MESNet: A Convolutional Neural Network for Spotting Multi-Scale Micro-Expression Intervals in Long Videos

Su-Jing Wang, *Senior Member, IEEE,* Ying He, Jingting Li, *Member, IEEE,* and Xiaolan Fu, *Member, IEEE*

***Abstract*—Micro-expression spotting is a fundamental step in the micro-expression analysis. This paper proposes a novel network based convolutional neural network (CNN) for spotting multi-scale spontaneous micro-expression intervals in long videos. We named the network as Micro-Expression Spotting Network (MESNet). It is composed of three modules. The first module is a 2+1D Spatiotemporal Convolutional Network, which uses 2D convolution to extract spatial features and 1D convolution to extract temporal features. The second module is a Clip Proposal Network, which gives some proposed micro-expression clips. The last module is a Classification Regression Network, which classifies the proposed clips to micro-expression or not, and further regresses their temporal boundaries. We also propose a novel evaluation metric for spotting micro-expression. Extensive experiments have been conducted on the two long video datasets: CAS(ME)^2 and SAMM, and the leave-one-subject-out cross-validation is used to evaluate the spotting performance. Results show that the proposed MESNet effectively enhances the F1-score metric. And comparative results show the proposed MESNet has achieved a good performance, which outperforms other state-of-the-art methods, especially in the SAMM dataset.**

***Index Terms*—convolutional neural network, deep learning, detection, long videos, micro-expression spotting**

I. Introduction

MICRO-EXPRESSION (ME) is a brief, involuntary facial expression that occurs when a person conceals his or her true emotion. It was first discovered and called "micro-momentary" expressions by Haggard and Isaacs [1] in 1966. And in 1969, Ekman and Friesen [2] also reported that they found a special facial expression: micro-expression. ME is an important clue for detecting lies [3], [4], which leads to considerable interest in both academic and commercial communities. Analyzing MEs is valuable for many potential applications, such as medical care [5], law enforcement [6], political psychology [7], national security [8] and much more. Compared with the common expressions called "macro-expressions", there are three distinguishing characteristics of MEs: short duration[1], low intensity, and local movements.

It is challenging for human beings to spot and recognize such brief and subtle expressions by naked eyes [9]. Although Ekman developed the Micro-Expression Training Tool (METT) to train people's ability to analyze MEs [10], the performance is still far below a desirable level [11]. Analyzing MEs by human beings is costly and time-consuming. It requires well-trained specialists but cannot get a satisfactory performance usually. Therefore, there is an urgent need to develop an automatic micro-expression analysis system. As a result, ME analysis is an important problem not only in the field of psychology but also in the field of computer vision.

In computer vision, generally, ME analysis includes two major steps: spotting and recognition. Spotting is to find the temporal location of the ME clip in a given video. And recognition is the emotional classification of the ME clip. Essentially, ME recognition is a classification problem and is easier than ME spotting. At the early stage of ME research, some traditional feature extraction methods are used to extract features in ME recognition. These traditional methods are Gabor [12], HOG [13], optical flow [14], [15], tensor subspace analysis [16], sparse representation [17], LBP-TOP and its variations [18]–[20] etc. Therefore, a lot of works on ME recognition were published.

Compared with works on ME recognition, however, works on ME spotting is rare and important. In the real applications, if we can spot a person's ME, we can know that he or she may lie or conceal his or her genuine emotions when ME occurs. Moreover, in ME analysis, spotting is the first step and provides reliable information for subsequent analysis, such as ME recognition.

ME movement variation is described by three time points: onset, apex, and offset. Onset is the time when the ME starts. Apex is the time when the ME reaches its maximum muscular contraction. Offset is the time when the ME ends. The corresponding frames are generally labeled in datasets as the illustration in Fig.1.

According to the different kinds of outputs, ME spotting methods are divided into apex frame spotting [21]–[24] and sequence spotting. This paper concentrates on ME sequence spotting methods, which means locating ME interval (multiple

This paper is supported in part by grants from the National Natural Science Foundation of China (U19B2032, 61772511, 62061136001), in part by grants from the China Postdoctoral Science Foundation (2020M680738), and in part by grants from he National Key Research and Development Project (2018AAA0100205).

S.J Wang is with the Key Laboratory of Behavior Sciences, Institute of Psychology, Chinese Academy of Sciences, Beijing, 100101, China, and also with the Department of Psychology, University of the Chinese Academy of Sciences, Beijing, 100049, China (e-mail:wangsujing@psych.ac.cn).

Y. He and J.T Li and are with the Key Laboratory of Behavior Sciences, Institute of Psychology, Chinese Academy of Sciences, Beijing, China, 100101.

X.L Fu is with the State Key Laboratory of Brain and Cognitive Science, Institute of Psychology, Chinese Academy of Sciences, Beijing, 100101, China, and also with the Department of Psychology, University of the Chinese Academy of Sciences, Beijing, 100049, China.

[1] 1/25 to 1/5 second, the precise definition varies, but the generally accepted upper limit is 0.5 second

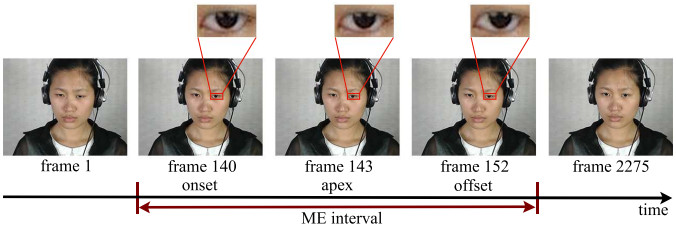


Fig. 1: Video s29_0502 with one labeled ME in CAS(ME)^2. (A ME is too subtle to be perceived by naked eyes. But when zooming the eye regions in the figure, we can see the eyelid contraction in the fear emotion.)

frames) in a given video. In other words, ME sequence spotting will identify the onset and offset of ME in a given video.

The traditional algorithm spots ME interval by comparing feature difference (FD) in a fixed-length time window. The features commonly used are listed as follows: LBP [25], HOG [26], optical flow [27]–[31], etc [32], [33]. The merit of FD methods is that the algorithm considers the temporal characteristic of ME, as the average duration of the ME determines the search window size. However, the final spotting result is obtained by setting a threshold in the FD value. The captured motion is not necessarily a ME, and it could be other kinds of facial movement which has similar intensity or duration. Therefore, the feature difference is weak in distinguishing ME from other facial movements.

Spotting methods combined with machine learning are developed in recent years. The main idea is to extract handcraft features and then use a classifier to recognize ME and non-ME frames [34]–[38]. The introduction of machine learning allows algorithms to learn features specific to ME, thereby enhancing the ability to differentiate ME among other facial movements. Yet, this kind of method is restricted by the small ME database. There are not enough ME samples to train a performant classifier. Moreover, the number of ME samples and Non-ME samples are hugely different. Thus, this kind of method also suffers the sample unbalanced problem. Also, ME spotting combined with machine learning is divided into frame-based classification [38] and interval-based classification [35], which means to determine whether the frame is a ME frame or whether the interval is a ME segment. The interval-based spotting could avoid the impact of inaccurate ME annotation, and meantime can reduce the number of true negatives in the spotting result. This type of method uses video clips as the input of the classification network, but the length of the ME sequence samples is not the same. Therefore, it requires time-domain normalization processing on the raw data. Through multi-scale analysis, such as normalization of different lengths or multi-scale video sampling, the interval-based spotting method can adapt and detect ME fragments of different lengths, and better distinguish ME from other kinds of facial movements. Here, multi-scale means the difference in the length of ME clips.

At the start of micro-expression research, since only the micro-expression sequence was considered for recognition when collecting micro-expression samples, the video clip when micro-expression occurred was recorded (maybe also a few frames before the start and a few frames after the shift), which becomes The so-called short video. Later, the importance of spotting the temporal location when the micro-expression occurred is realized. This led to the release of CAS(ME)^2 [39], where the average duration of video samples is 148s. Later, Manchester Metropolitan University also released a long videos version SAMM [40] database.

In most of the ME videos collected so far, participants were asked to keep their heads still and keep their faces expressionless. In short videos, there are very few other kinds of head movement, and the main case is the neutral expression. Meantime, the influence of the environment such as light changes on the image can also be ignored. Therefore, MEs are relatively obvious actions in short videos and are easy to be detected. In contrast, in long videos, participants inevitably have a lot of head movements such as blinking, swallowing, weak head rotation, and macro expressions. Furthermore, there will be noise caused by environmental changes. These will strongly affect the ME spotting performance. Therefore, exploring ME spotting in long videos has important requirements for the practical application of ME analysis. The community is also carrying out preliminary research in this area. For example, MEGC2019 [41] and MEGC2020 [42] set the task of ME spotting in long videos in the challenge.

There are a few attempts at spotting micro-expression on long videos. Sliding windows are used to split long videos into short videos, making it easy for the algorithm to focus on extracting the features of micro-expressions. [29], [31], [40], [43] apply the traditional way of feature differences to spot ME, and [44]–[46] combine machine learning technique to classify ME and non-ME frames. Nevertheless, the performance of ME spotting is very weak due to the influence of a large number of irrelevant movements or noise. It is still a challenge for current research to effectively extract or learn the most representative spatio-temporal features of ME from limited data and thereby accurately locate its time position on long videos.

Convolutional Neural Network (CNN) has achieved great success in numerous vision tasks. However, the penetration of this cutting-edge technology into ME analysis is very slow and small. Only Zhang *et al.* [24] proposed a relevant CNN, but it only spots the apex in the short video with only one ME.

This paper proposes a Micro-Expression Spotting Network (MESNet), which is the first work using CNN for spotting ME intervals in long videos as far as we know. MESNet includes three modules: 2+1D Spatiotemporal Convolutional Network, Clip Proposal Network, and Classification Regression Network. They extract spatial features, provide proposed clips, and further regress temporal boundaries of these proposed clips. We improve the evaluation metrics in the Second Facial Micro-Expressions Grand Challenge (MEGC2019) [41] and define more reasonable metrics. Experiment results on CAS(ME)^2 and SAMM show that despite the small number of samples and many parameters that need to be trained, MESNet achieves much better performance than state-of-the-art methods.

The main contributions of this paper can be summarized as:

- We propose a CNN-based method for spotting multi-scale ME intervals in long videos.

- There are several special tricks to deal with small sample size and sample unbalanced problems of ME.
- We propose a novel evaluation metric for ME spotting.

II. Related work

ME spotting is a location problem. In computer vision, there are two similar problems: object detection and temporal action localization.

