Dual-Branch Network With a Subtle Motion Detector for Microaction Recognition in Videos

Yang Mi, Xingyuan Zhang, Zhongguo Li, and Song Wang, *Senior Member, IEEE*

***Abstract*— By involving only subtle motions of body parts, video-based microaction recognition is a very important but challenging problem. Most existing action recognition methods are developed for general actions, and the current state-of-the-art methods usually largely rely on high-layer features learned from convolutional neural networks (CNNs). High-layer CNN features usually contain more semantic information but less detailed information. However, detailed information can be important for microactions due to the motion subtleness of such actions. In this paper, we propose to more effectively learn midlayer CNN features for enhancing microaction recognition. More specifically, we develop a new dual-branch network for microaction recognition: one branch uses the high-layer CNN features for classification, and the second branch further explores the midlayer CNN features for classification. In the second branch, we introduce a novel subtle motion detector consisting of three modules: 1) a discriminative spatial-temporal feature learning module, which further learns the subtle motion features corresponding to the discriminative spatial-temporal regions, 2) a parallel multiplier attention module, which further refines the features learned in channels and spatial-temporal domains, and 3) an activation fusion module, which fuses the max and average activations from midlayer CNN features for classification. In the experiments, we build a new microaction video dataset, where the micromotions of interest are mixed with other larger general motions such as walking. Comprehensive experimental results verify that the proposed method yields new state-of-the-art performance in two microaction video datasets, while its performance on two general-action video datasets is also very promising.**

***Index Terms*— Action recognition, microactions, dual-branch network, subtle motion detector, attention.**

Manuscript received October 2, 2019; revised March 1, 2020 and April 7, 2020; accepted April 14, 2020. Date of publication April 29, 2020; date of current version May 7, 2020. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61672376 and Grant U1803264. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Xiaolin Hu. *(Corresponding author: Song Wang.)*

Yang Mi is with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201 USA (e-mail: miy@email.sc.edu).

Xingyuan Zhang is with the Beijing Key Lab of Traffic Data Analysis and Mining, Beijing Jiaotong University, Beijing 100044, China (e-mail: xingyuanzhang@bjtu.edu.cn).

Zhongguo Li is with the College of Intelligence and Computing, Tianjin University, Tianjin 300072, China (e-mail: lizhongguo@tju.edu.cn).

Song Wang is with the Department of Computer Science and Engineering, University of South Carolina, Columbia, SC 29201 USA, and also with the College of Intelligence and Computing, Tianjin University, Tianjin 300072, China (e-mail: songwang@cec.sc.edu).

Digital Object Identifier 10.1109/TIP.2020.2989864

I. Introduction

In this paper, we study the problem of recognizing the micro human actions present in a video. Microactions are human actions, where only subtle body motion is apparent [1]. As we can see from Fig. 1, microactions such as slight head-turning or small hand-pointing only involve small movements of one or more local body parts (e.g., head or hand & arm) and are usually highly localized in the spatial-temporal domain compared with general actions such as golf and pullup, which contain relatively larger movements of human bodies and usually occur within much wider regions. This research can benefit many important video-based applications, such as video behavior analysis, video surveillance and social interaction understanding. For example, a subtle head-nodding usually indicates an agreement, and small-scale hand-pointing movement could mean sharing attention with another person. In law enforcement, from the videos taken by closed-circuit televisions, we can identify crime-related microactions, such as concealingly passing and receiving drugs between suspects, for enhancing security in public areas.

Video-based action recognition [2]–[8] has drawn significant attention in the computer vision and image processing communities. The most widely adopted approaches train classifiers on spatial-temporal features extracted from videos. Traditional methods extract handcrafted features, such as dense trajectories [2], followed by a classifier, e.g., support vector machine (SVM) [9], for recognizing different actions. Recently, more effective deep-learning-based methods have been developed to learn deep features via convolutional neural networks (CNNs) and obtain state-of-the-art performance on many general-action video datasets, such as HMDB51 [10] and UCF101 [11]. The performance of these CNN-based methods [12], [13] largely depends on the discriminativeness of high-layer deep features, since the final classification is performed on the high-level features pooled from the last convolutional layers of CNN. High-layer CNN features usually focus more on semantic information but less on detail information [14]. However, for microactions where the involved motions are subtle and highly localized, the high-layer CNN features, which are usually of small size and highly squeezed, may lose detailed information. For example, in state-of-the-art CNN architectures such as ResNet [15] and Inception [16], high-layer CNN features (e.g., the output feature maps of

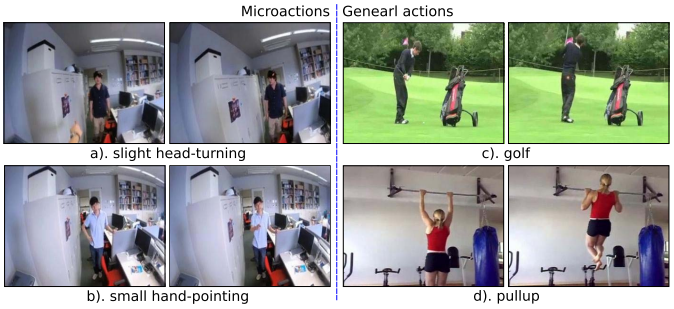


Fig. 1. Left: sample video frames of two microactions: a) slight head-turning; b) small hand-pointing. Right: sample video frames of two general actions: c) golf; d) pullup.

inception (5a-5b) or ResNet conv5_x), which are used for final classification, have a small size of 7×7 and may not preserve sufficient local details of microactions.

To address the above issue, instead of only using high-layer CNN features for final classification as in many existing deep-learning-based action recognition methods [12], [13], [17], we also thoughtfully consider the features from the middle layers of CNN, which we refer to midlayer CNN features for simplicity in this paper. The motivation behind this is inspired by the successful use of midlayer CNN features in fine-grained image classification, where the discriminative regions for fine-grained categories are usually highly localized [18]. Compared with high-layer CNN features, midlayer CNN features usually present more detailed information [14], which can be an important cue for detecting subtle motion for microactions. Note that midlayer CNN features have been successfully used for image-level tasks, including fine-grained image classification [18] and image object detection [19], but they have not been well explored in video classification tasks. In this paper, we propose a dual-branch network to use both the high-layer and midlayer CNN features for microaction recognition. We specifically propose a novel subtle motion detector, which is built on top of midlayer CNN features, to effectively capture subtle motion and learn local discriminative motion patterns for better recognition of microactions.

More specifically, we first sample several short video segments from the input video and then employ a state-of-the-art CNN architecture (e.g., BN-Inception [20]) as a backbone to extract both high-layer and midlayer CNN features from the video segments. We then propose a network with two branches, named the G-branch and L-branch, to explore high-layer and midlayer CNN features for classifications, respectively. For the G-branch, we simply use the high-layer CNN features to compute classification scores [12]. For the L-branch, we develop a novel subtle motion detector, which consists of three modules: 1) a discriminative spatial-temporal feature learning module, where we adopt the smallest reception field – one pixel and apply 3D convolutional layers with $1 \times 1 \times 1$ filters on midlayer CNN features to learn the highly localized motion patterns; 2) an activation fusion module, which uses max-pooling and average pooling to obtain max and average activations from the learned features, followed by fusion for classification; and 3) a parallel multiplier attention module, which is inserted before module 2) such that the learned features can focus on the discriminative areas. Finally, the classification scores of both branches are fused via weighted sum to generate the overall classification score for each video segment, and the classification scores from all the sampled segments are then combined for the final video-level classification. The whole network can be trained in an end-to-end fashion. In the experiments, we evaluate the proposed method not only on an existing microaction video dataset [1] but also on a new microaction video dataset that we collected. In this new dataset, the microactions of interest are usually mixed with other relatively larger general motions, such as walking, to better reflect the scenarios in real applications. We also evaluate the proposed method on a general-action dataset [10].

The main contributions of this paper are as follows: 1) we propose a novel dual-branch network with a subtle motion detector for better recognizing microactions; 2) we show that effectively learning the midlayer CNN features is essential for microaction recognition; 3) we introduce a new microaction video dataset, where the micromotions of interest are mixed with general motions; and 4) the proposed method achieves state-of-the-art performance on the microaction datasets and good results on a general-action dataset.

The remainder of this paper is organized as follows. Section II briefly overviews the related work. Section III introduces the proposed method in detail. Sections IV and V report the experimental results, followed by a brief conclusion in section VI.

II. Related Work

Handcrafted features for action recognition. Many traditional methods have been developed to extract handcrafted spatial-temporal features from video, such as space-time interest points (STIP) [21], 3-dimensional sift descriptors (SIFT-3D) [22], and 3D histograms of gradient (HOG3D) [23]. Improved dense trajectories (iDT) [24] are widely considered to be one of the most effective handcrafted features for video action recognition. They compute HOG (histograms of oriented gradients) [25], HOF (histograms of opticalflow) [26], and MBH (motion boundary histogram) [27] along trajectories within local 3D regions, and the computed features are further encoded by Fisher Vector (FV) [28], followed by SVM classification for action recognition.

Deeply learned features for action recognition Inspired by the success of CNNs in image classification, many methods were developed recently to learn deep features via CNNs for video action recognition. The performance of these deep-learning methods has gradually outperformed traditional methods. Typically, deep-learning methods first temporally sample a video into several short segments, and then apply either 2D or 3D CNNs on each segment for feature extraction. Karpathy *et al.* [29] extended the connectivity of the CNN architecture used in image classification to the spatial-temporal domain by using different temporal fusion strategies. Simonyan and Zisserman [30] proposed two-stream CNNs to learn spatial and temporal deep features on RGB frames and optical flows. Tran *et al.* [31] designed 3D CNNs to simultaneously learn both spatial and temporal deep features on RGB frames by

using 3D convolutions and poolings. Wang *et al.* [4], [12] proposed a temporal segment network based on two-stream CNNs to fuse video-clip representation into video representation. Feichtenhofer *et al.* [17] extended two-stream CNNs with multiplicative interactions between spatial and temporal streams. In [32], [33], recurrent neural networks (RNNs) with long short-term memory (LSTM) cells were developed to combine frame-level features into video-level representations. Later, the combination of 3D CNN with LSTM was explored in [34], in which 3D CNN was used to extract spatial-temporal features from saliency-aware videos followed by LSTM to further model temporal dynamics. Tran *et al.* [13] proposed a two-stream network with spatial-temporal blocks, which decompose 3D convolution into separate spatial and temporal components to learn spatial-temporal deep features. Most of these deep-learning methods are designed for general-action recognition, such as walking and jumping, which involve relatively large motions, by classifying high-layer CNN features. As mentioned before, they may not handle microaction recognition well without considering the problem of motion subtleness. In this paper, we follow the general framework of the recent state-of-the-art deep-learning methods and further incorporate a subtle motion detector to better utilize the midlayer CNN features for recognizing microactions.

