet-50		1008		44.4	76.0
NL I3D+GCN[18]	Resnet-50+GCN		1818		46.1	76.8
ECO[22]	BNInception+Resnet-18	16	64	Kinetics-400	41.4	-
ECO_{En}[22]	BNInception+Resnet-18	92	267	Kinetics-400	46.4	-
SAST[48]	BNInception+Resnet-18	16	-	Kinetics-400	44.3	-
SAST[48]	BNInception+Resnet-18	32	-	Kinetics-400	45.6	-
TPN[2]	Resnet-50	8×10	330	ImageNet	40.6	-
TRN[17]	BNInception	8	16	ImageNet	34.4	-
TRN(RGB+Flow)[17]	BNInception	8+8	-	ImageNet	42.0	-
TSM[15]	Resnet-50	8	33	ImageNet	45.6	74.2
TSM[15]	Resnet-50	16	65	ImageNet	47.2	77.1
TSM_{En}[15]	Resnet-50	8+16	98	ImageNet	49.7	78.5
MTD^2P[49]	Inception-V1	64	-	ImageNet	48.2	78.4
MTD^2P[49]	Resnet-50	16	-	ImageNet	49.4	78.1
CorrNet[12]	Resnet-50	32×10	1150	ImageNet	49.3	-
CorrNet[12]	Resnet-101	32×10	1870	ImageNet	50.9	-
TEFE(Ours)	Resnet-50	8	90	ImageNet	46.7	75.3
TEFE(Ours)	Resnet-50	16	181	ImageNet	50.4	78.9
TEFE(Ours)	Resnet-50	8+16	271	ImageNet	**51.4**	**79.6**

to testify the general performance of our network followed by the implementation details of TEFE. Then we conducted experiments on them to compare the performances with other state-of-the-art methods. Meanwhile, ablation studies are carried out to verify the effectiveness of each module as well as different possible combinations in our architecture.

A. DATASETS

We evaluate our method on 3 action recognition datasets.

Something-Something v1& v2: Something V1 and V2 [6] are 2 large scale challenging video datasets with daily actions and interactions between people and common objects (e.g. bottles, fruits, spoons etc.). Totally, there are 110k videos in V1 and 220k videos in V2, which consists of 174 fine-grained categories, different backgrounds and viewpoints. We place special emphasis on these two datasets, since the fine-grained categories need to be distinguished via temporal reasoning.

Jester. Jester [5] is a third-person view gesture dataset for generic human hand gestures recognition, which has 27 different categories with 118k videos for training, 14k videos for validation and 14k videos for testing.

B. IMPLEMENTATION DETAILS

Training. Our method doesn't use all the frames of video. According to the sparse sampling strategy proposed by TSN [9],we evenly divide frames of video into T segments with the same length. Next, 1 frame is randomly selected from each of the segment to form a sequence of T frames in total as the input of the network, where the short side of the frame is set in [256, 320].

Besides, we randomly crop the frame into patches of $224*224$ for augmentation purpose, and use them for training. The final input size of our network is expressed as: $N\times T\times 3\times 224\times 224$, where N and T represent batch size and frame numbers, respectively. And we compare the performances by setting T as 8 or 16. Our hardware platform is NVIDIA 2080Ti GPU with a mini-batch of 20 (when $T = 8$) or 10 (when $T = 16$). For Something - Something V1 & V2, we set an initial learning rate as 0.001, decays by 10 at 30/40/45 epochs, and stops at 50 epochs.mini-batch SGD is used as an optimizer, and the momentum and weight decay are set as 0.9 and $5e^{-4}$. And we only use the ImageNet to pre-trained model.

Inference. To obtain a comprehensive assessment of our model, we report Top1 & Top5 accuracy (%) and GFlops on validation sets. We resize video frames to 224×256 to cover the spatial dimensions and then randomly sample T frames for temporal dimension from a full-length video. And we only infer with single clip without other cropping strategies.

C. IMPLEMENTATION DETAILS RESULTS ON SOMETHING-SOMETHING DATASETS

In this section,we compare our network with other popular methods on Something-Something and Jester datasets.