A. Object Detection

Object detection is to determine where objects are located in a given image and which category each object belongs to [47]. Object detection includes two basic tasks: classification and regression. Due to the emergence of large-scale labeled data and increased computing power, methods based CNN make real-time and accurate object detection become more achievable.

Among these methods, R-CNN [48] is the first method for introducing CNN into object detection. Firstly, R-CNN uses selective search [49] to produce about 2,000 region proposals for a given image. Each region proposal is resized into a xed size and fed a CNN module to extract a 4096-dimensional feature as the representative feature. These representative features are fed into SVMs for multiple classes. When it belongs to a certain class, its bounding box is regressed by using a greedy non-maximum suppression. It is time-consuming that 2,000 region proposals are fed to a CNN module. Fast R-CNN [50] feed the whole image into a CNN module to produce feature maps. Fast R-CNN generates region proposals from feature maps instead of images. Each region proposal is extracted into a fixed-length feature vector with a region of interest (RoI) pooling layer. Then, each feature vector is fed into several fully-connected layers. Finally, Fast R-CNN has two parallel output layers. One is responsible for classification, and the other is responsible for regression. Faster R-CNN [51] uses Region Proposal Network (RPN) to produce region proposals.

Object detection is to locate in the spatial domain, while ME spotting is to spot in the temporal domain. Inspired by the above object detection methods, the proposed network also uses a network module to produce clip proposals in the temporal domain. Clip proposals are corresponding to region proposals in the spatial domain. The proposed network also has two parallel output layers, which are responsible for classification and regression.

B. Temporal Action Localization

Temporal action localization (TAL) is to find the temporal location of actions in a video. The great progress of CNN facilitates TAL's development. TAL's methods can be divided into three categories [52]: (1) methods performing frame or segment-level classification where the smoothing and merging steps are required to obtain the temporal boundaries [53], [54]; (2) methods using a two-step framework including proposal production, classification and boundary regression [55], [56];(3) methods developing end-to-end architectures integrating the proposal production and classification [57], [58]. In this paper, we also use a two-step framework to spot ME.

III. MESNet: Micro-Expression Spotting Network

This section will introduce the proposed Micro-Expression Spotting Network (MESNet), which is composed of three modules. The first module is a 2+1D Spatiotemporal Convolutional Network, which uses 2D convolution to extract spatial features and 1D convolution extract temporal features. The second module is a Clip Proposal Network, which gives some proposed clips for micro-expression. The last module is a Classification Regression Network, which classifies the proposed clips to ME or non-ME and further regresses their temporal boundaries.

A. 2+1D Spatiotemporal Convolutional Network

In 2+1D Spatiotemporal Convolutional Network, four 2D convolution layers with max pooling layers extract spatial features of each frame of micro-expressions and two 1D convolution layers extract temporal features of these spatial features.

Suppose that a micro-expression video clip $\mathcal{V}$ is a fourth order tensor $\mathcal{V} \in \mathbb{R}^{H\times W\times C\times N}$, where H is the height of frames of the clip, W is the width of frames of the clip, C is the number of channels, and N is frame number of the clip. For each frame $\mathcal{V}_n \in \mathbb{R}^{H\times W\times C}$ $(n = 1, 2, \ldots, N)$, a group of 2D CNN, which configuration is listed in Table I, extracts spatial features $\mathcal{F}_n \in \mathbb{R}^{\frac{H}{16}\times\frac{W}{16}\times 64}$. For a micro-expression video clip $\mathcal{V}$, there are N groups of 2D CNN with the same weights. Then $\mathcal{F}_n$ is vectorized as $\mathbf{f}_n \in \mathbb{R}^{\frac{H\times W}{4}}$. For convenience, we denote $L = \frac{H\times W}{4}$. N spatial features $\mathbf{f}_n$ consist of a matrix $\mathbf{F} = (\mathbf{f}_1, \mathbf{f}_2, \ldots \mathbf{f}_N) \in \mathbb{R}^{L\times N}$.

TABLE I: The configuration of the 2D CNN model

Layers	Kernel size	Padding	Stride
Conv1	$3 \times 3 \times 8$	2	1
Pool1	2×2	0	2
Conv2	$3 \times 3 \times 16$	2	1
Pool2	2×2	0	2
Conv3	$3 \times 3 \times 32$	2	1
Pool3	2×2	0	2
Conv4	$3 \times 3 \times 64$	2	1
Pool4	2×2	0	2

*1 Each convolutional layer is followed by the ReLU activation function.
*2 Each pooling layer is a max pooling.

Columns of $\mathbf{F}$ include spatial features, and rows of $\mathbf{F}$ include temporal information. In deep learning, a recurrent neural network (RNN) is usually used to extract temporal features. A RNN unit updates its internal hidden state $\mathbf{h_t}$ according to

$$\mathbf{h}_t = \sigma(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1}) \tag{1}$$

where $\sigma(\cdot)$ is an activation function, and $\mathbf{W}_{xh} \in \mathbb{R}^{m\times L}, \mathbf{W}_{hh} \in \mathbb{R}^{m\times m}$ are weights, and $\mathbf{h}_{t-1}, \mathbf{h_t} \in \mathbb{R}^m$ are the hidden states for step $t-1$ and t, and $\mathbf{x}_t \in \mathbb{R}^L$ is the input. Here m is the dimension of the output in each step. For convenience, all biases are omitted in this paper.

However, RNN can not well handle long videos, because of the vanishing gradient problem. This problem is well addressed by long short-term memory (LSTM) [59]. LSTM

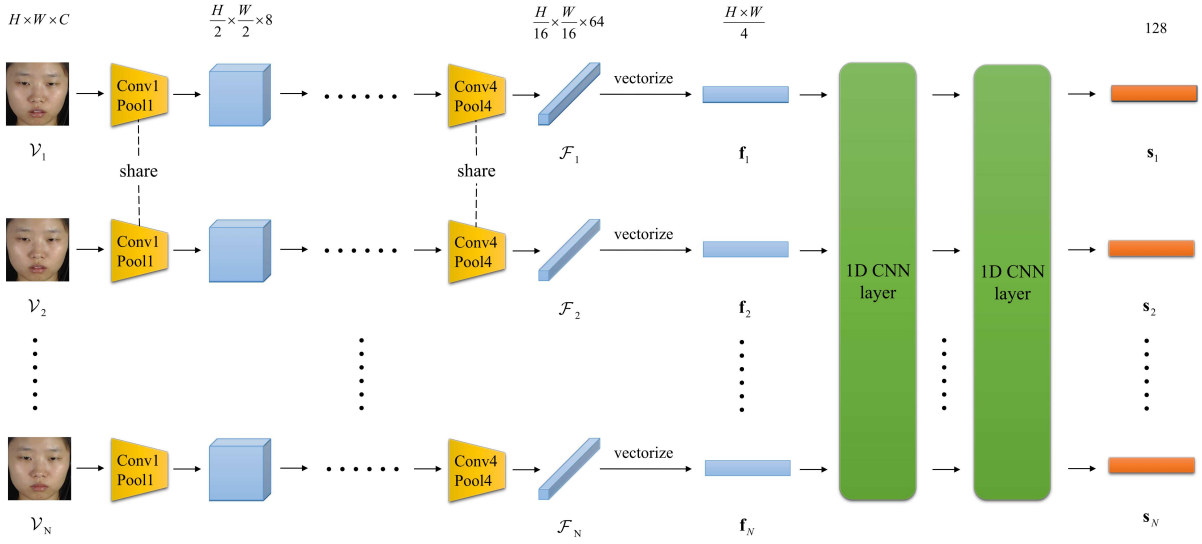
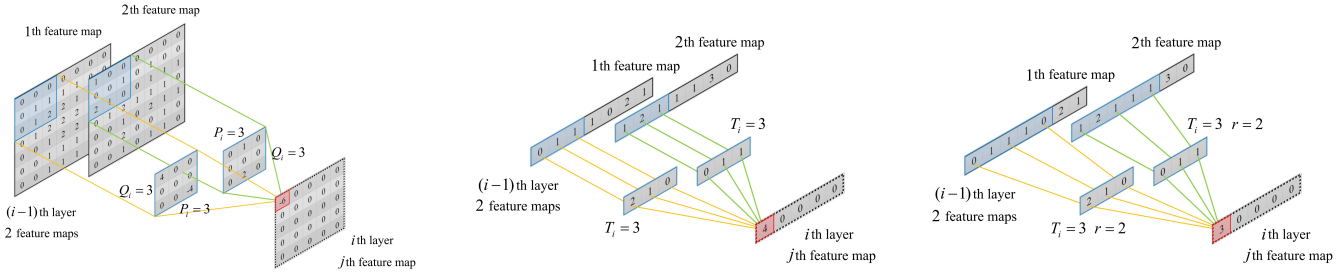


Fig. 2: 2+1D Spatiotemporal Convolutional Network.



(a) 2D convolution operation (b) 1D convolution operation (c) 1D dilated convolution operation

Fig. 3: Diagram of three kinds of convolution operation: (a) 2D convolution operation; (b) 1D convolution operation; (c) 1D dilated convolution operation. (Note: the activation function is omitted for convenience.)

incorporates memory units to make the network learn by *Forget Gate* $\mathbf{f}_t$ and *Input Gate* $\mathbf{i}_t$ when to forget previous hidden states and when to update hidden states given new information [60]. Except for $\mathbf{f}_t$ and $\mathbf{i}_t$, a LSTM unit still includes *Output Gate* $\mathbf{o}_t$ and *Input Modulation Gate* $\mathbf{g}_t$. Each *gate* has a set of weight parameters. The four gates are computed as the following formulas:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1}) \tag{2}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1}) \tag{3}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1}) \tag{4}$$

$$\mathbf{g}_t = \sigma(\mathbf{W}_{xg}\mathbf{x}_t + \mathbf{W}_{hg}\mathbf{h}_{t-1}) \tag{5}$$

Comparing the above four equations with Eq.(1), we can find that parameter numbers of RNN and LSTM are $m(L+m)$ and $4m(L+m)$ respectively. LSTM has four times the parameters of RNN. It's terrible for the small sample size problem in micro-expression analysis.