Microaction recognition. There are few prior works specializing in microaction recognition. Yonetani *et al.* [1] proposed manually synchronizing first-person (actor's view) and second-person (observer's view) videos and extracted features from both videos for microaction recognition. Specifically, they used cumulative displacement [35] encoded by pooled time series [36] from first-person videos and iDT from second-person videos, followed by a standard linear decision function to describe the relative importance of the two point-of-view features. They also built the first microaction video dataset for performance evaluation. However, first-person videos are not always available in practice, and the manual synchronization of two videos can be time consuming. Furthermore, they directly used the available methods that were developed for general actions to extract the second-person videos, without specifically considering the motion subtleness. In this paper, we perform microaction recognition by only using second-person videos and propose a subtle motion detector to address motion subtleness. Also related is our earlier work [37], which built a temporal pyramid on video segments to more finely represent microactions in the temporal domain. However, this earlier method still directly used high-layer CNN features to construct the pyramid, without considering the midlayer CNN features that can present more detailed information for microactions. In the experiments, we show that the proposed method significantly outperforms these two previous microaction recognition methods.

Midlevel video features. Wang *et al.* [18] constructed a discriminative local patch detector by directly applying a 2D convolutional layer with 1×1 filters on the **midlayer CNN features**, followed by a max-pooling layer to obtain the most discriminative local patch, which was then applied to improve the fine-grained image classification. In this paper, we extend this idea to video action classification by adjusting middle CNN layers and using 3D convolutional layers with $1 \times 1 \times 1$ filters as a local cubic detector in the spatial-temporal domain. Also different from their work, which only uses the max-pooling layer for extracting the max activation, we extract the global activation via average pooling in addition to the max activation to fully explore midlayer CNN features. In the experiments, we show that simply embedding their idea into 2D-CNN-based action recognition approaches does not lead to convincing performance improvement on microaction recognition. Liu *et al.* [19] proposed the single shot multibox detector (SSD) for image object detection by using multiple levels of CNN features to allow predictions for multiscale detections. In this paper, midlayer CNN features are used as a complementary feature resource for microaction recognition. Note that the proposed method is quite different from the methods [38]–[41] that build the **midlevel video representations** on low-level handcrafted features for action recognition. The proposed method is also different from the deep-learning-based image classification methods [42], [43] with **midlevel visual elements**, where midlevel image patches are first extracted and then fed into CNNs for feature learning.

Attention. The attention mechanism has also been studied in several action recognition methods by enabling the models to concentrate on key information. These can be categorized into hard attention and soft attention. Hard attention usually makes hard binary choices to select salient regions and may face training difficulties. For example, Mallya and Lazebnik [44] referred to the detected human bounding box as the salient region. Recently, soft attention has been developed by using weighted averages instead of a hard selection to avoid such difficulties. Sharma *et al.* [45] built a soft-attention long short-term memory (LSTM) on top of multilayered recurrent neural networks (RNNs) to focus selectively on parts of video frames. Li *et al.* [46] proposed an end-to-end sequence learning model of VideoLSTM by hardwiring convolutions in the soft-attention LSTM. In these methods, additional information is usually required to predict the attention at the next time by LSTM, which may cause substantial computational cost and yield limited improvement. To reduce the computational cost, a more advanced self-attention mechanism [47] was developed, without the need for extra information. Li *et al.* [48] proposed using 2D convolutional layers and LSTM cells to obtain spatial and temporal self-attention, respectively. Du *et al.* [49] used the entire spatial-temporal pyramid that consisted of feature maps with different sizes from different convolutional layers to learn spatial-temporal attention based on principal component analysis (PCA). Wang *et al.* [50] viewed self-attention as a nonlocal operation in the space-time domain and developed nonlocal blocks for video classification. Inspired by the self-attention mechanism, the proposed parallel multiplier attention extends the nonlocal operation to the channel domain and further combines both channel and spatial-temporal attention via a multiplier gate to improve action recognition.

For the other relevant video-based works, Song *et al.* [51] proposed a multimodal stochastic RNN for video captioning, where both visual and textual features were explored via multimodal LSTM to capture the high-level representation, and a backward stochastic LSTM was developed to

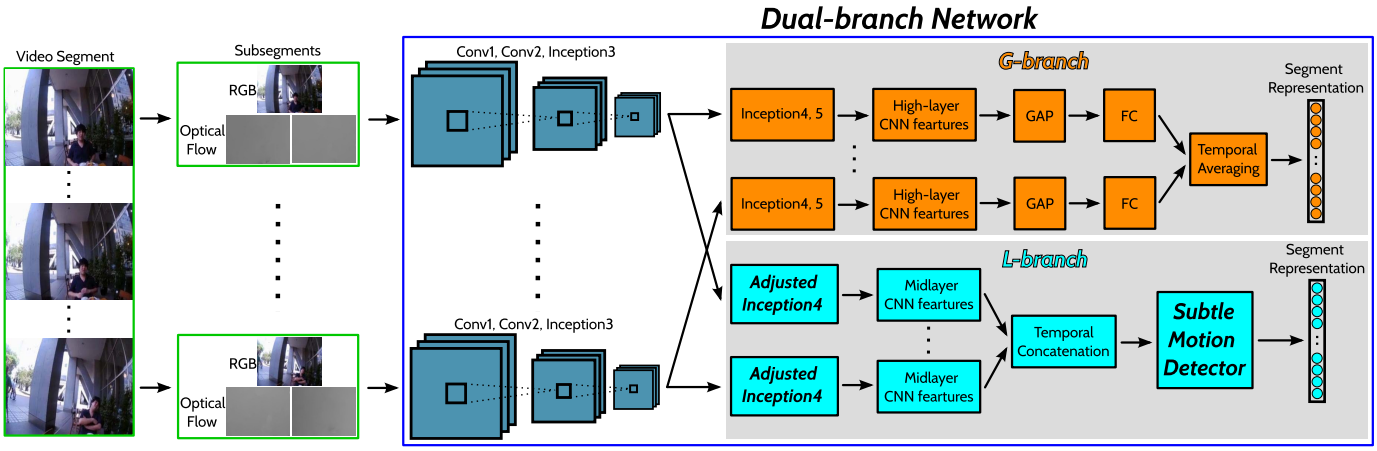


Fig. 2. An illustration of the overall architecture of the dual-branch network to obtain the video-segment representation. The proposed network first divides an input video into several video segments and further divides each segment into several subsegments for RGB frames and optical flows. The proposed network contains two branches: the G-branch and the L-branch, where the G-branch directly uses high-layer CNN features to compute the segment representation, and the L-branch includes a subtle motion detector to further learn midlayer CNN features for the segment representation.

support uncertainty propagation by introducing latent variables. Gao *et al.* [52] proposed a hierarchical LSTM with adaptive attention for video captioning, in which stacked LSTMs extracted both the visual and language information, while adaptive attention decided whether to rely on visual or language information. Different from these LSTM-based methods, which applied LSTM on high-layer CNN features for video-level analysis, the proposed method uses TSN [4] for temporally modeling videos and focuses on exploring midlayer CNN features for microaction recognition.

III. Proposed Method

In this section, we introduce the details of the proposed dual-branch network with a subtle motion detector for microaction recognition. We first describe the overall architecture of the network, which utilizes both high-layer and midlayer CNN features for classification. Then, we introduce the subtle motion detector by discussing the discriminative spatial-temporal feature learning on midlayer CNN features, the activation fusion, and the parallel multiplier attention. Finally, we elaborate on the loss function of the whole network.

A. Dual-Branch Network

Given an input microaction video V, we first sparsely sample it for several short segments $\{v_1, v_2, \cdots, v_N\}$ of equal time duration and then extract RGB frames and stacked optical flows from each segment to obtain the appearance and motion representation of the video. This sampling strategy has been shown to be effective for long-term temporal modeling [4], [12]. We then adopt the popular two-stream 2D CNN framework [4], [30] to process the RGB frames and the stacked optical flows. For the spatial or temporal stream, we propose a novel dual-branch network with a subtle motion detector for better recognizing microactions. The overall architecture of our dual-branch network is shown in Fig. 2. For one video segment, we further divide it into T subsegments and employ the state-of-the-art 2D CNN architecture (e.g., InceptionNet [16] or ResNet [15]) as the backbone to extract the spatial-temporal features from each subsegment, while the layers on all the subsegments share weights. Here, we use the batch normalization inception network (BN-Inception) [20] as the backbone for illustration. Notice that our dual-branch design is not limited to any specific CNN architecture. After the inception 3 (i.e., the inception (3a-3c)) layers in BN-Inception, we further design two branches, the G-branch and L-branch, to utilize high-layer and midlayer CNN features, respectively.

In the G-branch, we follow the temporal segment network (TSN) [4] to use the original BN-Inception for extracting high-layer CNN features and generate classification scores over all the action classes for each subsegment. Then, we average the classification scores from all subsegments to obtain the video-segment representation.