TABLE 2. **Comparison with the state-of-the-arts methods on something-something v2. '-' indicates this parameter is not provided.**

Method	Backbone	Frames	GFlops	top-1(%)	top-5(%)
TSN[9]	Resnet-50	8	33	27.8	57.6
TSN[9]	Resnet-50	16	66	30.0	60.5
TRN[17]	BNInception	8	16	48.8	77.6
TRN(RGB+Flow)[17]	BNInception	8+8	-	55.5	-
TSN+TPN[2]	Resnet-50	8×10	330	59.1	-
CPNet[50]	Resnet-18	16×6	-	54.1	82.1
CPNet[50]	Resnet-34	16×6	-	57.7	84.0
STIN[51]	-	-	-	60.2	84.4
TSM[15]	Resnet-50	8×6	198	59.1	85.6
TSM[15]	Resnet-50	16×6	390	63.4	88.5
SlowFast-R50[4]	Resnet-50	64	132	61.7	-
TEFE(Ours)	Resnet-50	8	90	59.6	85.7
TEFE(Ours)	Resnet-50	16	181	61.8	87.4
TEFE(Ours)	Resnet-50	8+16	271	**63.5**	**88.7**

Something-Something V1. The results are shown in Table 1 above, including names, accuracies, frames sampled, backbones and GFlops for a fair comparison.

The listed methods can roughly be classified into 2 types. TSN [9] is the baseline, followed by some 3D CNN based methods, including I3D [18], ECO [22] and SAST [48] in the middle, and some 2D CNN based methods, including TRN [17], TPN [2], TSM [15] MTD^2 P [49] and CorrNet [12] and the lower part.Compare the results of 8 frames,our network exceeds other 3D based methods on V1. As for the results of 24 frames, we surpass ECOEn [22] and SAST [48] by 5% and 5.8%, respectively. In the meantime, those 3D based methods are pre-trained on the very large dataset (e.g. Kinetics [1]) with much higher computational cost. Although our network also has some 3D convolution modules, it is only pre-trained on the ImageNet dataset (like other 2D based methods).

Obviously, the baseline method TSN [9] gets rather poor results due to the lack of temporal modeling ability. In comparison, our accuracy is 26.8% higher than TSN with only 8 frames sampled. With the same backbone (ResNet-50), we outperform CorrNet [12] 2.1% in accuracy. When using ResNet-101 as the backbone, CorrNet [12] obtains a decent accuracy, but its GFlops is too high (1180G). We achieve the highest accuracy (51.4% on Top 1%, and 79.6% on Top 5% respectively) among all the methods, while maintaining a relatively low GFlops.

Something-Something V2. The results on Something-Something V2 are shown in Table 2 below. Generally, our network gains a huge improvement over TSN [9] baseline again, which also surpasses other popular methods.

D. RESULTS ON JESTER DATASET

Jester [6] is a dataset created for subtle gesture recognition.Comparative results are shown in Table 3.We achieve the highest Top-1 accuracy over other methods.

TABLE 3. **Comparison with the state-of-the-art methods on jester.**

Method	Backbone	Frame s	Top-1 (%)	Top-5(%)
TSN[9]	Resnet-50	8	81.0	99.0
TRN[16]	BNInception	8	95.3	-
MFNet[39]	Resnet-50	7	96.1	99.7
MFF[31]	BNInception	8	96.3	99.9
STM[16]	Resnet-50	8×30	96.6	99.9
TEFE(Ours)	Resnet-50	8	96.7	99.9

TABLE 4. **Impact of te and fe module: the table shows the contributions of each module to tefe:.**

Model	top-1	Δtop-1
TSN	19.9	
only TE	43.8	+23.9
only FE	46.1	+26.2
TEFE	**50.4**	**+30.5**

Compared with the baseline TSN [9], we improve the Top-1 accuracy by 15.7%.

E. ABLATION STUDIES

Ablation studies have been conducted on Something - Something V1 to testify the efficiency of the network with different combinations of TE and FE modules. We sample 16 frames as input and use TSN [8] as the baseline. The results are shown in Table 4 and Table 5.