So, we use 1D CNN to extract temporal features. In 2D CNN used widely, 2D convolution operation is used to extract features from local neighborhood on feature maps in the previous layer. The notation v_{ij}^{xy} denotes the value at position (x, y) in the jth feature map in the ith layer. Then

$$v_{ij}^{xy} = f\left(\sum_k \sum_{p=0}^{P_i-1} \sum_{q=0}^{Q_i-1} w_{ijk}^{pq} v_{(i-1)k}^{(x+p)(y+q)}\right) \tag{6}$$

where $f(\cdot)$ is an activation function, k is the index over the set of feature maps in the $(i-1)$th layer connected to the current feature map, the kernel weight w_{ijk}^{pq} is the value at the position (p, q) of the kernel connected to the kth feature map, and P_i and Q_i are the width and height of the kernel, respectively [60]. The 2D convolution operation is illustrated in Fig.3(a). Similarly, 1D CNN only extract features from one temporal dimension. Formally, the value at position x on the jth feature map in the ith layer is given by

$$v_{ij}^{x} = f\left(\sum_k \sum_{t=0}^{T_i-1} w_{ijk}^{t} v_{(i-1)k}^{(x+t)}\right) \tag{7}$$

where T_i is the size of the 1D kernel along the temporal dimension, w_{ijk}^{t} is the tth value of the kernel connected to the kth feature map in the previous layer. The 1D convolution operation is illustrated in Fig.3(b).

Here, 1D CNN with two layers is implemented on rows of $\mathbf{F}$ to extract spatiotemporal features $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \ldots \mathbf{s}_N) \in$

$\mathbb{R}^{128\times N}$ of the video clip. Both layers are with the same configuration: kernel size 3, padding 2, stride 1, output channels 128, and ReLU activation function [61]. An 1D CNN layer has only $3mL$ parameters. The parameters are reduced by $m(L+4m)$ compared with LSTM. This reduction is good for dealing with the small sample size problem.

2+1D Spatiotemporal Convolutional Network is used as the backbone of MESNet. The architecture is illustrated in Fig.2. To get better performances, we pre-train the backbone to classify a given video clip into ME or non-ME. So, a classification network with three fully-connected layers is appended to the backbone. The first two fully-connected layers with a ReLU activation function have 300 neurons, respectively. To reduce overfitting, a dropout layer with 0.5 ratios follows the second fully-connected layer. The classification network classifies a given video clip into two classes. So, the last fully-connected layer with a Softmax activation function only has two neurons. $\mathbf{S}$ is flattened to a vector to feed the network. The output of the network is denoted as $\hat{\mathbf{y}}_i = (\hat{y}_1, \hat{y}_2)^T$. The true label, ME or non-ME, is converted to a one-hot vector $\mathbf{y}_i = (y_1, y_2)^T$. We optimize the following loss.

$$L_{\text{pre-train}} = -\sum_{i=1}^{2} y_i \log \hat{y}_i + \frac{\lambda}{2}\sum_{w} w^2 \tag{8}$$

where the first item represents the cross-entropy loss function, and the second item represents the L_2 regularization loss [62]. All the trainable parameters w include four 2D convolution layers, two 1D convolution layers, and two fully-connected layers, except the last fully-connected layer parameters. The regularization coefficient λ is set as 0.01. Since ME samples are too few to provide enough prior knowledge for training the parameters. So, we add the L_2 penalty item to force the weights to become small and constrain the network's complexity by introducing extra distribution prior knowledge. It helps to alleviate the overfitting problem of ME.

For pre-training the backbone, we need training samples belonging to the two classes: ME and non-ME. For ME training samples, we collect video clips labeled ME from long videos in the ME database. The number of ME training samples is denoted as n_l. Non-ME training samples include video clips labeled macro-expression and some video clips without any label. We randomly select $n_l \times r_{noL}$ video clips without any label. The frame number of each clip is l_{noL}. Here r_{noL} is the ratio of the number of video clips without any label to n_l. Similarly, We randomly select $n_l \times r_{MaE}$ video clips labeled macro-expression, which begin with the onset frame and end with the (apex+k)th frame because some macro-expressions don't have offset frames. All clips are sampled to N_{pre} frames.

B. Clip Proposal Network

Clip Proposal Network (CPN) takes spatiotemporal features $\mathbf{S}$ as input and outputs a set of clip proposals. In this paper, CPN uses five parallel sub-networks to output five clip proposals with different scales.

To get clip proposals with different scales, we need to use different sizes of receptive fields on $\mathbf{S}$. *Receptive field* of the neuron v in the ith layer is a particular region in the $(i-m)$th layer ($m = 1, \ldots, i-1$) which influences the value of v. In the 1D convolution operation, as shown in Eq.(7), the value of neuron v_{ij}^x is influenced by the region from $v_{(i-1)k}^x$ to $v_{(i-1)k}^{(x+T_i-1)}$ in the $(i-1)$th layer. So, in the $(i-1)$th layer, the size of the receptive field of v_{ij}^x is T_i. When more such layers are stacked, the size of the receptive field grows. A natural idea is expanding the receptive field's size to a desirable size by stacking more 1D convolution layers. The different number of stacked layers will get receptive fields with different sizes. With the increasing number of stacked layers, the number of weight parameters will dramatically increase. It is unsuitable for the small sample size problem of ME. Yu and Koltun [63] proposed a new type of convolution operation called *dilated convolution*, which can effectively expand the receptive field using the same kernel size as common convolution.

In 1D dilated convolution operation, the value at position x on the jth feature map in the ith layer can be computed by

$$v_{ij}^x = f\left(\sum_k \sum_{t=0}^{T_i-1} w_{ijk}^t v_{(i-1)k}^{(x+rt)}\right) \tag{9}$$

where r is the *dilation rate*, which is a positive integer. Different dilation rate settings will get different sizes of receptive fields. Fig.3(b) illustrates the 1D dilated convolution operation with the kernel size $T_i = 3$ and the dilation rate $r = 2$. We can see that the value is computed from the input segment with temporal range 5, i.e. the size of the receptive field is 5. If we use the 1D convolution with the same kernel size (see Fig.3(b)), the size of the receptive field is 3. When $r = 1$, the dilated convolution degenerates into the common convolution.

TABLE II: The configuration of the five sub-networks of Clip Proposal Network

Sub-networks	Layers	Kernel size	Dilation rate	Activation
Sub-network1	S1C1	3	1	ReLU
	S1C2	1	1	Softmax
Sub-network2	S2C1	3	1	ReLU
	S2C2	3	1	Softmax
Sub-network3	S3C1	3	1	ReLU
	S3C2	3	2	Softmax
Sub-network4	S4C1	3	1	ReLU
	S4C2	3	3	Softmax
Sub-network5	S5C1	3	2	ReLU
	S5C2	3	3	Softmax

*1 Every layer is the 1D dilated convolution layer with stride 1 and no paddings.

CPN includes five parallel sub-networks with two 1D dilated convolutional layers. Each sub-network is the equivalent of a fixed-length window sliding on videos. Its output is a set of probabilities that the video clip corresponding to the sliding window belongs to ME. The architecture of CPN is illustrated in Fig.4. The detailed configuration of the five sub-networks is in Table.II. The first layers of the first four sub-networks have the same configuration. So, they share the same weights to reduce the number of parameters further. The second layer has two neurons as the output denoted by $\hat{\mathbf{y}} = (\hat{y}_1, \hat{y}_2)^T \in \mathbb{R}^2$.

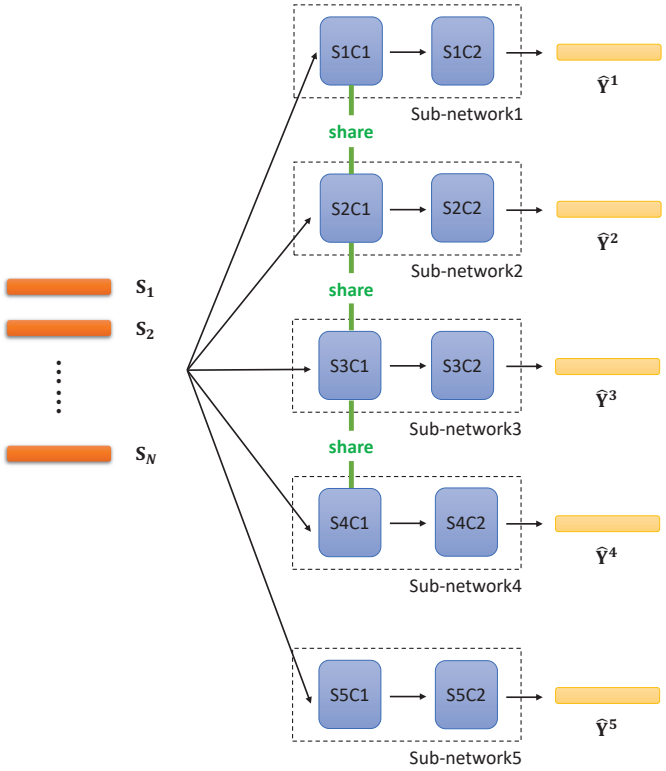


Fig. 4: Clip Proposal Network

The sth sub-network outputs a matrix $\hat{\mathbf{Y}}^s = (\hat{\mathbf{y}}^s_1, \hat{\mathbf{y}}^s_2, \dots \hat{\mathbf{y}}^s_{N_s}) \in \mathbb{R}^{2\times N_s}$, where $N_s = N - (k^s_1 - 1)r^s_1 - (k^s_2 - 1)r^s_2$ represents the sliding window number of the sth sub-network, where k^s_1, k^s_2 and r^s_1, r^s_2 are the kernel size and the dilation rate of the first, second layer of the sth sub-network.

In this paper, the length of a sliding window is expressed as the size of the receptive field of the sub-network's output on the spatial feature matrix $\mathbf{F}$ in Section III-A. Let's analyze the size of the receptive field of the third sub-network's output on $\mathbf{F}$.

Suppose v^x_i is one of outputs of the third sub-network. According to Eq.(9), $v^x_{i-1}, \dots, v^{x+r^3_2\times(k^3_2-1)}_{i-1}$ in the previous layer's outputs will influence the value v^x_i. According to Table.II, $k^3_2 = 3, r^3_2 = 2$. v^x_{i-1}, v^{x+2}_{i-1} and v^{x+4}_{i-1} influence v^x_i. Similarly $v^x_{i-2}, \dots, v^{x+r^3_1\times(k^3_1-1)}_{i-2}$ in the previous layer's outputs will influence the value v^x_{i-1}. v^x_{i-2}, v^{x+1}_{i-2} and v^{x+2}_{i-2} influence v^x_{i-1}. Similarly $v^{x+2}_{i-2}, v^{x+3}_{i-2}, v^{x+4}_{i-2}$ influence v^{x+2}_{i-1} and $v^{x+4}_{i-2}, v^{x+5}_{i-2}, v^{x+6}_{i-2}$ influence v^{x+4}_{i-1}. So, $v^x_{i-2}, v^{x+1}_{i-2}, \dots, v^{x+6}_{i-2}$ influence v^x_i. In other words, the size of receptive field of v^x_i on $(i-2)$th layer is 7. Further, we can get the size of receptive field of the third sub-network's output on $\mathbf{F}$ is 11. In the same way, we can deduce that sizes of receptive fields of five sub-network's outputs on $\mathbf{F}$ are 7, 9, 11, 13, 15 respectively.