In the L-branch, we first adjust the original inception4 (i.e., the inception (4a-4e)) layers in BN-Inception to produce midlayer CNN feature maps with higher resolution for preserving more precise spatial information. We conduct this adjustment because the middle CNN layers in the recent state-of-the-art CNN architectures, such as the inception (4a-4e) layers in InceptionNet and the conv4_x layers in ResNet, have relatively small sizes for their output feature maps (i.e., 14×14), which may be insufficient for detecting small discriminative spatial-temporal features in microaction videos. Figure 3 shows the adjusted inception4 in the L-branch. For simplicity, the batch normalization (BN) [20] layers and rectified linear unit (ReLU) [53] layers, which are placed after the convolutional layers, are not displayed here. To preserve as much architecture of the inception 4 layers as possible, we only adjust the inception (4e) layers, which are the only group of layers where downsampling is performed. As shown in Fig. 3, we employ three changes on the original inception (4e) to avoid downsampling: 1) using a smaller stride (i.e., 1) for the last convolutional layer on the left side; 2) removing the last convolutional layer in the middle; and 3) removing the pooling layer on the right side. The resulting midlayer CNN

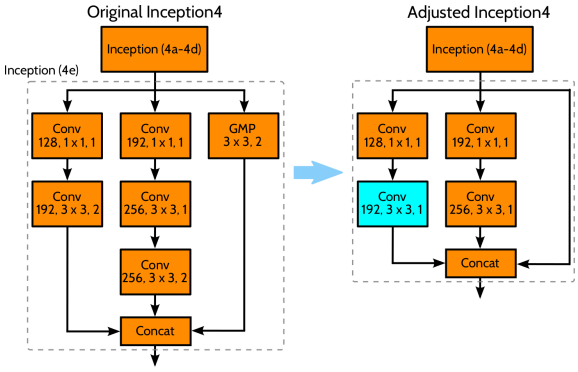


Fig. 3. An illustration of constructing the adjusted Inception4 layers. We denote convolutional layers as Conv, e.g., Conv $128, 1 \times 1, 1$ for a convolutional layer with 128 output channels, 1×1 filter, and stride 1. We denote the global max-pooling layer as GMP, e.g., GMP, $3 \times 3, 2$ for a global max-pooling layer with 3×3 kernel and stride 2.

features from the adjusted inception4 have the same number of channels but twice the spatial size as those from the original inception4.

We denote $\mathbf{F}_t \in R^{C \times H \times W}$ as midlayer CNN features of the t-th subsegment extracted from the adjusted inception4, where $t = 1, 2, \cdots, T$, C, H and W are the number of channels, height and width of the feature map, respectively. We temporally concatenate the features from all subsegments as $\mathbf{F} \in R^{C \times T \times H \times W}$. In the next subsection, we propose a novel module, named the subtle motion detector, to learn discriminative yet subtle motion patterns from these extracted midlayer CNN features and produce the video-segment representation.

In either the G-branch or L-branch, we follow the TSN to combine the segment-level representation for the video-level representation. Finally, we compute the video prediction $\mathcal{P}$ by merging the two branches as follows:

$$\mathcal{P} = \mathcal{S}\left(\mathcal{U}\left(F_g^1, F_g^2, \cdots, F_g^N\right)\right) + \mathcal{S}\left(\mathcal{U}\left(F_l^1, F_l^2, \cdots, F_l^N\right)\right), \tag{1}$$

where $F_g^1, F_g^2, \cdots, F_g^N$ and $F_l^1, F_l^2, \cdots, F_l^N$ are the representations of N segments for the G-branch and L-branch, respectively. $\mathcal{U}$ is the aggregation function for combining the segment-level representations into video-level representations. Here, we choose average pooling as the aggregation function $\mathcal{U}$, since it is shown to be the most effective method in TSN. $\mathcal{S}$ is the $Softmax$ function that gives the prediction probability for each action class of the video. The two branches are complementary to each other, and their respective predictions are fused via summation with equal weights.

B. Subtle Motion Detector

To further learn the midlayer CNN features $\mathbf{F} \in R^{C \times T \times H \times W}$ of a video segment, we propose a novel subtle motion detector, which consists of three key components: discriminative spatial-temporal feature learning, activation fusion, and parallel multiplier attention. Figure 4 shows the overall construction of the subtle motion detector.

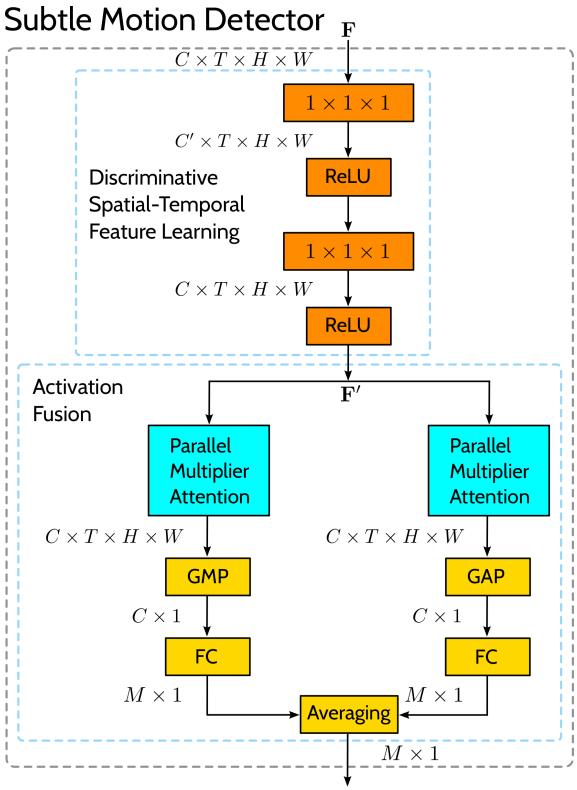


Fig. 4. An illustration of constructing the subtle motion detector. Here we show the version of averaging fusion. The $1 \times 1 \times 1$ block denotes the convolutional layer with $1 \times 1 \times 1$ filters and stride 1. ReLU, GMP, GAP and FC denotes the rectified linear unit, global max-pooling, global average pooling, and fully connected layers, respectively. Notice that, the batch normalization layers after the convolutional layers are not displayed for simplicity.

Discriminative Spatial-Temporal Feature Learning. The midlayer CNN features $\mathbf{F} \in R^{C \times T \times H \times W}$ can be considered as a group of $C \times 1 \times 1 \times 1$ cubes, where each cube at a fixed location of x represents a corresponding small spatial-temporal region in the original microaction video. Assume that we could learn a response for a cube at each location – the higher the activation, the more discriminative its representing region. Here, we choose the smallest receptive field, i.e., $1 \times 1 \times 1$, because microactions are highly localized and usually involve only subtle movement of small body parts. As shown in Fig. 4, we employ two convolutional layers with a $1 \times 1 \times 1$ filter and stride 1, where each layer is followed by ReLU and BN layers. The first convolutional layer has $C' = \frac{C}{8}$ output channels, while the second layer has C output channels. The output feature maps keep the same dimension $C \times T \times H \times W$, while the activation at each cube is further learned. Compared with just using one convolutional layer with C output channels, our design reduces the computations to a quarter of the original computations while increasing the complexity of the learned features by adding nonlinearities, i.e., the additional ReLU layer. We refer to this learning progress as discriminative spatial-temporal feature learning.

Activation Fusion. After having the activation at each location of the midlayer CNN feature maps, the discriminative spatial-temporal region can be focused on by selecting the cube with the highest activation, which can be achieved via

the global max-pooling (GMP) operation. This is shown to be very effective for image classification as in [18]. However, for video action classification, the discriminative spatial-temporal regions usually involve different spatial locations in the temporal domain due to the movement of human body parts over time. Therefore, the discriminative spatial-temporal region with the maximum activation obtained by GMP over the 3D feature map volume may not be sufficient for representing an entire video segment.

To address this issue, we propose the activation fusion as shown in Fig. 4, which utilizes GMP and global average pooling (GAP) to extract max and average activations $A_{max} \in R^{C\times 1}$ and $A_{avg} \in R^{C\times 1}$, respectively. Compared with GMP, which only considers the region with maximum activation, GAP considers the contributions in all regions. After GMP or GAP operation, a fully connected (FC) layer is used to compute the classification scores. The two FC layers do not share weights because the max and averaged activations usually refer to different types of information. Before the pooling operations, we insert an attention module, which will be discussed in the next subsection, to ensure that the feature maps concentrate on key areas and avoid involving the activations from nondiscriminative regions during the pooling operation. Finally, to combine these two activations for classification, we propose three types of fusion schemes to compute the final classification scores $Q \in R^{M\times 1}$, where M is the number of action classes.

Averaging. In this fusion scheme, we perform an average operation over the classification scores for each class. The final classification scores are computed as follows:

$$Q = \frac{1}{2}\left(FC_{max}\left(A_{max}\right) + FC_{avg}\left(A_{avg}\right)\right), \tag{2}$$

where Q denotes the classification score of all action classes for a video segment and FC_{max} and FC_{avg} denote the FC layers for max and averaged activations, respectively. The basic motivation behind averaging is to leverage both of these activations for microaction recognition by considering their contributions equally for the final prediction. In this way, both the max and averaged activations learned from midlayer CNN features are jointly modeled.

Maxing. The maxing is one alternative scheme to the averaging, where we select the max classification score for each class. The final classification scores are computed as follows:

$$Q = Max\left(FC_{max}\left(A_{max}\right), FC_{avg}\left(A_{avg}\right)\right). \tag{3}$$

The intuition of maxing is to seek a single representation (either the max or averaged activation) for each action class and only preserve the strongest activation as the final response for this category. Compared with the averaging, the maxing emphasizes one activation and neglects the other activation.

Concatenation. In this fusion scheme, we concatenate the max and averaged activations and then use an FC layer to obtain the classification scores for an action class by computing the weighted sum from all the max and averaged activations. The final classification scores are computed as follows:

$$Q = FC\left(Concat\left(A_{max}, A_{avg}\right)\right). \tag{4}$$

The basic assumption underlying concatenation is that these two types of activations may play unequal roles in recognizing microactions. This fusion scheme is designed to compute different weights for these activations. Compared with the previous averaging and maxing fusion schemes, the concatenation enables a flexible selection of the max and averaged activations.