1) INVESTIGATE THE IMPACT OF TWO MODULES

The two modules can be attached to a standard ResNet architecture independently. To validate the contributions of

TABLE 5. we place different number of te and fe modules in different stages of resnet-50.

stage2	stage3	stage4	stage5	Top-1 (%)
TE=3, FE=0	TE=4, FE=0,	TE=6, FE=0	TE=3, FE=0	46.7
TE=0, FE=0,	TE=0, FE=4,	TE=0, FE=6	TE=0, FE=3	46.3
TE=3, FE=0	TE=3, FE=1,	TE=3, FE=3	TE=3, FE=0	47.8
TE=2, FE=0	TE=2, FE=2	TE=2, FE=4	TE=2 FE=1	48.7
TE=1, FE=0	TE=1, FE=3	TE=1, FE=5	TE=1, FE=2	**50.4**

each module in TEFE, we investigate the results of placing vs not placing the two modules (listed in Table 4). Recall that the TE module learns motion features efficiently, which improves Top-1 accuracy by 23.9% on the basis of TSN, while FE module learns fine-grained spatial-temporal information and further improves Top-1 accuracy by 26.2% accordingly. The combination of TE and FE allows us to learn richer motion and spatial-temporal features, and achieve higher accuracy.

2) INVESTIGATION OF DIFFERENT LOCATIONS AND NUMBER OF TE AND FE MODULES

An original ResNet-50 architecture has 5 stages, which consists of 3, 4, 6, 3 blocks in stage 2∼5, respectively. We try to place different number of TE and FE modules in different stages, and compare their Top-1 accuracies (as shown in Table 5). First of all, we do not place any FE modules in stage 2 in order to reduce computation. We find that either TE or FE modules alone can improve the performance to some extent compared with baseline. We also confirm our initial hypothesis that TE and FE are indeed complementary to each other. When we insert both of them, we gain an even higher accuracy. Table 5 shows the accuracy by placing different number of TE and FE modules in different stages. Especially, the last combination achieves the best results.

F. COMPARATIVE ANALYSIS

In this section, we compare the effectiveness of TEFE in classifying different actions (from Something-Something V1) with our baseline TSN [9]. Fig. 7 and Fig. 8 show 2 sequences of similar actions. Fig. 9 compared the accuracies of similar action recognition achieved by our method and TSN, and Fig. 10 compared the accuracies for some strong temporal related actions.As can be seen in Fig.9, due to our fine-grained classification ability, we can improve the accuracy of similar actions.As shown in Fig. 10, TSN does not perform well for actions with strong temporal dependences, while we significantly improve the performances over TSN. The reason behind this phenomenon is that our method explores temporal relations in video sequences.

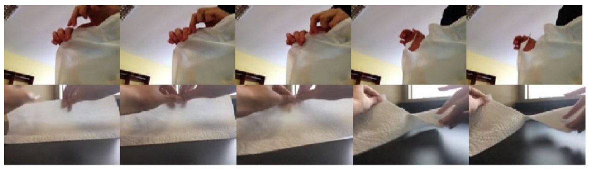

FIGURE 7. Two similar actions in Something-Something V1 dataset. Top: Tearing something just a little bit;Bottom:Tearing something into two pieces.

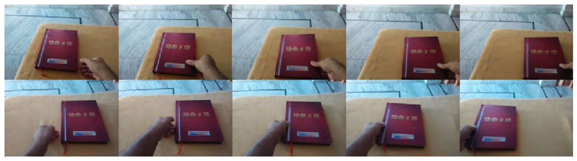

FIGURE 8. Two strong temporal-related actions in Something-Something V1 dataset. Top: Pulling something from left to right;;Bottom:Pulling something from right to left.