Suppose the interval of the clip corresponding to $\hat{\mathbf{y}}^s_n$ is $[\hat{a}, \hat{b}]$ and the interval of the nearest true ME from the clip is $[a, b]$. The true label corresponding to $\hat{\mathbf{y}}^s_n$ is a one-hot vector and defined by

$$\mathbf{y}^s_n = \begin{cases} (1,0)^T \text{ means ME} & \text{if } \frac{|[\hat{a},\hat{b}]\cap[a,b]|}{|[\hat{a},\hat{b}]\cup[a,b]|} > 0.7 \\ (0,1)^T \text{ means non-ME} & \text{otherwise.} \end{cases} \tag{10}$$

where $|\cdot|$ denotes the number of elements in a set. The loss of CPN is computed by

$$L_{\text{CPN}} = -\frac{1}{\sum M(\hat{\mathbf{y}}^s_n)} \sum_{s=1}^{5}\sum_{n=1}^{N_s} M(\hat{\mathbf{y}}^s_n) \sum_{i=1}^{2} y^s_{in} \log \hat{y}^s_{in} + \frac{\lambda}{2}\sum_w w^2 \tag{11}$$

where the first item represents the cross entropy loss function, and the second item represents the L_2 regularization loss of all trainable parameters. $\lambda = 0.01$. $M(\hat{\mathbf{y}}^s_n) = 0$ or 1. It is introduced into the loss function in order to alleviate the sample imbalance problem of ME and non-ME. The sample number of ME is far less than that of non-ME. So, when $\mathbf{y}^s_n$ is ME, $M(\hat{\mathbf{y}}^s_n) = 1$. When $\mathbf{y}^s_n$ is non-ME, the probability that $M(\hat{\mathbf{y}}^s_n)$ is set as 1 is the ratio of ME to non-ME. CPN will propose a set of temporal segment proposals. The spatial features $\mathbf{f}_n$ corresponding to the proposals are fed into the last module, Classification Regression Network.

C. Classification Regression Network

Classification Regression Network (CRN) classifies the spatial features $\mathbf{f}_n$ corresponding to the proposals into ME or non-ME and further regresses the temporal boundaries of proposals belonging to ME.

Suppose the interval of the proposal is $[a', b']$. The corresponding spatial features $(\mathbf{f}_{a'}, \mathbf{f}_{a'+1}, \cdots, \mathbf{f}_{b'})$ are normalized into a fixed temporal length N_{CR}, and then are fed into CRN. The architecture of CRN is illustrated in Fig.5. It starts with two 1D convolution layers with the same configuration of kernel size 3, no paddings, stride 1, output channels 128, and ReLU activation function. 1D convolution layers are followed by two fully-connected layers with the same configuration of 300 neurons and the ReLU activation function. A dropout layer with 0.5 ratio follows fully-connected layers to reduce overfitting. Finally, two parallel fully-connected layers are appended. One has 2 neurons and a Softmax activation function. It outputs a vector $\mathbf{y}_{\text{cr}} = (y'_1, y'_2)^T$ to classify the proposed clip into "ME" or "non-ME". Another also has 2 neurons but without any activation function. It outputs a vector $\mathbf{r}_{\text{cr}} = (r'_1, r'_2)^T$ to regresses the temporal boundaries of the proposed clip. The loss function of CRN is defined as

$$L_{\text{CRN}} = \frac{1}{N_{\text{pro}}}\sum_{\mathcal{V}_{\text{pro}}}\left[-\sum_{i=1}^{2}\hat{y}'_i \log y'_i + S(\hat{\mathbf{y}}_{\text{cr}})E\left(r'_1 - \frac{a-a'}{b-a}\right) + S(\hat{\mathbf{y}}_{\text{cr}})E\left(r'_2 - \frac{b-b'}{b-a}\right)\right] + \frac{\lambda}{2}\sum_w w^2 \tag{12}$$

where

$$E(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{13}$$

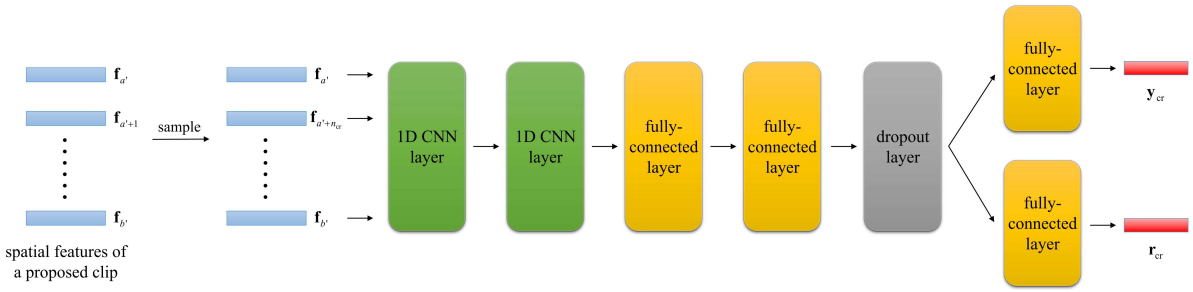


Fig. 5: Classification Regression Network

is the Smooth L_1 loss function [51]. The true label corresponding to $\hat{\mathbf{y}}_{\text{cr}}$ is a one-hot vector. It's definition is similar to Eq.(10). If the true label is "ME", the sign function $S(\hat{\mathbf{y}}_{\text{cr}})$ is set as 1. and suppose the true interval is $[a, b]$; if not, we let $S(\hat{\mathbf{y}}_{\text{cr}}) = 0$. Iterms $E\left(r'_1 - \frac{a-a'}{b-a}\right)$ and $E\left(r'_2 - \frac{b-b'}{b-a}\right)$ will be ignored.

In Eq.(12), the first item is the cross entropy loss function of classification and the Smooth L_1 loss of regression, and the second item is the L_2 regularization loss of trainable parameters in all layers except the last two fully-connected layers. $\lambda = 0.01$. $\mathcal{V}_{\text{pro}}$ represents every clip proposed by CPN. N_{pro} is the number of total proposed clips. If the prediction value $\mathbf{y}_{\text{cr}}$ reveals that the probability of ME is not less than a threshold T_{CRN}, then the proposed clip is output with the interval $[a' + r'_1(b-a), b' + r'_2(b-a)]$.

Specifically, there are two tricks in MESNet deal with ME spotting problem. The first is that we sample the spatial features instead of the corresponding spatiotemporal feature, in order to obtain a fixed-length input. The general way of computer vision is to downsample the backbone output features to get more shared calculations [51], [56]. However, for ME, the methods focus on improving spotting performance. Hence, we don't use the down-sampled spatiotemporal feature to approximate the spatiotemporal feature of the down-sampled frames. The second trick is that CRN doesn't share parameters of the two 1D CNN layers with the 2+1D Spatiotemporal Convolutional Network. The experiment result in subsection IV-C shows that this strategy can improve the performance of CRN, although it nearly doubles the network parameters.

D. Training Strategy, Post Process and Data Preparation

The total loss of MESNet is computed by

$$L_{\text{MESNet}} = L_{\text{CPN}} + L_{\text{CRN}} \tag{14}$$

All parameters are optimized by using mini-batch gradient descent with the batch size of 2 and the learning rate of 1×10^{-4}.

The direct output of MESNet is probabilities and regression values. Yet, in the prediction stage, we need to obtain ME intervals as the spotting result. Therefore, a simple post process is added to convert the output values to spotted intervals. A Non-Maximum Suppression (NMS) algorithm with threshold T_{nms} is then applied to remove some overlapping intervals as in [56], [64].

Suppose l_{MESNet} is the length of input video clips and n_{me} is the number of ME samples. In the training stage, numbers of clips labeled as ME, macro-expression and of clips without any label are $r_{tME} \cdot n_{\text{me}}$, $r_{tMaE} \cdot n_{\text{me}}$ and $r_{tNL} \cdot n_{\text{me}}$ respectively where r_{tME}, r_{tMaE} and r_{tNL} are ratios. Clips without any label are selected randomly for training.

In the prediction stage, the long video is split into several short clips to solve the memory limitation. The clips are with the fixed-length l_{MESNet}, and every two adjacent clips (except the final two clips) have an overlap with the length l_{overlap}. The overlap makes sure that every ME can be completely contained in a certain video clip when splitting the long video.

The l_{MESNet} and l_{overlap} are configured to different values according to different video FPS (frames per second). All prepared clips are then downsampled to 100 frames.

IV. Experiments

A. Datasets, Performance Metrics and Configuration

1) DataSets: CAS(ME)^2 [39] and SAMM [40] are used for performance validation of our proposed method. CAS(ME)^2 provides 98 long videos with 30 FPS. The average duration is 86 seconds. There are 22 subjects, with 57 MEs and 300 macro-expressions. SAMM provides 224 long videos with 200 FPS. The average duration is 35s. There are 32 subjects with 159 MEs. Both the macro-expressions and micro-expressions are labeled.

2) Performance metrics: The Second Facial Micro-Expressions Grand Challenge (MEGC2019) developed new metrics [41] for ME spotting result evaluation. However, this measure could not be applied in the case where multiple spotted intervals correspond to one same ground truth. For instance, as illustrated in Fig. 6, there are 2 MEs (ground truth) in the video and three spotted intervals. The first spotted interval is false positive, and the last two spotted intervals both have a large overlap with the second ME. According to the metrics of MEGC2019, the precision and the recall are 2/3 and 2/2. However, the precision should be 2/3 and the recall should be 1/2. Therefore, to adapt the ME interval spotting methods which may output intervals with overlap, we improve the metrics in MEGC2019 and redefine the performance metrics as follows.