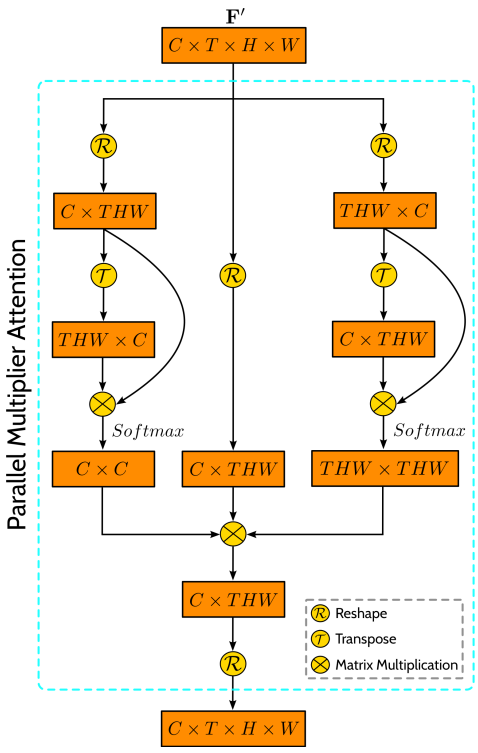


Fig. 5. An illustration of constructing the parallel multiplier attention. The left and right sides compute the channel and spatial-temporal attention maps, respectively. The middle reshapes the input feature maps. Finally, they are merged via a multiplier gate.

Parallel Multiplier Attention. Before applying pooling operations, we propose the parallel multiplier attention to further refine the convolutional feature maps $\mathbf{F}' \in R^{C\times T\times H\times W}$ resulting from discriminative spatial-temporal feature learning. This attention module is designed to focus features on salient regions in both the channel and spatial-temporal domains. In this way, the following pooling operations are less influenced by irrelevant regions, and the pooled patterns are more robust and consistent. Specifically, the parallel multiplier attention computes the channel attention map and the spatial-temporal attention map in parallel and then applies the attention maps on the original features via a multiplier gate, as shown in Fig. 5.

Channel attention map. Channel attention usually tells the network "what" to look at. By applying the channel attention, we can enable the features to concentrate on "what" is meaningful given an input image [54]. Inspired by the nonlocal operation [50] in the spatial-temporal domain, we extend this operation to the channel domain and compute the channel

attention as follows. Specifically, we reshape the input feature $\mathbf{F}'$ into a matrix $\mathbf{X_1} \in R^{C\times THW}$ and obtain its transpose matrix $\mathbf{X_1^T} \in R^{THW\times C}$. We compute the channel attention map by multiplying $\mathbf{X_1}$ and $\mathbf{X_1^T}$, followed by the $Softmax$ function over the second dimension, as follows:

$$\mathcal{Z}_c = Softmax\left(\mathbf{X_1}\mathbf{X_1^T}\right), \tag{5}$$

where $\mathcal{Z}_c \in R^{C\times C}$. The value at the i-th row and j-th column of $\mathcal{Z}_c$ measures the relationship between the i-th and j-th channels in $\mathbf{F}'$. Note that we do not follow [50], which applied linear embedding (convolutional layers) to transform the features before the reshaping operations because the convolutional layers may undesirably change the original channel information.

Spatial-temporal attention map. The spatial-temporal attention map tells the network "where" to look over space and time. By applying spatial-temporal attention, we can enable the features to concentrate on salient areas. We follow [50] to compute the spatial-temporal attention map. Specifically, we reshape the input feature $\mathbf{F}'$ into a matrix $\mathbf{X_2} \in R^{THW\times C}$ and obtain its transpose matrix $\mathbf{X_2^T} \in R^{C\times THW}$. We compute the spatial-temporal attention map by multiplying $\mathbf{X_2}$ and $\mathbf{X_2^T}$, followed by the $Softmax$ function over the second dimension, as follows:

$$\mathcal{Z}_{st} = Softmax\left(\mathbf{X_2}\mathbf{X_2^T}\right), \tag{6}$$

where $\mathcal{Z}_{st} \in R^{THW\times THW}$. The value at the i-th row and j-th column of $\mathcal{Z}_{st}$ measures the relationship between the i-th and j-th positions in $\mathbf{F}'$. Note that we also do not apply linear embedding to the features before the reshaping operations. This guarantees that the spatial-temporal and channel attention maps are computed based on the same visual information.

After having the channel and spatial-temporal attention maps, we apply them to the reshaped input features $\mathbf{X_1}$ through a multiplier gate. The refined features $\mathbf{F_r}$ are computed as follows:

$$\mathbf{F_r} = \mathcal{Z}_c\mathbf{X_1}\mathcal{Z}_{st}, \tag{7}$$

where $\mathbf{F_r} \in R^{C\times THW}$. In the end, we reshape $\mathbf{F_r}$ back to its original shape $C \times T \times H \times W$. These final output features have a concentration on discriminative channels and spatial-temporal regions, which helps the latter activation fusion to avoid nondiscriminative areas during the pooling operations.

C. Loss Function

The proposed dual-branch network computes the standard categorical cross-entropy losses – $\mathcal{L}_g$ and $\mathcal{L}_l$ for the G-branch and L-branch, respectively. We combine the two losses of both branches to compute the final loss $\mathcal{L}$. For an input video V with the ground-truth labels $\{y_1, y_2, \cdots, y_M\}$ regarding M action classes, the final loss function and two branches' loss functions are defined as:

$$\mathcal{L} = \mathcal{L}_g + \mathcal{L}_l, \tag{8}$$

$$\mathcal{L}_g = -\sum_{i=1}^{M} y_i \left(E_g^i - \log\sum_{j=1}^{M}\exp\left(E_g^j\right)\right), \tag{9}$$

$$\mathcal{L}_l = -\sum_{i=1}^{M} y_i \left(E_l^i - \log\sum_{j=1}^{M}\exp\left(E_l^j\right)\right), \tag{10}$$

where $E_g = \mathcal{U}\left(F_g^1, \cdots, F_g^N\right)$ and $E_l = \mathcal{U}\left(F_l^1, \cdots, F_l^N\right)$ are the video-level representations computed by the G-branch and L-branch, respectively; y_i is the ground-truth label of the i-th action class; E_g^i and E_g^j are the i-th and j-th dimensions of E_g; E_l^i and E_l^j are the i-th and j-th dimensions of E_l. Here, we simply sum the two losses with equal weights.

IV. Experiments on Microaction Videos

In this section, we first introduce the evaluation datasets and then describe the experimental setup. Finally, we report the experimental results on these datasets.

A. Datasets

PEV dataset. Paired egocentric video (PEV) is an existing microaction video dataset [1], which contains 911 pairs of first-person (actor's view) and second-person (observer's view) videos. Each video pair includes one microaction performed by a person A and viewed by another person B. Since the proposed method focuses on recognizing microactions present in a video, we use the 911 second-person videos for performance evaluation in this section. In the PEV dataset, there are 7 microactions: *pointing*, *attention*, *positive*, *negative*, *passing*, *receiving*, and *gesture*, and each action has 182, 97, 159, 40, 119, 143, and 171 videos, respectively. Each video has 90 frames with 60 fps and 320 × 180 spatial resolution. Sample videos of the microactions are shown in Fig. 6.

Our newly collected dataset. In the existing PEV dataset [1], the microaction performer is always standing or sitting, i.e., only the microactions of interest are presented. However, in many real applications, microactions of interest may also be combined with larger general motions, such as walking, and it is highly desirable to recognize microactions of interest in such cases. To the best of our knowledge, there are no such publicly available video datasets. To address this issue, we collect a new video dataset and release it to the public. For this new dataset, we choose 10 microactions that are very common in daily life: *Applaud*, *HandSalute*, *Nod*, *ShakeHead*, *PutPalmsTogether*, *HeadScratch*, *ShrugShoulders*, *Stop*, *ThrowUpHands*, *Waving*. For convenience, we call this newly collected dataset Micro-Action 10 (MA10), in which each video only contains one microaction of interest. For each action, 100 videos were recorded for mixing with each of four general actions, including walking, running, standing still, and sitting. The sample video frames are shown in Fig. 7. In each video, only one person (the performer of action) is present, and the microaction is performed 2 or 3 times in the video. In these videos, the performers are always walking when performing the microaction of interest. The average length of our collected videos is approximately 4.5 seconds. All videos are recorded by GoPro cameras with 30 fps. Each frame is downsampled to the spatial resolution of 340 × 256 pixels. Fifteen different performers and ten different camera wearers are used to collect the videos in

Fig. 6. Exemplar videos for the 7 different microactions in the existing dataset PEV [1], where each column shows two sample frames of one action video: pointing, attention, positive, negative, passing, receiving and gesture (from left to right).

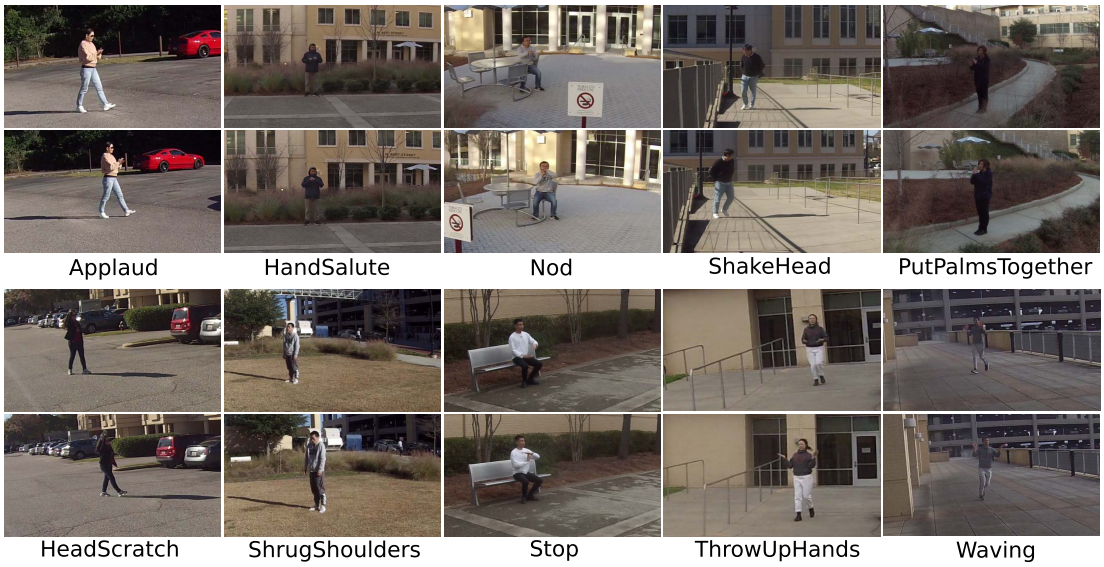


Fig. 7. Exemplar videos for the 10 different microactions in our newly collected dataset MA10, where two sample frames are shown for each action video. Top: Applaud, HandSalute, Nod, ShakeHead and PutPalmsTogether. Bottom: HeadScratch, ShrugShoulders, Stop, ThrowUpHands, and Waving.

fifteen different scenarios. The camera wearers freely move in recording, which leads to the complexities of camera motion, view difference, lighting change, etc.