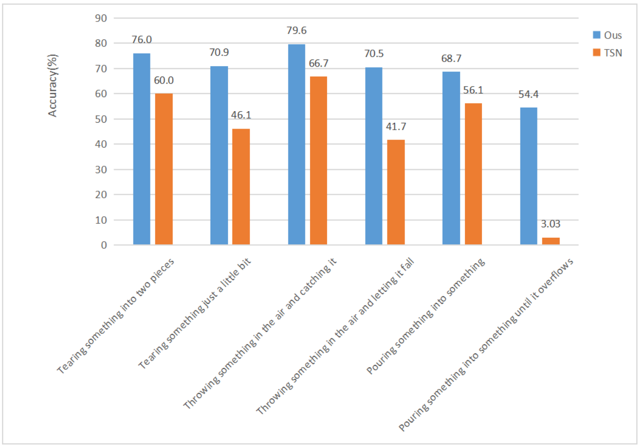

FIGURE 9. Comparison of recognition accuracy between our method and baseline method TSN on some similar actions.

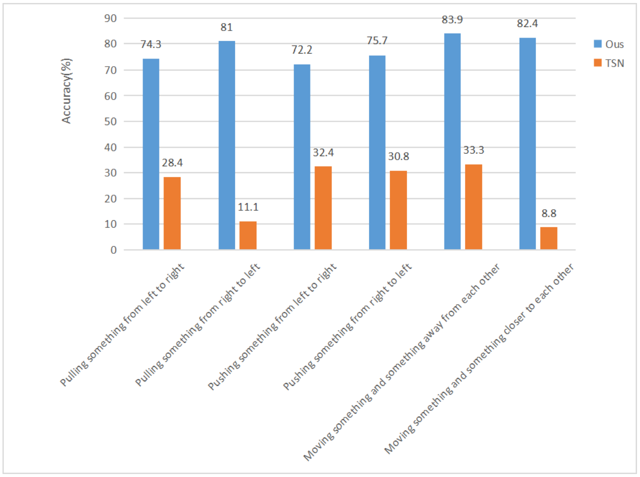

FIGURE 10. Comparison of recognition accuracy between our method and baseline method TSN on some strong temporal-related actions.

G. VISUALIZATION OF ACTIMATION MAPS

We use Grad-Cam [52] to visualize activation maps. We sample 16 frames as the input and only visualize the activation

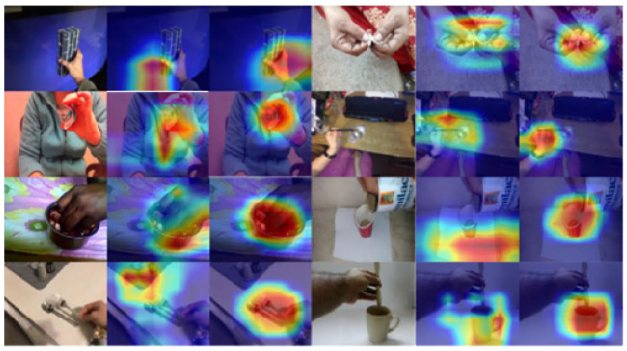

FIGURE 11. Visualization results on Something-Something V1. Left: Input video frames; Middle: baseline; Right: TEFE.

map of the middle frame (Fig. 11). As shown in Fig. 7.The left column display the original video frames, the middle column show the results of the baseline, and the right column exhibit our results. Apparently, TEFE is more concentrated on the hand motion and interaction with objects, while the baseline obviously lose focuses for most of the cases. The activation maps reflect the fact that TEFE is able to capture the motion regions due to its strong temporal modeling ability.

V. CONCLUSION

We present TEFE, an action recognition network with temporal extraction and feature enhancement modules to reach a balance between recognition accuracy and computation load. In particular, our TE module captures both short-term and long-term temporal features, while FE module obtains rich fine-grained spatial and temporal cues which are extracted by TE. Furthermore, TEFE is also appealing in the sense that it is pluggable to any ResNet architecture.

Extensive experiments have been conducted on 3 mainstream datasets to verify the efficacy of our network. The experimental results prove that TEFE achieves superior performances than other popular methods. Finally, ablation studies are carried out to demonstrate the efficiency of each module in the entire architecture.

REFERENCES

[1] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6299–6308.

[2] C. Yang, Y. Xu, J. Shi, B. Dai, and B. Zhou, "Temporal pyramid network for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 591–600.