For a spotted interval W_{spotted}, the prediction is considered as a true positive (TP) when there is a ground truth interval

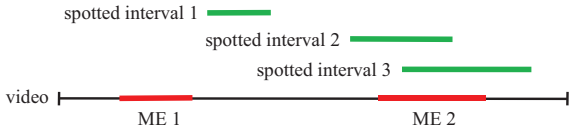


Fig. 6: Example of ME interval spotting method output. The spotted intervals may all have a large overlap with the same ground truth ME interval.

$W_{\text{groundTruth}}$ fitting the following condition:

$$\frac{W_{\text{spotted}} \cap W_{\text{groundTruth}}}{W_{\text{spotted}} \cup W_{\text{groundTruth}}} \geq T_{\text{eval}}, \tag{15}$$

where T_{eval} is the evaluation threshold. Suppose that: in terms of ground truth, there are m ground truth intervals in the dataset, and the spotting method finds a ground truth intervals, i.e., each of them has at least one spotted interval fulfilling the condition of Eq. 15). In terms of spotted intervals, there are n spotted intervals with b TPs. Then the recall, precision and F1-score are evaluated as follows:

$$Recall = \frac{a}{m}, \ Precision = \frac{b}{n}.$$

$$F1-score = \frac{2 \bullet Recall \bullet Precision}{Recall + Precision} = \frac{2ab}{an + bm}.$$

We use the F1-score to evaluate the ME spotting method. Other metrics such as the amount of TPs are also considered for a comprehensive analysis.

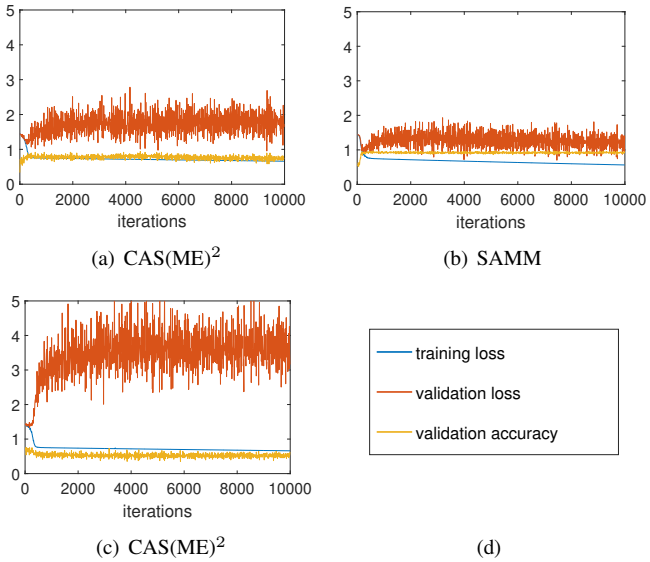
3) Experiment settings: Leave-one-subject-out (LOSO) cross-validation is utilized to evaluate the performance of MESNet. For each fold, all the videos in the dataset are divided into three sets: the training set, the validation set, and the testing set. The validation set is used to select a group of qualified 2+1D Spatiotemporal Convolutional Network's parameters to deal with the randomness of deep learning. The testing set consists of the videos that belong to the subject left. The videos of the other subjects are randomly split into the training set and the validation set based on the ratio of 7:3. Furthermore, the MEs in the training set are not less than 64% of the total MEs and the MEs in the validation set are not less than 15%. Thus, the numbers of MEs for training and for validation should be at least 37 and 9 in CAS(ME)^2, and at least 102 and 24 in SAMM. If this condition is not met, the videos in training and validation sets are randomly split again until the condition is met. In experiments, the input of MESNet is optical flow instead of original frames.

For 2+1D Spatiotemporal Convolutional Network's sample generation, we set $r_{noL} = 1.0$, $r_{MaE} = 0.5$ and $l_{noL} = 16$ for CAS(ME)^2, and set $r_{noL} = 1.0$ and $l_{noL} = 100$ for SAMM. For training of 2+1D Spatiotemporal Convolutional Network, we take iterative 10000 epochs on the training set, and the model is validated every 10 epochs. The 2+1D Spatiotemporal Convolutional Network is selected according to the validation results. If the overall results are bad, we randomly initialize the parameters and re-train the network until a relatively good one is found. The selected network's parameters are used to initiate the MESNet with the pre-trained parameters.

For MESNet sample generation, we set $r_{tME} = 10$, $r_{tMaE} = 4$, $r_{tNL} = 2$, $l_{\text{MESNet}} = 100$ and $L_{\text{overlap}} = 20$ for CAS(ME)^2, and set $r_{tME} = 10$, $r_{tNL} = 4$, $l_{\text{MESNet}} = 670$ and $l_{\text{overlap}} = 134$ for SAMM. The threshold T_{eval} is set to 0.5 by default if not specifically stated in this article. For MESNet training, the backbone parameters are initiated with the pre-trained ones from the selected 2+1D Spatiotemporal Convolutional Network's parameters. The training stage respectively takes iterative 30 epochs for CAS(ME)^2 and 85 epochs for SAMM. It is the early stop strategy to reduce overfitting.

B. results and analysis of 2+1D Spatiotemporal Convolutional Network

The LOSO protocol validates the proposed network. We need to train one 2+1D Spatiotemporal Convolutional Network model for each subject. In each dataset, the validation results of all models are similar. Thus, the performance analysis of 2+1D Spatiotemporal Convolutional Network is presented by only one model for each dataset in the paper. We select the models of leaving the 33rd subject in CAS(ME)^2 and leaving the 20th subject in SAMM, respectively. The results are shown in Fig. 7 and Table III.



(a) CAS(ME)^2 (b) SAMM

(c) CAS(ME)^2 (d)

Fig. 7: 2+1D Spatiotemporal Convolutional Network's result curves of training loss (mean batch), validation loss (mean batch) and validation accuracy: (a) leave-the 33th subject-out in CAS(ME)^2; (b) leave-the 20th subject-out in SAMM; (c) a bad parameter initialization of (a); (d) the legend.

In Fig. 7, with the increase of iterations, the training loss decreases. Meantime, the validation loss first drops and then rises. It is a typical overfitting phenomenon. It is reasonable because we inevitably face the small sample size problem of ME analysis, considering that 2+1D Spatiotemporal Convolutional Network has tens of thousands of parameters, but there are only dozens or hundreds of ME samples for training. Nevertheless, the validation accuracy maintains a relatively high level after increasing.

In CAS(ME)^2, the verification results are important for selecting a good parameter initiation. Fig.7(a) shows an example

of the good initiation models selected by constraining the verification results, marked as A. Fig.7(c) shows an example of the bad initiation models which use the same data, marked as B. The statistical analysis on A and B is listed in Table.III. It can be found that the overfitting of B is much more serious than the overfitting of A. The average accuracy of B (53.09%) is significantly lower than the average accuracy of A (77.31%). So when training 2+1D Spatiotemporal Convolutional Network in CAS(ME)^2, we randomly initiate the parameters again if the validation results are as bad as the performance of B. Generally, good models as A can be obtained in this way. Differently, SAMM has more ME samples for training, and most non-ME samples are neutral faces. Thus we can easily get good results, proven by comparing Fig.7(a) and Fig.7(b). As shown in Table.III, the average accuracy of the 10000 models can reach 91.71%.

TABLE III: The 2+1D Spatiotemporal Convolutional Network's validation results of leaving-the 33th subject-out in CAS(ME)^2 and leaving-the 20th subject-out in SAMM. (The best results are highlighted with the bold font.)

Models	CAS(ME)^2			SAMM	
	average A[*1]	average B	select[*2]	average	select
Total clips	48	48	48	88	88
ME clips	18	18	18	44	44
T[*3]	37	25	**44**	81	**86**
TP	10	4	**16**	35	**43**
Loss	1.72	3.46	**1.09**	1.28	**0.80**
Acurracy	77.31%	53.09%	**91.67%**	91.71%	**97.73%**

[*1] "average" means the average value of the total 10000 iterations. A represents the selected good parameter initialization models. B represents the bad initialization models.
[*2] "select" means the value of the selected good model.
[*3] T represents the number of all correct predictions, including TPs and true negatives (TNs).)

According to the validation results, we select a good model among the 10000 models. As shown in Table.III, in CAS(ME)^2, the average accuracy and TP of A are 77.31% and 10, but those of the selected model are 91.67% and 16. In SAMM, the average accuracy and TP of the models are 91.71% and 35, but those of the selected model are 97.73% and 43. The selected 2+1D Spatiotemporal Convolutional Network's parameters are used to initiate the MESNet backbone.

C. MESNet Results and Analysis

As the ME sample size in CAS(ME)^2 is small, the random division of the training and validation sets makes the results fluctuate. Hence, we perform the experiments three times with three different divisions. The result reported in this paper is the median, not the best or the worst. SAMM has more MEs. Thus the result difference caused by the random division and initiation is negligible.

1) Result details of CPN and CRN: Table.IV shows the ME spotting results of the two MESNet modules: CPN and CRN. The F1-score of the CRN is higher than the F1-score of the corresponding CPN, which proves that the two-stage prediction strategy effectively increases the F1-score by improving the precision. Compared with the first-stage (i.e. CPN) prediction, the second-stage (i.e. CRN) prediction produces fewer clips, fewer TPs, lower recall, higher precision, and higher F1-score. The different performances of CPN and CRN can meet different needs in real-life applications.

TABLE IV: Spotting results of CPN and CRN.

Dataset	CAS(ME)^2		SAMM	
MESNet module	CPN	CRN	CPN	CRN
MEs	57	57	159	159
Prediction[*1]	2799	893	3861	1336
TP	22	12	54	37
Find[*2]	18	10	52	36
Recall	0.32	0.18	0.33	0.23
Precision	0.008	0.013	0.014	0.028
F1-score	0.015	**0.026**	0.027	**0.049**

[*1] "Prediction" represents the number of total predicted clips.
[*2] "Find" represents the number of true MEs spotted by the algorithm.

2) The importance of 2+1D Spatiotemporal Convolutional Network: Table.V presents the comparison of the MESNet models with and without 2+1D Spatiotemporal Convolutional Network, using the same division of training and validation sets for each subject left out. The metrics of F1-score and the number of ground truth found are shown in the table. When MESNet isn't initiated with the pre-trained 2+1D Spatiotemporal Convolutional Network, the found ground truth intervals are much fewer and, the F1-score is much lower. The comparison reveals that 2+1D Spatiotemporal Convolutional Network is important for improving the performance of MESNet. It effectively transfers the knowledge in pre-trained parameters of 2+1D Spatiotemporal Convolutional Network into the MESNet model.