B. Experimental Setup

Network input. In training, we uniformly divide a sample video into $N = 3$ video clips and randomly choose a segment with $L = 20$ consecutive frames from each clip. We evenly divide each segment into $K = 4$ subsegments and resize each video frame to a spatial resolution of 340×256. For each subsegment, the spatial-stream and temporal-stream networks take the input of a single RGB frame and optical flow stacks, respectively, where the optical flow is computed via the TVL1 optical flow algorithm [55]. Warped optical flow stacks are also used for temporal stream networks to compensate for camera motion. Specifically, the warped optical flow is computed between two adjacent frames by using their estimated homography matrix as in [2].

Network training. The parameters of the proposed network are initialized with the weights pretrained on ImageNet [56]. The stochastic gradient descent algorithm is used to train the network. The minibatch size is set to 32, weight decay is set to 0.0005, momentum is set to 0.9, and initial learning rate is set to 0.001. For the spatial stream network, the learning rate is decreased by a factor of 0.1 after 50 and 100 epochs. The training process stops after 150 epochs. For the temporal stream network with the input modality of optical flow or warped optical flow stacks, the learning rate is decreased by a factor of 0.1 after 150 and 250 epochs, and the training stops after 300 epochs. To effectively train the network, we follow the good practice in [4]: regularization: partial BN and extra dropout layer after the global pooling layer in BN-Inception; 2) data augmentation: center cropping and scale jittering. We use two NVIDIA Tesla V100 GPU cards with 32 GB memory for each card to train the network. It takes approximately 1.5 hours and 9 hours to train the spatial and temporal streams, respectively, on the PEV dataset.

Network testing. We follow the testing scheme of the original two-stream CNNs [30] to sample each testing video into 25 segments. We use the same GPU resource as in training to test the network. For each testing video, the spatial stream takes approximately 0.16 seconds, while the temporal stream takes approximately 0.18 seconds. We then use the weighted averaging scheme to combine the classification scores from different streams for fusion. When combining the

two streams, the weights of the spatial stream and temporal stream are assigned to 1 and 1.5, respectively. This is because motion-based input (optical flow) usually contributes more than appearance-based input (RGB) for action recognition. When both optical flow-based modalities are used, warped optical flow stacks serve as complementary modalities, and the weights of the temporal stream are set to 1 for warped optical flow stacks and 1.5 for optical flow stacks.

Evaluation scheme. For evaluation of the PEV dataset, we follow the original evaluation scheme to conduct a three-fold cross validation on all the second-person (911) videos in this dataset. For evaluation of our newly collected MA10 video dataset, we also conduct a three-fold cross validation on all (4,000) videos. Specifically, one dataset is randomly split into three subsets of similar size: two subsets are used for training, and the remaining subset is used for testing. We calculate the average accuracies over all action classes as the recognition performance for each dataset. All experiments are run on PyTorch [57].

C. Results

In the result analysis, we use the G-branch of our dual-branch network as the baseline method, which is actually TSN [12]. Note that the original TSN takes the input of three segments with five frames in each segment. A little different from this setting, the G-branch in our dual-branch network uses TSN with 12 segments (subsegments constructed in this paper), each of which contains five frames. We show that the performance can be significantly improved by the dual-branch network, which includes both the L-branch and the G-branch. Specifically, we first perform the ablation study on different parts of the L-branch, including the midlayer adjustment (i.e., the adjustment on inception 4), along with three components of the subtle motion detector. All experiments in this ablation study are performed on the PEV dataset. We then study the impact of modality fusion and try different CNN backbones for both microaction datasets. We also show comparison results and per-class analysis on both microaction datasets.

Impact of Midlayer Adjustment and Discriminative Spatial-Temporal Feature Learning. We first study their impact by directly connecting discriminative spatial-temporal feature learning with pooling and fully connected layers for classification in the L-branch without considering the other two components in the subtle motion detector. We conduct experiments on different input modalities, including RGB frames, optical flow stacks, and warped optical flow stacks, which are denoted as RGB, OF, and WOF, respectively. We choose the G-branch as the baseline method and apply several variations on the L-branch to study the impact: 1) G-branch + L-branch w/o Extras, where the L-branch directly uses the midlayer CNN features followed by pooling and fully connected layers for classification, without involving extra modules; 2) G-branch + L-branch w/DPD, where the L-branch uses the discriminative patch detector (DPD) [18], which contains the 1×1 convolutional layer and max-pooling layer; 3) G-branch + L-branch w/MLA, where the L-branch uses the proposed midlayer adjustment (MLA); 4) G-branch + L-branch w/DFL, where the L-branch uses the proposed discriminative spatial-temporal feature learning (DFL); 5) G-branch + L-branch w/MLA+DFL, where the L-branch uses both the MLA and DFL. The recognition accuracies on the PEV dataset are shown in Table I.

TABLE I

PERFORMANCE OF THE G-BRANCH AND THE L-BRANCH WITH SEVERAL VARIATIONS, ON THE PEV DATASET

Modality	RGB	OF	WOF
G-branch	58.2%	77.2%	74.2%
G-branch + L-branch w/o Extras	59.0%	77.7%	75.4%
G-branch + L-branch w/ DPD	59.2%	77.9%	75.6%
G-branch + L-branch w/ MLA	59.8%	78.3%	77.5%
G-branch + L-branch w/ DFL	60.3%	78.5%	77.6%
G-branch + L-branch w/ MLA+DFL	**61.2**%	**79.0**%	**78.1**%
L-branch w/o Extras	52.3%	70.6%	69.1%
L-branch w/ MLA+DFL	58.5%	75.6%	74.3%
G-branch + L-branch w/ DC	59.3%	77.9%	75.8%
G-branch + L-branch w/ DC+DFL	60.5%	78.6%	77.8%

As we can see, the performance can be slightly improved by directly using midlayer CNN features for classification (i.e., G-branch + L-branch w/o extras). The performance improvement brought by involving DPD in the L-branch is also small. These results show that directly using midlayer CNN features or embedding DPD into an action recognition framework, which has been shown to be effective for image classification, cannot lead to large performance improvement in handling microaction videos. In contrast, the performance improvement is larger and more noticeable by using the proposed MLA or DFL. Among all the methods, G-branch + L-branch w/MLA+DFL obtains the best performance. This verifies the usefulness of the proposed MLA, which preserves more precise spatial information for extracting midlayer CNN features, and shows the effectiveness of DFL, which learns the highly localized motion patterns from midlayer CNN features for microaction.

We also study the impact of using L-branch alone. Two variations are considered: L-branch w/o Extras, where only the L-branch without extra modules is used; L-branch w/MLA+DFL, where only the L-branch with MLA and DFL is used. The latter variation shows a much larger improvement over the former, which verifies the effectiveness of the proposed MLA and DFL. When combining these variations with the baseline G-branch, the performance is largely improved, which further shows the usefulness of the proposed dual-branch network.

To further justify the proposed midlayer adjustment, we also try the approach of applying an upsampling operation as deconvolution in the original inception 4 and see its results. Specifically, we insert a deconvolutional layer with a 3×3 kernel and stride 2 at the end of the original inception 4 to upsample its output feature maps. The resulting feature maps are twice as large as the original feature maps. With this, we apply two additional variations on the L-branch: 8) G-branch + L-branch w/DC, where the L-branch uses deconvolution (DC) for upsampling; 9) G-branch + L-branch w/DC+DFL, where the L-branch uses DC and DFL. As we

TABLE II

PERFORMANCE OF THE PROPOSED METHOD WITH THE PARALLEL MULTIPLIER ATTENTION ON THE PEV DATASET

Modality	RGB	OF	WOF
DFL	61.2%	79.0%	78.1%
DFL w/ Channel Attention	61.9%	79.3%	78.5%
DFL w/ Spatial-Temporal Attention	62.4%	79.5%	78.7%
DFL w/ Parallel Multiplier Attention	**62.9**%	**79.7**%	**79.1**%

TABLE III

PERFORMANCE OF THE PROPOSED METHOD WITH THE MAX/AVERAGED ACTIVATION, AND THEIR DIFFERENT TYPES OF FUSIONS ON THE PEV DATASET

Modality	RGB	OF	WOF
Max Activation	62.9%	79.8%	79.1%
Averaged Activation	61.2%	79.7%	78.6%
Averaging Fusion	63.5%	80.3%	79.8%
Maxing Fusion	63.4%	80.0%	79.6%
Concatenation Fusion	**63.9**%	**80.6**%	**80.1**%

TABLE IV

COMPLEXITY ANALYSIS. #PARAM. DENOTES THE NUMBER OF PARAMETERS; FLOPS/VIDEO DENOTES THE NUMBER OF FLOATING-POINT OPERATIONS PER VIDEO

Method	#Param.	FLOPs/Video
G-branch	10.28 M	34.95G×3
G-branch + L-branch w/o Extras	10.29 M	34.98G×3
G-branch + L-branch w/MLA	10.51 M	35.69G×3
G-branch + L-branch w/MLA+DFL	10.78 M	36.54G×3
G-branch + L-branch w/MLA+DFL+PMA	10.78 M	42.62G×3
G-branch + L-branch w/MLA+DFL+PMA+AF	10.79 M	42.64G×3

can see from the last two rows in Table I, the recognition accuracies drop when replacing MLA with DC, which verifies the effectiveness of the proposed MLA.

In the following, we use G-branch + L-branch w/MLA+DFL for the remaining experiments since it leads to the best results.