[3] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "SlowFast networks for video recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6202–6211.

[4] C.-Y. Wu, R. Girshick, K. He, C. Feichtenhofer, and P. Krahenbuhl, "A multigrid method for efficiently training video models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 153–162.

[5] N. Esfandiari and A. Bastanfard, "Improving accuracy of pedestrian detection using convolutional neural networks," in *Proc. 6th Iranian Conf. Signal Process. Intell. Syst. (ICSPIS)*, Dec. 2020, pp. 1–6.

[6] (2019). *The 20bn-Jester Dataset V1*. [Online]. Available: https://20bn.com/datasets/jester

[7] R. Goyal, S. E. Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, and M. Mueller-Freitag, "The 'something–something' video database for learning and evaluating visual common sense," in *Proc. ICCV*, vol. 2, 2017, p. 8.

[8] C. Feichtenhofer, A. Pinz, and A. Zisserman, "Convolutional two-stream network fusion for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1933–1941.

[9] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. V. Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. ECCV*, Oct. 2016, pp. 20–36.

[10] A. Abdelbaky and S. Aly, "Two-stream spatiotemporal feature fusion for human action recognition,' *Vis. Comput.*, vol. 37, pp. 1821–1835, Aug. 2020.

[11] C. Liu, J. Ying, and H. Yang, "Two-stream spatiotemporal action recognition approach based on two-stream convolutional neural network model," *Vis. Comput.*, vol. 37, no. 6, pp. 1327–1341, Jun. 2021.

[12] H. Wang, D. Tran, L. Torresani, and M. Feiszli, "Video modeling with correlation networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 352–361.

[13] H. Shao, S. Qian, and Y. Liu, "Temporal interlacing network," in *Proc. AAAI*, vol. 34, Apr. 2020, pp. 11966–11973.

[14] A. Diba, V. Sharma, and L. Van Gool, "Deep temporal linear encoding networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 1541–1550.

[15] J. Lin, C. Gan, and S. Han, "TSM: Temporal shift module for efficient video understanding," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 7083–7093.

[16] B. Jiang, M. Wang, W. Gan, W. Wu, and J. Yan, "STM: SpatioTemporal and motion encoding for action recognition," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 2000–2009.

[17] B. Zhou, A. Andonian, A. Oliva, and A. Torralba, "Temporal relational reasoning in videos," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 803–818.

[18] X. Wang and A. Gupta, "Videos as space-time region graphs," in *Proc. ECCV*, Sep. 2018, pp. 399–417.

[19] S. Xie, C. Sun, J. Huang, Z. Tu, and K. Murphy, "Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification," in *Proc. ECCV*, Sep. 2018, pp. 305–321.

[20] C. Feichtenhofer, "X3D: Expanding architectures for efficient video recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 203–213.

[21] A. Diba, M. Fayyaz, V. Sharma, A. H. Karami, M. M. Arzani, R. Yousefzadeh, and L. Van Gool, "Temporal 3D ConvNets: New architecture and transfer learning for video classification," 2017, *arXiv:1711.08200*.

[22] M. Zolfaghari, K. Singh, and T. Brox, "ECO: Efficient convolutional network for online video understanding," in *Proc. ECCV*, Sep. 2018, pp. 695–712.

[23] J. Cai and J. Hu, "3D RANs: 3D residual attention networks for action recognition," *Vis. Comput.*, vol. 36, no. 6, pp. 1261–1270, Jun. 2020.

[24] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi, "Human action recognition using factorized spatio-temporal convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4597–4605.

[25] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[27] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. CVPR*, Jun. 2009, pp. 248–255.

[28] A. Bastanfard, M. Aghaahmadi, M. Fazel, and M. Moghadam, "Persian viseme classification for developing visual speech training application," in *Proc. Pacific–Rim Conf. Multimedia*, Dec. 2009, pp. 1080–1085.

[29] M. Aghaahmadi, M. M. Dehshibi, A. Bastanfard, and M. Fazlali, "Clustering Persian viseme using phoneme subspace for developing visual speech application," *Multimedia Tools Appl.*, vol. 65, no. 3, pp. 521–541, Aug. 2013.