TABLE V: Result comparison of MESNet initiated with and without 2+1D Spatiotemporal Convolutional Network. (The better results are in bold.)

Dataset	CAS(ME)^2		SAMM	
with or not	without	with	without	with
CPN Find	4	**18**	15	**52**
CPN F1-score	0.013	**0.015**	0.014	**0.027**
CRN Find	0	**10**	4	**36**
CRN F1-score	0	**0.026**	0.018	**0.050**

3) The importance of not sharing temporal parameters: Both CPN and CRN need to extract temporal information from the spatial features of the same proposal clips. The effect of sharing the temporal parameters, i.e. setting separate parameters for 1D CNN layers in CRN from 1D CNN layers in 2+1D Spatiotemporal Convolutional Network is explored in this part. In the experiments, the two kinds of models use the same 2+1D Spatiotemporal Convolutional Network for initiation, and use the same data for training. The comparison results are listed in the Table.VI. The two architectures have similar performance of CPN. However, in terms of CRN, the architecture which do not share temporal parameters is better than the other one. The former CRN predicts more clips, which makes the F1-score much higher. It reveals the importance of not sharing temporal parameters, although it almost doubles the total parameters. The result is reasonable. For the same clip, the original spatial features fed to CPN and the down-sampled spatial features fed to CRN have different temporal patterns.

TABLE VI: Result comparison of two architectures: one is that CRN shares the temporal parameters with CPN, and another is that CRN has separate temporal parameters from CPN. (The better results are in bold.)

Dataset	CAS(ME)^2		SAMM	
share temporal parameters or not	share	not share	share	not share
CPN Prediction	2376	2799	3782	3848
CPN Find	16	18	50	52
CPN F1-score	**0.015**	**0.015**	0.025	**0.027**
CRN Prediction	772	871	2149	1318
CRN Find	6	10	35	36
CRN F1-score	0.014	**0.026**	0.034	**0.050**

4) Results with different threshold settings: We have done experiments to explore how the thresholds T_{CPN}, T_{CRN}, T_{nms} and T_{eval} influence the results.

First of all, we study the results with different CPN and CRN prediction thresholds, i.e. T_{CPN} and T_{CRN}. We set T_{CRN} to 0.40 and vary T_{CPN} from 0.46 to 0.80 with a step-size of 0.04, in order to observe the result changes caused by different T_{CPN}. Then, we set T_{CPN} to 0.50 and vary T_{CRN} from 0.40 to 0.80 with a step-size of 0.04, to observe the different results with T_{CRN} variation. The experiment results are shown in Table.VII. With the increase of a threshold, the number of total spotted clips and ground truth intervals found gradually decrease, and the F1-score first increases and then decreases. Different threshold settings cause the different performance of MESNet. The most true MEs are found when $T_{\text{CPN}} = 0.48$ and $T_{\text{CRN}} = 0.40$. CPN and CRN spot 21 and 13 MEs in CAS(ME)^2, and spot 65 and 42 MEs in SAMM. The highest F1-score of CPN and CRN are both 0.040 in CAS(ME)^2, and 0.083 and 0.088 in SAMM. A good model could be chosen by considering the recall and the F1-score jointly.

Secondly, the influence of NMS threshold T_{nms} on the spotted results is studied. We vary T_{nms} from 0.05 to 0.95 with a step-size of 0.10. The experiment results are shown in Table.VIII. In addition, the results of the setting without NMS and with $T_{\text{nms}} = 0.50$ are also listed in the table. The NMS process effectively reduces the number of spotted intervals without much degradation of F1-score and the number of true intervals found. For example, when there is no NMS, the CPN spots 28226 intervals in SAMM. With the decrease of T_{nms}, the number of spotted intervals gradually decreases. When $T_{\text{nms}} = 0.05$, the CPN only spots 2772 intervals. The number of spotted intervals is greatly reduced, 25454 less than the result without NMS. Meanwhile, the number of true intervals found is only decreased by 17, from 65 to 48, and the F1-score is only decreased by 0.022, from 0.055 to 0.033. The NMS process removes large amounts of overlapping intervals. When $T_{\text{nms}} = 0.5$, the number of TPs and the number of true intervals found are similar, which reveals the removal of overlapping intervals is sufficient. Hence, we set $T_{\text{nms}} = 0.5$ in our model to report.

Thirdly, we study the model performance with different evaluation metrics by varying the evaluation threshold T_{eval} from 0.05 to 0.95 with a step-size of 0.10. The results are shown in Table.IX, and the result with $T_{\text{eval}} = 0.5$ is also listed. With the decrease of T_{eval}, the metrics of TP, recall, and F1-score increase. Because the constraint on judging whether a prediction is correct gradually weakens as T_{eval} decreases. The metrics become the highest when T_{eval} is set to 0.05. CPN and CRN find 32 and 23 MEs in CAS(ME)^2, with the F1-scores of 0.053 and 0.107. In SAMM, CPN and CRN find 78 and 57 MEs, with the F1-scores of 0.049 and 0.090. The meaning of the weak constraint evaluation is to evaluate the algorithm's ability to locate the approximate moment when the subject lies, without the need of pointing out ME interval location precisely. We report the results with $T_{\text{eval}} = 0.5$ to evaluate the algorithm's ability of precisely spotting ME intervals.

5) The significant improvement compared with the state-of-the-art methods: Up to now, there are limited ME spotting methods. LBP-χ^2 [65], MDMD [29] and LTP-ML [38] are the state-of-the-art methods published. We compare our MESNet with these two methods. LBP-χ^2 and MDMD settings is the same as [43], and LTP-ML details follow [38]. All the intervals that are too long or too short are removed as [43]. All three methods use the same preprocessed images as those used by MESNet.

MESNet, LBP-χ^2, and MDMD have thresholds that can be adjusted. To compare different methods more comprehensively, we evaluate each method with two kinds of settings. One is marked as "General", which selects the threshold value to produce more true MEs found as well as a relatively high F1-score. Another is marked as "High F1-score", which selects the threshold value to produce a higher F1-score regardless of the few true MEs found. Concerning the "General" settings, in LBP-χ^2 and MDMD, the parameter p used to determine the threshold is set to 0.02 for CAS(ME)^2 and 0.01 for SAMM. In MESNet, all thresholds are set to 0.5. Regarding the "High F1-score" settings, p is set to 0.19 for CAS(ME)^2 and 0.01 for SAMM in MDMD; in LBP-χ^2, p is set to 0.31 for CAS(ME)^2 and 0.01 for SAMM. In MESNet, T_{CPN} and T_{CRN} are set to 0.56 and 0.40 for CAS(ME)^2, and set to 0.60 and 0.40 for SAMM.

The comparison results are shown in Table.X. Except for the "General" setting in CAS(ME)^2, both MESNet CPN and MESNet CRN have the highest F1-score and find the most MEs, no matter with the setting of "General" or "High F1-score", and no matter in CAS(ME)^2 or SAMM. CRN has a much higher F1-score, and CPN finds much more MEs. With the "General" setting in CAS(ME)^2, the performance of CPN and CRN is better than traditional methods when considering the F1-score and the true MEs found jointly, although the F1-score of CPN is not the highest and the true MEs found of CRN are not the most. In sum, our proposed MESNet outperforms the published state-of-the-art methods. Besides, the performance improvement is significant, especially in SAMM, because of more training samples.

V. Conclusion

To our knowledge, this paper first proposes the CNN-based method to spot multi-scale spontaneous ME intervals in long videos. The proposed MESNet contains two-stage predictions: one is the prediction of CPN; the other is the further prediction of CRN. We use the recently published spontaneous ME

TABLE VII: Result comparison with different CPN and CRN prediction thresholds (T_{CPN} and T_{CRN}). (The best results are in bold.)

	CAS(ME)^2									SAMM								
	vary T_{CPN}, $T_{\text{CRN}} = 0.40$						$T_{\text{CPN}} = 0.50$, vary T_{CRN}			vary T_{CPN}, $T_{\text{CRN}} = 0.40$						$T_{\text{CPN}} = 0.50$, vary T_{CRN}		
T[*1]	C-p[*2]	R-p	C-f	R-f	C-s	R-s	R-p	R-f	R-s	C-p	R-p	C-f	R-f	C-s	R-s	R-p	R-f	R-s
0.40	N/A	N/A	N/A	N/A	N/A	N/A	1433	**12**	0.019	N/A	N/A	N/A	N/A	N/A	N/A	1806	**38**	0.040
0.44	N/A	N/A	N/A	N/A	N/A	N/A	1221	11	0.020	N/A	N/A	N/A	N/A	N/A	N/A	1606	37	0.043
0.48	4034	2093	**21**	**13**	0.012	0.013	981	10	0.023	6254	2912	**65**	**42**	0.021	0.028	1408	37	0.048
0.52	1913	968	16	10	0.018	0.021	772	9	0.026	2376	1149	42	32	0.035	0.049	1222	36	0.053
0.56	935	438	12	8	0.028	**0.036**	596	6	0.024	1098	585	33	27	0.055	0.073	1073	32	0.053
0.60	444	195	5	3	0.023	0.024	459	6	0.027	659	364	30	23	0.077	**0.088**	927	29	0.055
0.64	230	107	4	1	0.033	0.012	324	5	0.031	438	263	24	16	**0.083**	0.076	792	26	0.056
0.68	123	61	3	1	**0.040**	0.017	233	5	**0.040**	301	187	13	9	0.057	0.052	675	25	0.060
0.72	67	34	2	0	0.032	0.000	155	3	0.028	218	129	11	7	0.058	0.049	564	23	**0.064**
0.76	36	19	1	1	0.022	0.026	90	2	0.027	163	103	9	6	0.056	0.046	442	17	0.057
0.80	19	11	1	0	0.026	0.000	42	2	**0.040**	124	78	7	4	0.049	0.034	335	12	0.049

[*1] "T" represents a certain threshold (T_{CPN} or T_{CRN}).

[*2] For the capital letter on the left of "-", "C" and "R" respectively represent CPN and CRN. For the lowercase letter on the right of "-", "p", "f" and "s" respectively represent "Prediction", "Find" and "F1-score".

TABLE VIII: Result comparison with different NMS thresholds (T_{nms}). (The best results are in bold.)