Impact of Parallel Multiplier Attention. The parallel multiplier attention is built upon the output from discriminative spatial-temporal feature learning (DFL). To study its impact, we conduct experiments by applying the two components of parallel multiplier attention (i.e., channel attention and spatial-temporal attention) and parallel multiplier attention after DFL. As we can see from Table II, the performance can be improved via either channel or spatial-temporal attention. By combining these two components in parallel multiplier attention, the performance is further improved. These results verify the effectiveness of the proposed parallel multiplier attention, which can avoid the subsequent pooling operations to extract information from nondiscriminative regions.

Impact of Activation Fusion. The activation fusion is built upon the discriminative spatial-temporal feature and parallel multiplier attention. To study its impact, we conduct experiments with three fusion schemes – averaging, maxing, and concatenation, in comparison with using only max or average activation. The recognition accuracies on the PEV dataset are shown in Table III. As can be seen, the max activation contributes more than the averaged activation for performance improvement. Our explanation is that microactions are usually highly localized so that the max-responded information is relatively more important than the globally averaged information. Concatenation fusion achieves the best performance among all three fusion schemes. This is because the concatenation fusion learns the weights for each max/averaged activation by training, while the other two fusion schemes actually use handcrafted weights. These results verify the effectiveness of the activation fusion, which can take advantage of both the most discriminative and global averaged midlayer CNN representations for microaction recognition.

Complexity analysis. We count the increase in the number of parameters as well as the number of floating-point operations brought by all the proposed components, including midlayer adjustment (MLA), discriminative feature learning (DFL), parallel multiplier attention (PMA), and activation fusion (AF). Specifically, for the number of floating-point operations per video, we report them as the number of floating-point operations per single "block" (temporal clip with spatial crop) × the number of such blocks, by following [7]. As can be seen from Table IV, the proposed method (G-branch + L-branch w/MLA+DFL+PMA+AF) only introduces a small number of extra parameters (0.51 M) and FLOPs (7.69 G×3), when compared with the baseline G-branch. For each component, MLA, DFL, PMA and AF introduce 0.22 M, 0.27 M, 0.00 M and 0.01 M parameters, respectively, while they introduce 0.71 G×3, 0.85 G×3, 6.08 G×3 and 0.02 G×3 FLOPs, respectively. Note that PMA does not have any learnable parameters because no learnable layers (e.g., convolutional or fully connected layers) are used. It mainly involves the softmax functions and matrix multiplications in Eqs (5), (6), and (7).

Visual analysis. To interpret the subtle motion detector, we randomly select 2 microaction videos and conduct qualitative analysis for its three components, as shown in Fig. 8. Specifically, we use Grad-CAM [58] to visualize sampled video frames for each action: 1) guided Grad-CAM [58], which shows pixel-space gradient visualizations for class-discriminative information learned by discriminative feature learning (DFL); 2) attention maps, which are computed by parallel multiplier attention (PMA), and attentive frames are generated to show which body parts are emphasized. The locations of the max activations extracted from the activation fusion (AF) via global max-pooling are shown in red boxes on the original frames. We can see that the proposed PMA accurately locates the salient body parts (e.g., head and arm & hand) within each frame, while DFL captures the subtle yet discriminative information accordingly. The max activations from AF reflect the most discriminative regions, while AF further enhances the motion representation from long-term dynamics since it also fuses the information extracted from all regions across frames via global average pooling. These visual results verify the effectiveness of the proposed subtle motion detector.

Impact of Input Modalities. We study the impact of using single or multiple input modalities. Here, we use the proposed dual-branch network with the midlayer adjustment and all

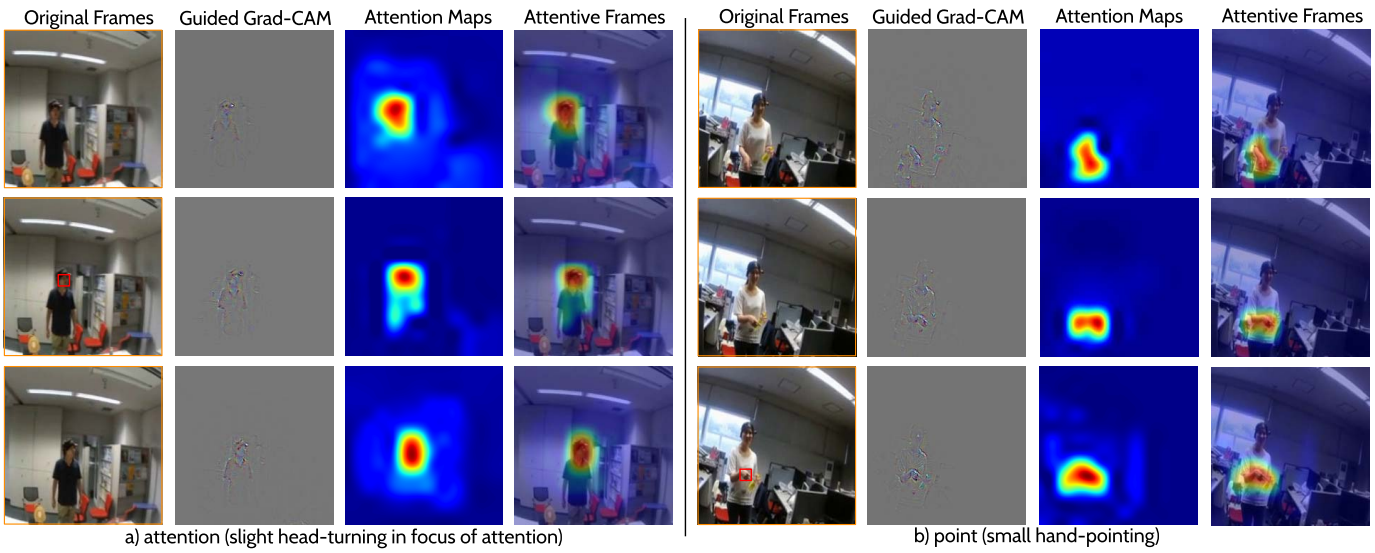


Fig. 8. Visualization of the subtle motion detector on microaction video frames. We randomly select 2 microaction videos: attention (left) and point (right). For each action video, we first show the original frames. Then, we present the guided Grad-CAM [58] to display the class-discriminative information learned by the DFL. Finally, we present the attention maps computed by the PMA, along with the attentive frames to illustrate which body parts are emphasized.

TABLE V

PERFORMANCE OF THE PROPOSED METHOD BY COMBINING DIFFERENT INPUT MODALITIES ON PEV AND MA10 DATASETS

Dataset	PEV	MA10
RGB	63.9%	60.6%
OF	80.6%	47.2%
WOF	80.1%	45.5%
RGB+OF	81.3%	62.0%
RGB+WOF	80.0%	61.6%
OF+WOF	81.5%	49.4%
RGB+OF+WOF	**83.9**%	**63.5**%

TABLE VI

PERFORMANCE OF THE PROPOSED METHOD WITH DIFFERENT CNN BACKBONES

Dataset	PEV	MA10
Proposed Method on VGGNet-16	73.8%	52.8%
Proposed Method on GoogleNet	76.9%	55.2%
Proposed Method on ResNet-152	81.6%	61.4%
Proposed Method on BN-Inception	**83.9**%	**63.5**%

three components of the subtle motion detector, since the above ablation study shows that this network achieves the best performance on the PEV dataset. We conduct experiments on both the PEV and MA10 datasets. Table V shows the recognition accuracies on these datasets. By using a single modality, RGB leads to lower accuracies than optical-flow-based modalities OF and WOF on the PEV dataset. However, it obtains higher accuracies than OF and WOF on the MA10 dataset. This may be because the action performers are smaller on MA10 than on PEV. WOF shows slightly worse performance than OF. Our explanation is that WOF uses a homography transform to compensate for camera motion, and the homography transform may not accurately represent camera motion, especially when the video background is not planar [59]. By using multiple modalities, the performance can be further improved because these modalities are complementary to each other. The proposed network achieves the best performance when using all three modalities. We use the combination of RGB, OF, and WOF for the remaining experiments, since they lead to the best performance.

Impact of Different CNN Backbones. The above experiments are conducted by using BN-Inception [20] as the base CNN architecture. Here, we also conduct experiments with other popular CNN backbones, including VGGNet-16 [60], GoogleNet [16], and ResNet-152 [15], on both microaction datasets to study the impact of choosing different CNN backbones. As shown in Table VI, the proposed method can be implemented on different CNN base architectures, resulting in good performance. The proposed method achieves the best performance when using BN-Inception as the base CNN architecture.

Comparison with Existing Methods. We choose six existing methods for comparison. The first method is a handcrafted-feature-based method [2], which detects the improved dense trajectories (iDT). The second one was proposed by Yonetani *et al.* [1], where multiple point-of-view features (MPOV) from paired first-person and second-person videos are used. The third method is proposed by Tran *et al.* [31], where deep 3D convolutional neural networks (C3D) are used to learn features from fixed-length video clips and the network is pretrained on Sports-1 M [29]. The fourth method is a two-stream deep-learning method [4], which develops a temporal segment network (TSN) to model the long-term temporal structure. Note that we use TSN as the baseline architecture to build the proposed network. The fifth is a recent state-of-the-art R(2+1)D-Two Stream network [13], which builds a two-stream 3D convolutional neural network with blocks of 2D spatial convolution plus 1D temporal

TABLE VII

COMPARISON RESULTS AGAINST SEVERAL EXISTING ACTION RECOGNITION METHODS ON PEV AND MA10 DATASETS. #PARAM. DENOTES THE NUMBER OF PARAMETERS; FLOPS/VIDEO DENOTES THE NUMBER OF FLOATING-POINT OPERATIONS PER VIDEO; FLEX. DENOTES THE FLEXIBLE NUMBER OF BLOCKS GENERATED IN C3D