[30] D. Tran, H. Wang, L. Torresani, J. Ray, Y. LeCun, and M. Paluri, "A closer look at spatiotemporal convolutions for action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6450–6459.

[31] O. Kopuklu, N. Kose, and G. Rigoll, "Motion fused frames: Data level fusion strategy for hand gesture recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2103–2111.

[32] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and A. L. Fei-Fei, "Large-scale video classfication with convolutional neural networks," in *Proc. CVPR*, Jun. 2014, pp. 1725–1732.

[33] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. NIPS*, 2014, pp. 568–576.

[34] C. Feichtenhofer, A. Pinz, and R. Wildes, "Spatiotemporal residual networks for video action recognition," in *Proc. Adv. neural Inf. Process. Syst.*, 2016, pp. 3468–3476.

[35] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4768–4777.

[36] J. Weng, C. Weng, and J. Yuan, "Spatio-temporal naive-bayes nearest-neighbor (ST-NBNN) for skeleton-based action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4171–4180.

[37] J. Weng, M. Liu, X. Jiang, and J. Yuan, "Deformable pose traversal convolution for 3D action and gesture recognition," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 136–152.

[38] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 1–10.

[39] M. Lee, S. Lee, S. Son, G. Park, and N. Kwak, "Motion feature network: Fixed motion filter for action recognition," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 387–403.

[40] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear CNN models for fine-grained visual recognition," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1449–1457.

[41] J.-H. Kim, K. W. On, W. Lim, J. Kim, J.-W. Ha, and B.-T. Zhang, "Hadamard product for low-rank bilinear pooling," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[42] S. Kong and C. Fowlkes, "Low-rank bilinear pooling for fine-grained classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7025–7034.

[43] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, "Compact bilinear pooling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 317–326.

[44] R. Girdhar and D. Ramanan, "Attentional pooling for action recognition," in *Proc. NIPS*, 2017, pp. 34–45.

[45] Y. Wang, M. Long, J. Wang, and P. S. Yu, "Spatiotemporal pyramid network for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2097–2106.

[46] Z. Qiu, T. Yao, C.-W. Ngo, X. Tian, and T. Mei, "Learning spatio-temporal representation with local and global diffusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12056–12065.

[47] N. Modhej, A. Bastanfard, M. Teshnehlab, and S. Raiesdana, "Pattern separation network based on the hippocampus activity for handwritten recognition," *IEEE Access*, vol. 8, pp. 212803–212817, 2020.

[48] F. Wang, G. Wang, Y. Huang, and H. Chu, "SAST: Learning semantic action-aware spatial-temporal features for efficient action recognition," *IEEE Access*, vol. 7, pp. 164876–164886, 2019.

[49] J. Wang, Y. Lin, M. Zhang, Y. Gao, and A. J. Ma, "Multi-level temporal dilated dense prediction for action recognition," *IEEE Trans. Multimedia*, early access, Jun. 7, 2021, doi: 10.1109/TMM.2021.3087023.

[50] X. Liu, J.-Y. Lee, and H. Jin, "Learning video representations from correspondence proposals," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4273–4281.

[51] J. Materzynska, T. Xiao, R. Herzig, H. Xu, X. Wang, and T. Darrell, "Something-else: Compositional action recognition with spatial-temporal interaction networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1049–1059.

[52] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 336–359, Oct. 2019.

JIE JIANG received the B.S. degree from Sichuan University, Chengdu, China, in 2019, where she is currently pursuing the M.S. degree, under the supervision of Associate Professor Y. Zhang. Her current research interests include action recognition and computer vision.

YI ZHANG received the B.S. degree from the University of Electronic Science and Technology of China, in 2004, and the Ph.D. degree from the National University of Singapore, in 2010. Since 2011, he has been an Associate Professor with the Faculty of Computer Science, Sichuan University, China. He has held a number of patents of computer vision. His current research interests include object tracking, action recognition, and semantic segmentation.