	CAS(ME)^2						SAMM					
	CPN			CRN			CPN			CRN		
T_{nms}	Prediction	TP / Find	F1-score	Prediction	TP / Find	F1-score	Prediction	TP / Find	F1-score	Prediction	TP / Find	F1-score
0.05	**1238**	13 / 13	0.020	**536**	8 / 8	0.027	**2772**	48 / 48	0.033	**1054**	30 / 30	0.049
0.15	2125	18 / 18	0.017	712	9 / 9	0.023	3003	50 / 50	0.032	1100	32 / 32	0.051
0.25	2270	18 / 18	0.015	737	9 / 9	0.023	3215	51 / 50	0.030	1135	32 / 32	0.049
0.35	2446	18 / 18	0.014	788	10 / 9	0.023	3458	52 / 51	0.029	1184	34 / 34	0.051
0.45	2703	19 / 18	0.014	839	12 / 10	0.026	3689	52 / 51	0.027	1260	35 / 34	0.049
(0.50)	(2799)	(22) / (18)	(0.015)	(871)	(12) / (10)	(0.026)	(3848)	(54) / (52)	(0.027)	(1318)	(37) / (36)	(0.050)
0.55	3034	23 / 19	0.015	924	13 / 11	0.026	4154	58 / 53	0.027	1396	40 / 37	0.051
0.65	3943	35 / 19	0.017	1113	18 / 12	0.030	5194	68 / 54	0.025	1705	54 / 39	0.056
0.75	5123	45 / 19	0.017	1412	23 / 12	0.030	8075	133 / 62	0.032	2335	81 / 40	0.061
0.85	7973	83 / 20	0.020	1851	31 / 13	0.031	17540	494 / 64	0.053	3961	173 / 43	0.075
0.95	10144	**113 / 20**	**0.022**	2316	**42 / 13**	**0.034**	27511	831 / 65	**0.056**	6232	322 / 44	0.087
(N/A)	(10144)	**(113) / 20**	**(0.022)**	(2335)	**(42) / (13)**	(0.033)	(28226)	**(838) / (66)**	(0.055)	(6792)	**(358) / (44)**	**(0.089)**

TABLE IX: Result comparison with different evaluation thresholds (T_{eval}). (The best results are in bold.)

	CAS(ME)^2				SAMM			
	CPN		CRN		CPN		CRN	
T_{eval}	TP / Find	F1-score	TP / Find	F1-score	TP / Find	F1-score	TP / Find	F1-score
0.05	**78 / 32**	**0.053**	**54 / 23**	**0.107**	**100 / 78**	**0.049**	**68 / 57**	**0.090**
0.15	27 / 21	0.019	17 / 15	0.036	89 / 74	0.044	60 / 52	0.080
0.25	25 / 21	0.017	15 / 13	0.032	78 / 69	0.039	54 / 47	0.072
0.35	24 / 20	0.017	15 / 13	0.032	72 / 67	0.036	51 / 45	0.068
0.45	23 / 19	0.016	13 / 11	0.028	59 / 55	0.029	42 / 41	0.057
(0.50)	(22) / (18)	(0.015)	(12) / (10)	(0.026)	(54) / (52)	(0.027)	(37) / (36)	(0.050)
0.55	21 / 18	0.015	9 / 9	0.019	45 / 45	0.022	34 / 34	0.046
0.65	14 / 13	0.010	7 / 7	0.015	27 / 27	0.013	24 / 24	0.033
0.75	7 / 7	0.005	1 / 1	0.002	10 / 10	0.005	13 / 13	0.018
0.85	5 / 5	0.004	0 / 0	0.000	1 / 1	0.001	3 / 3	0.004
0.95	0 / 0	0.000	0 / 0	0.000	0 / 0	0.000	2 / 2	0.003

long video datasets: CAS(ME)^2 and SAMM to evaluate the algorithm performance. Experiment results prove that the two-stage design can effectively enhance the F1-score metric. And the proposed MESNet outperforms the published state-of-the-art ME spotting methods regardless of the occurrence of over-fitting. Especially in SAMM, the performance improvement is very significant.

Moreover, we only use dozens or hundreds of MEs to train tens of thousands of parameters. It reveals the potential of the proposed method to achieve superior performance when there are more data available in the future. This work is an exploration of CNN-based ideas for ME spotting, lots of improvements are still needed. Further researches to explore better models is expected in future work.

REFERENCES

[1] E. A. Haggard and K. S. Isaacs, "Micromomentary facial expressions as indicators of ego mechanisms in psychotherapy," in *Methods of Research in Psychotherapy*, 1966, pp. 154–165.

[2] P. Ekman and W. V. Friesen, "Nonverbal leakage and clues to deception," *Psychiatry*, vol. 32, no. 1, pp. 88–106, 1969.

[3] P. Ekman, "Darwin, deception, and facial expression," *Annals of the New York Academy of Sciences*, vol. 1000, no. 1, pp. 205–221, 2003.

[4] ——, "Lie catching and microexpressions," *The Philosophy of Deception*, pp. 118–133, 2009.

TABLE X: Result comparison with the state-of-the-art methods. (The highest F1-scores are in bold. The second highest F1-scores are underlined.)

Dataset	Model selection	Method	MEs	Prediction	TP / Find	Recall	Precision	F1-score
CAS(ME)^2	General	MESNet-CPN (proposed)	57	2799	22 / 18	0.32	0.008	0.015
		MESNet-CRN (proposed)	57	893	12 / 10	0.18	0.013	**0.026**
		MDMD [29]	57	3814	18 / 18	0.32	0.005	0.009
		LBP-χ^2 [65]	57	566	6 / 6	0.11	0.011	<u>0.019</u>
		LTP-ML [44]	57	3136	4 / 4	0.05	0.001	0.002
	High F1-score	MESNet-CPN (proposed)	57	935	14 / 12	0.21	0.024	<u>0.028</u>
		MESNet-CRN (proposed)	57	438	9 / 8	0.14	0.032	**0.036**
		MDMD [29]	57	307	5 / 5	0.09	0.016	0.027
		LBP-χ^2 [65]	57	185	3 / 3	0.05	0.016	0.025
		LTP-ML [44]	57	3136	4 / 4	0.05	0.001	0.002
SAMM	General	MESNet-CPN (proposed)	159	3861	54 / 52	0.33	0.014	<u>0.027</u>
		MESNet-CRN (proposed)	159	1336	37 / 36	0.23	0.028	**0.049**
		MDMD [29]	159	4990	31 / 31	0.19	0.006	0.012
		LBP-χ^2 [65]	159	768	8 / 8	0.05	0.010	0.017
		LTP-ML [44]	159	1199	11 / 11	0.07	0.009	0.016
	High F1-score	MESNet-CPN (proposed)	159	659	32 / 30	0.19	0.049	<u>0.077</u>
		MESNet-CRN (proposed)	159	364	23 / 23	0.14	0.063	**0.088**
		MDMD [29]	159	4990	31 / 31	0.19	0.006	0.012
		LBP-χ^2 [65]	159	768	8 / 8	0.05	0.010	0.017
		LTP-ML [44]	159	1199	11 / 11	0.07	0.009	0.016

[5] J. Endres and A. Laidlaw, "Micro-expression recognition training in medical students: a pilot study," *BMC Medical Education*, vol. 9, no. 1, p. 47, 2009.

[6] M. Frank, D. Kim, S. Kang, A. Kurylo, and D. Matsumoto, "Improving the ability to detect micro expressions in law enforcement officers," *Manuscript in preparation*, 2014.

[7] P. A. Stewart, B. M. Waller, and J. N. Schubert, "Presidential speechmaking style: Emotional response to micro-expressions of facial affect," *Motivation and Emotion*, vol. 33, no. 2, p. 125, 2009.

[8] M. OSullivan, M. G. Frank, C. M. Hurley, and J. Tiwana, "Police lie detection accuracy: The effect of lie scenario," *Law and Human Behavior*, vol. 33, no. 6, pp. 530–538, 2009.

[9] P. Ekman, M. O'Sullivan, and M. G. Frank, "A few can catch a liar," *Psychological Science*, vol. 10, no. 3, pp. 263–266, 1999.

[10] P. Ekman, "MicroExpression Training Tool (METT)," *San Francisco: University of California*, 2002.

[11] M. Frank, M. Herbasz, K. Sinuk, A. Keller, and C. Nolan, "I see how you feel: Training laypeople and professionals to recognize fleeting emotions," in *International Communication Association*, 2009.

[12] Q. Wu, X. Shen, and X. Fu, "The machine knows what you are hiding: an automatic micro-expression recognition system," in *international conference on affective computing and intelligent Interaction*. Springer, 2011, pp. 152–162.

[13] S. Polikovsky, Y. Kameda, and Y. Ohta, "Facial micro-expressions recognition using high speed camera and 3D-gradient descriptor," in *International Conference on Crime Detection and Prevention*, 2010.

[14] M. Shreve, S. Godavarthy, V. Manohar, D. Goldgof, and S. Sarkar, "Towards macro-and micro-expression spotting in video using strain patterns," in *Workshop on Applications of Computer Vision*, 2009, pp. 1–6.

[15] Y.-J. Liu, J.-K. Zhang, W.-J. Yan, S.-J. Wang, G. Zhao, and X. Fu, "A main directional mean optical flow feature for spontaneous micro-expression recognition," *IEEE Transactions on Affective Computing*, vol. 7, no. 4, pp. 299–310, 2015.

[16] S.-J. Wang, H.-L. Chen, W.-J. Yan, Y.-H. Chen, and X. Fu, "Face recognition and micro-expression recognition based on discriminant tensor subspace analysis plus extreme learning machine," *Neural processing letters*, vol. 39, no. 1, pp. 25–43, 2014.

[17] S.-J. Wang, W.-J. Yan, G. Zhao, X. Fu, and C.-G. Zhou, "Micro-expression recognition using robust principal component analysis and local spatiotemporal directional features," in *European Conference on Computer Vision*. Springer, 2014, pp. 325–338.

[18] T. Pfister, X. Li, G. Zhao, and M. Pietikäinen, "Recognising spontaneous facial micro-expressions," in *IEEE International Conference on Computer Vision*, 2011, pp. 1449–1456.

[19] X. Huang, S.-J. Wang, G. Zhao, and M. Pietikäinen, "Facial micro-expression recognition using spatiotemporal local binary pattern with integral projection," in *IEEE International Conference on Computer Vision Workshops*, 2015, pp. 1–9.

[20] Y. Wang, J. See, R. C.-W. Phan, and Y.-H. Oh, "Efficient spatio-temporal local binary patterns for spontaneous facial micro-expression recognition," *Plos One*, vol. 10, no. 5, p. e0124674, 2015.