Method	Pretrain	#Param.	FLOPs/Video	PEV	MA10
iDT [2]	None	–	–	43.0%	26.2%
C3D [31]	Sports-1M	79.0 M	296.7G×Flex.	53.8%	30.2%
MPOV [1]	None	–	–	69.0%	–
TSN [4]	ImageNet	10.3 M	8.8G×3	77.7%	45.7%
R(2+1)D-Two Stream [13]	Kinetics	63.8M	304.0G×115	65.4%	47.6%
Nonlocal [50]	Kinetics	35.3 M	282.0G×30	78.6%	55.7%
SlowFast [62]	Kinetics	32.9M	36.1G×30	76.9%	54.4%
STP [37]	ImageNet	10.3 M	35.0G×3	81.7%	53.7%
Proposed	ImageNet	10.8M	42.6G×3	**83.9**%	**63.5**%

TABLE VIII

COMPARISON RESULTS AGAINST SEVERAL EXISTING ACTION RECOGNITION METHODS ON THE HMDB51 DATASET

Method	Pretraining Dataset	Accuracy
iDT [2]	None	57.2%
MoFAP [40]	None	61.7%
VA Net [45]	ImageNet	41.3%
VideoLSTM [46]	ImageNet	56.4%
Attention Cluster [63]	ImageNet	69.2%
RSTAN [6]	ImageNet	70.5%
Interaction-aware STPA Net [49]	ImageNet	70.7%
scLSTM [34]	Sports-1M	55.1%
Two Stream [30]	ImageNet	59.4%
TDD [64]	ImageNet	63.2%
Two-Stream Fusion [65]	ImageNet	65.4%
Spatiotemp. MultiNet [17]	ImageNet	68.9%
TSN [4]	ImageNet	69.4%
Two-Stream 3D Fusion [66]	Sports-1M	70.5%
R(2+1)D-Two Stream [13]	Sports-1M	**72.7**%
Proposed	ImageNet	**72.7**%

convolution, and it achieves the best results when pretrained on the Kinetics [61] dataset. The sixth method is our earlier microaction recognition method, which employs the segment-level temporal pyramid (STP) [37] to capture more information in the temporal domain. The seventh is the nonlocal network [50], which develops space-time nonlocal operations for capturing long-range dependencies. The eighth network is the SlowFast network [62], which operates at low and high frame rates to capture spatial semantics and temporal resolution, respectively. For the results of iDT, MPOV and STP on the PEV dataset, we directly use those reported in the respective papers [1], [37]. For all the other results of the comparison methods, we use their released codes with default settings and parameters to conduct experiments and ensure that the loss converges during the training process. Note that we do not test MPOV on the MA10 dataset because MPOV requires features from first-person videos, which are not available in the MA10 dataset. The results are shown in Table VII.

We can see that deep-learning-based methods perform better than the handcrafted feature-based method due to the power of CNN. MPOV performs better than iDT and C3D due to the use of additional features from the first-person videos. STP obtains the best accuracy among all comparison methods on PEV because of the rich representation in the temporal domain. However, STP does not perform as well on MA10 as on PEV, which shows that it cannot handle the case where microactions are mixed with different general actions well. The proposed method significantly outperforms all the comparison methods on both datasets and achieves a new state-of-the-art performance. These results verify the effectiveness of the proposed method, which can effectively learn midlayer CNN features to capture discriminative subtle motion features for better recognizing microactions.

In addition to comparing the final recognition accuracies, we compare the model size and computational cost in terms of the number of parameters and floating-point operations per video for all deep-learning-based methods, as shown in the third and fourth columns of Table VII. Specifically, for the number of floating-point operations per video, we report them as the number of floating-point operations per single "block" (temporal clip with spatial crop) $\times$ the numbers of such blocks, by following [62]. Note that for the comparison method C3D, the number of blocks is flexible since a video is split into nonoverlapped 16-frame clips, i.e., the number of temporal clips may differ between videos. The proposed method significantly outperforms the baseline TSN while introducing only 0.5M more parameters. When compared with recent state-of-the-art methods for general actions, including R(2 + 1)D-Two Stream, Nonlocal, and SlowFast, the proposed method achieves superior performance while enjoying a much smaller network size and much lower computational cost (i.e., much higher speed). These results further verify the effectiveness of the proposed method.

Per-Class Analysis. We also report the performance of the proposed and comparison methods on each action class. As shown in Fig. 9, the proposed method outperforms the comparison methods on six out of seven action classes and achieves the third highest performance on the other action class on the PEV dataset. On the MA10 dataset, the proposed method outperforms the comparison methods on eight out of ten action classes and achieves the second and fourth highest performance on Nod and ThrowUpHands, respectively.

V. EXPERIMENTS ON GENERAL-ACTION VIDEOS

While the proposed method is developed for microaction recognition, we also test it on two general-action video datasets HMDB51 [10] and Kinetics [61] for a thorough evaluation. HMDB51 is a widely used benchmark for general-action recognition, which contains 6,766 videos from 51 action categories. We follow the original evaluation scheme by using three training/testing splits and report the average accuracy over these splits. The remaining experimental setup is the same as that described in section IV-B. Kinetics is a large-scale video action recognition dataset that has 300,000 trimmed human action videos from 400 action classes. We follow the standard evaluation scheme to train our model on the training set and evaluate on the validation set and report the recognition accuracy. The minibatch size is set to 64 for training, and the other experimental setup is the same as the one described in section IV-B.

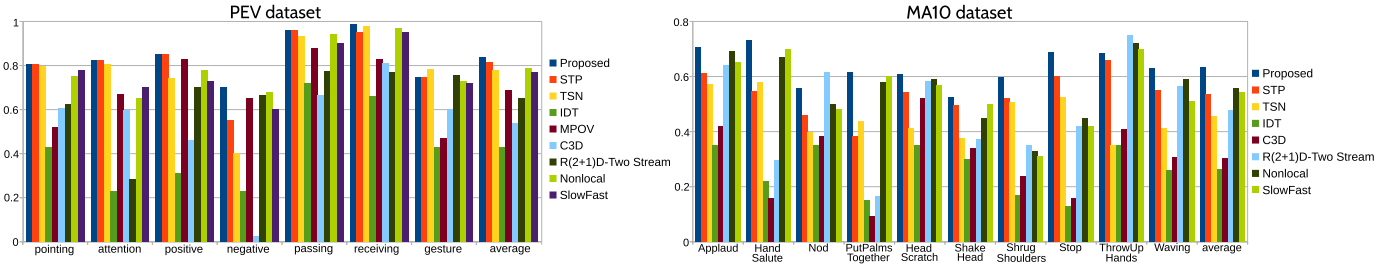


Fig. 9. Per-class action recognition accuracy of the proposed method and comparison methods on the PEV and MA10 datasets.

Results on HMDB51. We compare the proposed method with several existing methods, including two handcrafted feature-based methods: improved dense trajectories (iDT) [2] and multilevel video representation by stacking the activations of motion features, atoms, and phrases (MoFAP) [40], and five recent attention-based deep-learning methods: visual attention network (VA Net), VideoLSTM network [46], attention cluster [63], recurrent spatial-temporal attention network (RSTAN) [6] and interaction-aware spatial-temporal pyramid attention network (Interaction-aware STPA Net) [49]. We also choose eight deep-learning-based comparison methods: saliency-aware 3D-CNN with LSTM (scLSTM) [34], the original two-stream convolutional neural networks (Two Stream) [30], trajectory-pooled deep-convolutional descriptors (TDD) [64], convolutional two-stream network fusion (two-stream fusion) [65], spatiotemporal multiplier networks (Spatiotemp. MultiNet) [17], temporal segment network (TSN) [4], two-stream 3D network fusion [66] (Two-Stream 3D Fusion), and two-stream residual networks with blocks of 2D spatial convolution plus 1D temporal convolution (R(2+1)D-Two Stream) [13]. For these comparison methods, we use the results reported in their respective papers for comparison.

As we can see from Table VIII, the proposed method significantly outperforms the five attention-based methods – VA Net [45], VideoLSTM [46], Attention Cluster [63], and Interaction-aware STPA Net [49]. The proposed method also outperforms the six out of the other seven deep-learning-based comparison methods and improves the recognition accuracy by 3.3% to 72.7% when compared to the baseline TSN method. These results verify that the proposed method can also benefit general-action recognition by effectively learning midlayer CNN features for classification. The performance of the proposed method is comparable to a recent state-of-the-art R(2 + 1)D-Two-Stream method that was pretrained on Sports-1 M [29], a large-scale dataset designed for video classification.

Results on Kinetics. We compare the proposed method with four state-of-the-art approaches, TSN [4], R(2 + 1)D-Two-Stream [13], Nonlocal Net [50], and SlowFast [62]. For TSN, we use their result reported on their official website; for other methods, we use the result reported in their papers. The top-1 and top-5 classification accuracies on the validation set are shown in Table IX. As we can see, the proposed method improves the top-1 and top-5 accuracy by 1.4% and 0.6%, respectively, when compared to the baseline method TSN. These results verify the effectiveness of the proposed method. Additionally, our performance is compatible with the recent state-of-the-art approaches R(2 + 1)D-Two-Stream, SlowFast, and nonlocal Net, while our method enjoys much smaller numbers of parameters and floating-point operations.

TABLE IX

COMPARISON RESULTS AGAINST SEVERAL STATE-OF-THE-ART ACTION RECOGNITION METHODS ON KINETICS. #PARAM. DENOTES THE NUMBER OF PARAMETERS; FLOPS/VIDEO DENOTES THE NUMBER OF FLOATING-POINT OPERATIONS PER VIDEO

Method	Backbone	Pretrain	#Param.	FLOPs/Video	Top-1	Top-5
TSN [4]	BN-Inception	ImageNet	10.3 M	8.8G×3	73.9%	91.1%
R(2+1)D-Two Stream [13]	ResNet-34	Sports-1M	63.8M	304.0G×115	75.4%	91.9%
Nonlocal [50]	ResNet-50	ImageNet	35.3 M	282.0G×30	**76.5%**	**92.6%**
SlowFast [62]	ResNet-50	None	32.9M	36.1G×30	75.6%	92.1%
Proposed	BN-Inception	ImageNet	10.8M	42.6G×3	75.3%	91.7%

VI. CONCLUSION

In this paper, we developed a new deep-learning-based method for recognizing micro human actions in videos by effectively learning midlayer CNN features via a dual-branch network with a new subtle motion detector. The proposed network consists of two branches – a G-branch and an L-branch – which use high-layer and midlayer CNN features for classification, respectively. For the L-branch, we first introduced the adjusted Inception 4 for obtaining midlayer CNN features with more detailed information and then proposed a subtle motion detector to effectively learn subtle yet discriminative motion patterns. The classification scores predicted in the two branches were fused for the final classification. In the experiments, we introduced a new microaction video dataset, where the microactions of interest were mixed with relatively larger general motions. We evaluated the proposed method on the new dataset and an existing microaction dataset and achieved new state-of-the-art performances. We also tested the proposed method on two general-action video datasets with very good results.