[21] S.-T. Liong, J. See, K. Wong, and R. C.-W. Phan, "Automatic micro-expression recognition from long video using a single spotted apex," in *Asian Conference on Computer Vision*. Springer, 2016, p. 345360.

[22] ——, "Less is more: Micro-expression recognition from video using apex frame," *Signal Processing: Image Communication*, vol. 62, pp. 82–92, 2018.

[23] Y. Li, X. Huang, and G. Zhao, "Can micro-expression be recognized based on single apex frame?" in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 3094–3098.

[24] Z. Zhang, T. Chen, H. Meng, G. Liu, and X. Fu, "SMEConvNet: A convolutional neural network for spotting spontaneous facial micro-expression from long videos," *IEEE Access*, vol. 6, pp. 71 143–71 151, 2018.

[25] X. Li, X. Hong, A. Moilanen, X. Huang, T. Pfister, G. Zhao, and M. Pietikäinen, "Towards reading hidden emotions: A comparative study of spontaneous micro-expression spotting and recognition methods," *IEEE Transactions on Affective Computing*, vol. 9, no. 4, pp. 563–577, 2018.

[26] A. Davison, W. Merghani, C. Lansley, C.-C. Ng, and M. H. Yap, "Objective micro-facial movement detection using facs-based regions and baseline evaluation," in *Automatic Face & Gesture Recognition (FG 2018), 2018 13th IEEE International Conference on*. IEEE, 2018, pp. 642–649.

[27] D. Patel, G. Zhao, and M. Pietikäinen, "Spatiotemporal integration of optical flow vectors for micro-expression detection," in *International Conference on Advanced Concepts for Intelligent Vision Systems*, 2015, pp. 369–380.

[28] X. Li, J. Yu, and S. Zhan, "Spontaneous facial micro-expression detection based on deep learning," in *International Conference on Signal Processing*, 2016, pp. 1130–1134.

[29] S.-J. Wang, S. Wu, X. Qian, J. Li, and X. Fu, "A main directional maximal difference analysis for spotting facial movements from long-term videos," *Neurocomputing*, vol. 230, pp. 382–389, 2017.

[30] Y. Han, B. Li, Y.-K. Lai, and Y.-J. Liu, "Cfd: A collaborative feature difference method for spontaneous micro-expression spotting," in *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2018, pp. 1942–1946.

[31] L.-W. Zhang, J. Li, S.-J. Wang, X.-H. Duan, W.-J. Yan, H.-Y. Xie, and S.-C. Huang, "Spatio-temporal fusion for macro-and micro-expression spotting in long video sequences," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*. IEEE Computer Society, 2020, pp. 245–252.

[32] C. A. Duque, O. Alata, R. Emonet, A. C. Legrand, and H. Konik, "Micro-expression spotting using the riesz pyramid," in *IEEE Winter Conference on Applications of Computer Vision*, 2018, pp. 66–74.

[33] H. Lu, K. Kpalma, and J. Ronsin, "Micro-expression detection using integral projections," 2017.

[34] Z. Xia, X. Feng, J. Peng, X. Peng, and G. Zhao, "Spontaneous micro-expression spotting via geometric deformation modeling," *Computer Vision and Image Understanding*, vol. 147, p. 8794, 2016.

[35] T.-K. Tran, X. Hong, and G. Zhao, "Sliding window based micro-expression spotting: A benchmark," in *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, 2017, pp. 542–553.

[36] D. Borza, R. Danescu, R. Itu, and A. Darabant, "High-speed video system for micro-expression detection and recognition," *Sensors*, vol. 17, no. 12, p. 2913, 2017.

[37] P. Husák, J. Čech, and J. Matas, "Spotting facial micro-expressions" in the wild," in *22nd Computer Vision Winter Workshop*, 2017.

[38] J. Li, C. Soladié, and R. Séguier, "Local temporal pattern and data augmentation for micro-expression spotting," *IEEE Transactions on Affective Computing*, 2020.

[39] F. Qu, S. J. Wang, W. J. Yan, H. Li, S. Wu, and X. Fu, "CAS(ME)^2: A database for spontaneous macro-expression and micro-expression spotting and recognition," *IEEE Transactions on Affective Computing*, vol. 9, no. 4, pp. 424–436, 2017.

[40] C. Yap, C. Kendrick, and M. Yap, "Samm long videos: A spontaneous facial micro-and macro-expressions dataset," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, 2020, pp. 194–199.

[41] J. See, M. H. Yap, J. Li, X. Hong, and S.-J. Wang, "MEGC 2019–the second facial micro-expressions grand challenge," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*. IEEE, 2019, pp. 1–5.

[42] J. Li, S.-J. Wang, M. H. Yap, J. See, X. Hong, and X. Li, "MEGC2020-the third facial micro-expression grand challenge," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, 2020, pp. 234–237.

[43] Y. He, S.-J. Wang, J. Li, and M. H. Yap, "Spotting macro-and micro-expression intervals in long video sequences," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*. IEEE, 2020, pp. 742–748.

[44] J. Li, C. Soladié, R. Séguier, S.-J. Wang, and M. H. Yap, "Spotting micro-expressions on long videos sequences," in *IEEE International Conference on Automatic Face and Gesture Recognition*, 2019, pp. 1–5.

[45] M. Verburg and V. Menkovski, "Micro-expression detection in long videos using optical flow and recurrent neural networks," in *2019 14th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*. IEEE, 2019, pp. 1–6.

[46] H. Pan, L. Xie, and Z. Wang, "Local bilinear convolutional neural network for spotting macro-and micro-expression intervals in long video sequences," in *2020 15th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2020)*, 2020, pp. 343–347.

[47] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[48] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[49] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International journal of computer vision*, vol. 104, no. 2, pp. 154–171, 2013.

[50] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[51] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2017.

[52] R. Zeng, W. Huang, M. Tan, Y. Rong, P. Zhao, J. Huang, and C. Gan, "Graph convolutional networks for temporal action localization," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7094–7103.

[53] R. Zeng, C. Gan, P. Chen, W. Huang, Q. Wu, and M. Tan, "Breaking winner-takes-all: Iterative-winners-out networks for weakly supervised temporal action localization," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 5797–5808, 2019.

[54] Z. Shou, J. Chan, A. Zareian, K. Miyazawa, and S.-F. Chang, "Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5734–5743.

[55] H. Xu, A. Das, and K. Saenko, "R-C3D: Region convolutional 3D network for temporal activity detection," in *IEEE International Conference on Computer Vision*, 2017, pp. 5783–5792.

[56] Y. W. Chao, S. Vijayanarasimhan, B. Seybold, D. A. Ross, J. Deng, and R. Sukthankar, "Rethinking the Faster R-CNN architecture for temporal action localization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1130–1139.

[57] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles, "End-to-end, single-stream temporal action detection in untrimmed videos," 2019.

[58] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2678–2687.

[59] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[60] S.-J. Wang, B.-J. Li, Y.-J. Liu, W.-J. Yan, X. Ou, X. Huang, F. Xu, and X. Fu, "Micro-expression recognition with small sample size by transferring long-term convolutional neural network," *Neurocomputing*, vol. 312, pp. 251–262, 2018.

[61] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *International Conference on Artificial Intelligence and Statistics*, vol. 15, 2011, pp. 315–323.

[62] K. J. Lang and G. E. Hinton, "Dimensionality reduction and prior knowledge in e-set recognition," in *Advances in Neural Information Processing Systems*, 1990, pp. 178–185.

[63] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," in *International Conference on Learning Representations (ICLR)*, 2016.

[64] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6299–6308.

[65] A. Moilanen, G. Zhao, and M. Pietikäinen, "Spotting rapid facial movements from videos using appearance-based feature difference analysis," in *International Conference on Pattern Recognition*, 2014, pp. 1722–1727.

Su-Jing Wang (M'12-SM'19) is an Associate Researcher, master supervisor at the Institute of Psychology, Chinese Academy of Sciences. He received the Ph.D degree from the College of Computer Science and Technology of Jilin University in 2012. He was a postdoctoral researcher at the Institute of Psychology, Chinese Academy of Sciences from 2012 to 2015. Since July 2015, he has joined the Chinese Academy of Sciences. His current research interests include pattern recognition and machine learning, especially the micro-expression analysis. He has published more than 50 scientific papers in several important national and international journals and conferences, including TIP, TNN, and ECCV etc. Since 2014, he has served as an associate editor of Neurocomputing (Elsevier). He is also a CCF Distinguished Member, an IEEE Senior Member, a technical committee member of the CCF-Computer Vision, a technical committee member of the Artificial Intelligence and Artificial Emotion of the Chinese Association for Artificial Intelligence (CAAI), and a technical committee member of computer vision of the China Society of Image and Graphics (CSIG). He presided over 2 projects of the National Natural Science Foundation of China, 1 project of the Beijing Natural Science Foundation, and 2 Chinese postdoctoral funds. He won the first prize of the 8th Wu Wenjun Artificial Intelligence Science and Technology Award in 2018. He was called as Chinese Hawking by the Xinhua News Agency.

Ying He received the Master's degree in computer applied technology from the College of Information Engineering, North China University of Science and Technology, China, in 2019. She majored in computer science and minored in e-commerce before pursuing the Master's degree, from 2011 to 2016. During the period of the master, she has experiences of researches on computational mathematics, in College of Science, North China University of Science and Technology, and researches on object detection based on deep learning, in Institute of Psychology, Chinese Academy of Sciences, from 2016 to 2019. She has been researching on micro-expression spotting based on deep learning, in Institute of Psychology, Chinese Academy of Sciences, from 2019 to now. Her research interests include machine learning, computer vision and approximation theory.

Jingting Li (M'20) is currently a postdoc at the Institute of Psychology, Chinese Academy of Sciences. She received her master degree from Beihang university in 2016 in major of Electronic and Communication Engineering. She was a PhD student in FAST (Facial Analysis, Synthesis and Tracking) research team of CentraleSuplec and she received the PhD degree in Signal, Image, Vision in 2019. Her current research interests include image processing, computer vision and pattern recognition, especially facial micro-expression analysis.

Xiaolan Fu (M'13) received her Ph. D. degree in 1990 from Institute of Psychology, Chinese Academy of Sciences. Currently, she is a Senior Researcher at Cognitive Psychology. Her research interests include visual and computational cognition: (1) attention and perception, (2) learning and memory, and (3) affective computing. At present, she is the director of Institute of Psychology, Chinese Academy of Sciences and the director of department of psychology, University of the Chinese Academy of Sciences.