REFERENCES

[1] R. Yonetani, K. M. Kitani, and Y. Sato, "Recognizing micro-actions and reactions from paired egocentric videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2629–2638.

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

[3] F. S. Khan, J. Xu, J. van de Weijer, A. D. Bagdanov, R. M. Anwer, and A. M. Lopez, "Recognizing actions through action-specific person detection," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4422–4432, Nov. 2015.

[4] L. Wang *et al.*, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 20–36.

[5] C. Jia, M. Shao, S. Li, H. Zhao, and Y. Fu, "Stacked denoising tensor auto-encoder for action recognition with spatiotemporal corruptions," *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1878–1887, Apr. 2018.

[6] W. Du, Y. Wang, and Y. Qiao, "Recurrent spatial-temporal attention network for action recognition in videos," *IEEE Trans. Image Process.*, vol. 27, no. 3, pp. 1347–1360, Mar. 2018.

[7] Y. Mi, K. Zheng, and S. Wang, "Recognizing actions in wearable-camera videos by training classifiers on fixed-camera videos," in *Proc. ACM Int. Conf. Multimedia Retr.*, Jun. 2018, pp. 169–177.

[8] B. Xu, H. Ye, Y. Zheng, H. Wang, T. Luwang, and Y.-G. Jiang, "Dense dilated network for video action recognition," *IEEE Trans. Image Process.*, vol. 28, no. 10, pp. 4941–4953, Oct. 2019.

[9] J. A. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

[10] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "HMDB: A large video database for human motion recognition," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 2556–2563.

[11] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," 2012, *arXiv:1212.0402*. [Online]. Available: http://arxiv.org/abs/1212.0402

[12] L. Wang *et al.*, "Temporal segment networks for action recognition in videos," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2740–2755, Nov. 2019.

[13] 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.

[14] W. Yu, K. Yang, H. Yao, X. Sun, and P. Xu, "Exploiting the complementary strengths of multi-layer CNN features for image retrieval," *Neurocomputing*, vol. 237, pp. 235–241, May 2017.

[15] 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.

[16] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 1–9.

[17] 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. 7445–7454.

[18] Y. Wang, V. I. Morariu, and L. S. Davis, "Learning a discriminative filter bank within a CNN for fine-grained recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4148–4157.

[19] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, Oct. 2016, pp. 21–37.

[20] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.

[21] I. Laptev, "On space-time interest points," *Int. J. Comput. Vis.*, vol. 64, nos. 2–3, pp. 107–123, Sep. 2005.

[22] P. Scovanner, S. Ali, and M. Shah, "A 3-dimensional sift descriptor and its application to action recognition," in *Proc. 15th Int. Conf. Multimedia MULTIMEDIA*, 2007, pp. 357–360.

[23] A. Klaeser, M. Marszalek, and C. Schmid, "A spatio-temporal descriptor based on 3D-gradients," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1–275.

[24] H. Wang, D. Oneata, J. Verbeek, and C. Schmid, "A robust and efficient video representation for action recognition," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 219–238, Sep. 2016.

[25] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 886–893.

[26] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 1932–1939.

[27] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *Int. J. Comput. Vis.*, vol. 103, no. 1, pp. 60–79, May 2013.

[28] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the Fisher kernel for large-scale image classification," in *Proc. Eur. Conf. Comput. Vis.*, 2010, pp. 143–156.

[29] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.

[30] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. Conf. Neural Inf. Process. Syst.*, 2014, pp. 568–576.

[31] 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.

[32] J. Donahue *et al.*, "Long-term recurrent convolutional networks for visual recognition and description," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, pp. 677–691, Apr. 2017.

[33] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 4694–4702.

[34] X. Wang, L. Gao, J. Song, and H. Shen, "Beyond frame-level CNN: Saliency-aware 3-D CNN with LSTM for video action recognition," *IEEE Signal Process. Lett.*, vol. 24, no. 4, pp. 510–514, Apr. 2017.

[35] Y. Poleg, C. Arora, and S. Peleg, "Temporal segmentation of egocentric videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2537–2544.

[36] M. S. Ryoo, B. Rothrock, and L. Matthies, "Pooled motion features for first-person videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 896–904.

[37] Y. Mi and S. Wang, "Recognizing micro actions in videos: Learning motion details via segment-level temporal pyramid," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2019, pp. 1036–1041.

[38] A. Jain, A. Gupta, M. Rodriguez, and L. S. Davis, "Representing videos using mid-level discriminative patches," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2571–2578.

[39] W. Zhang, M. Zhu, and K. G. Derpanis, "From actemes to action: A strongly-supervised representation for detailed action understanding," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2248–2255.

[40] L. Wang, Y. Qiao, and X. Tang, "MoFAP: A multi-level representation for action recognition," *Int. J. Comput. Vis.*, vol. 119, no. 3, pp. 254–271, Sep. 2016.

[41] J. Hou, X. Wu, J. Chen, J. Luo, and Y. Jia, "Unsupervised deep learning of mid-level video representation for action recognition," in *Proc. Conf. Artif. Intell.*, Apr. 2018, pp. 6910–6917.

[42] A. Diba, A. M. Pazandeh, H. Pirsiavash, and L. V. Gool, "DeepCAMP: Deep convolutional action & attribute mid-level patterns," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 3557–3565.

[43] Y. Li, L. Liu, C. Shen, and A. V. D. Hengel, "Mining mid-level visual patterns with deep CNN activations," *Int. J. Comput. Vis.*, vol. 121, no. 3, pp. 344–364, Feb. 2017.

[44] A. Mallya and S. Lazebnik, "Learning models for actions and person-object interactions with transfer to question answering," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 414–428.

[45] S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," 2015, *arXiv:1511.04119*. [Online]. Available: http://arxiv.org/abs/1511.04119

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

[47] Z. Lin *et al.*, "A structured self-attentive sentence embedding," 2017, *arXiv:1703.03130*. [Online]. Available: http://arxiv.org/abs/1703.03130

[48] D. Li, T. Yao, L.-Y. Duan, T. Mei, and Y. Rui, "Unified spatio-temporal attention networks for action recognition in videos," *IEEE Trans. Multimedia*, vol. 21, no. 2, pp. 416–428, Feb. 2019.

[49] Y. Du, C. Yuan, B. Li, L. Zhao, Y. Li, and W. Hu, "Interaction-aware spatio-temporal pyramid attention networks for action classification," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 373–389.

[50] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7794–7803.

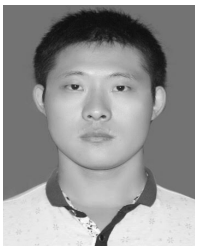
[51] J. Song, Y. Guo, L. Gao, X. Li, A. Hanjalic, and H. T. Shen, "From deterministic to generative: Multimodal stochastic RNNs for video captioning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 10, pp. 3047–3058, Oct. 2019.

[52] L. Gao, X. Li, J. Song, and H. T. Shen, "Hierarchical LSTMs with adaptive attention for visual captioning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 1, pp. 1112–1131, May 2019.

[53] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. Int. Conf. Mach. Learn.*, 2010, pp. 807–814.

[54] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, Sep. 2018, pp. 3–19.
[55] C. Zach, T. Pock, and H. Bischof, "A duality based approach for realtime TV-L^1 optical flow," in *Proc. Joint Pattern Recognit. Symp.*, 2007, pp. 214–223.
[56] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[57] A. Paszke *et al.*, "Automatic differentiation in pytorch," in *Proc. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1–4.
[58] 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," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 618–626.
[59] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.
[60] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: http://arxiv.org/abs/1409.1556
[61] 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. 4724–4733.
[62] 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.
[63] X. Long, C. Gan, G. de Melo, J. Wu, X. Liu, and S. Wen, "Attention clusters: Purely attention based local feature integration for video classification," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7834–7843.
[64] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 4305–4314.
[65] 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.
[66] X. Wang, L. Gao, P. Wang, X. Sun, and X. Liu, "Two-stream 3-D convNet fusion for action recognition in videos with arbitrary size and length," *IEEE Trans. Multimedia*, vol. 20, no. 3, pp. 634–644, Mar. 2018.

Yang Mi received the B.S. degree in information engineering from the Beijing Institute of Technology, China, in 2009, and the M.S. degree in computer science and engineering from the University of South Carolina in 2018, where he is currently pursuing the Ph.D. degree in computer science and engineering. His research interests include computer vision and machine learning.



Xingyuan Zhang received the B.S. degree in computer science and technology from Tangshan University in 2013. He is currently pursuing the Ph.D. degree with the School of Computer and Information Technology, Beijing Jiaotong University, China. In 2019, he was a Visiting Ph.D. Student with the University of South Carolina, Columbia, SC, USA. His research interests include deep learning, computer vision, and digital image processing.

Zhongguo Li received the B.S. degree in computer science and technology from the School of Computer Science and Technology, Tianjin University, China, in 2016, where he is currently pursuing the M.E. degree in computer technology with the College of Intelligence and Computing. His research interests include computer vision and deep learning.

Song Wang (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign (UIUC) in 2002. From 1998 to 2002, he also worked as a Research Assistant with Image Formation and Processing Group, Beckman Institute, UIUC. In 2002, he joined the Department of Computer Science and Engineering, University of South Carolina, where he is currently a Professor. His research interests include computer vision, medical image processing, and machine learning. He is a Senior Member of the IEEE Computer Society. He is currently serving as an Associate Editor of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, *Pattern Recognition Letters*, and *Electronics Letters*.