Breaking Winner-Takes-All: Iterative-Winners-Out Networks for Weakly Supervised Temporal Action Localization

Runhao Zeng, Chuang Gan, Peihao Chen, Wenbing Huang, Qingyao Wu, and Mingkui Tan

***Abstract*— We address the challenging problem of weakly supervised temporal action localization from unconstrained web videos, where only the video-level action labels are available during training. Inspired by the adversarial erasing strategy in weakly supervised semantic segmentation, we propose a novel iterative-winners-out network. Specifically, we make two technical contributions: we propose an iterative training strategy, namely, winners-out, to select the most discriminative action instances in each training iteration and remove them in the next training iteration. This iterative process alleviates the "winner-takes-all" phenomenon that existing approaches tend to choose the video segments that strongly correspond to the video label but neglects other less discriminative video segments. With this strategy, our network is able to localize not only the most discriminative instances but also the less discriminative ones. To better select the target action instances in winners-out, we devise a class-discriminative localization technique. By employing the attention mechanism and the information learned from data, our technique is able to identify the most discriminative action instances effectively. The two key components are integrated into an end-to-end network to localize actions without using the frame-level annotations. Extensive experimental results demonstrate that our method outperforms the state-of-the-art weakly supervised approaches on ActivityNet1.3 and improves mAP from 16.9% to 20.5% on THUMOS14. Notably, even with weak video-level supervision, our method attains comparable accuracy to those employing frame-level supervisions.**

***Index Terms*— Weakly supervised learning, action localization, winners-out, untrimmed video.**

I. Introduction

ACTION localization has attracted more and more interest in recent years, owing to its numerous potential applications in video retrieval, video surveillance and other areas. Given a video, this task aims to solve two problems simultaneously: (1) identifying the start time and end time of each action instance in the video and (2) recognizing the category of each action instance. Performing action localization is central to video understanding especially for untrimmed videos, as they usually contain both action and background instances and we are interested in the actions only.

Early works mainly focus on classifying proposals generated by sliding windows with hand-crafted features [1], [2]. More recently, great progress has been achieved in action localization [3]–[7] with the help of deep learning based image or video analysis methods [8]–[18]. However, all these methods heavily rely on the temporal annotations: the category label of each action item and its time interval. Clearly, manually annotating actions frame by frame is not only time-consuming but also subjective to the annotators, leading the annotations to be severely biased. In this paper, we attempt to perform action localization under the weakly supervised condition, where only the video-level action labels are provided, but the action time intervals and their exact labels are no longer required.

Weakly supervised learning has been studied in the domain of still images. Taking the task of semantic segmentation for example, it aims to predict the category of each pixel in a given image. Since collecting the pixel-level annotations is extremely costly, researchers seek to perform semantic segmentation by using only the image-level class labels. To this end, the Adversarial Erasing approach proposed by [19] has achieved state-of-the-art results. Specifically, in this method, the regions (or pixels) belonging to the target class of the given image are iteratively selected to obtain the segmentation result and the selected regions (or pixels) will be removed from the image in the next training step.

Given its success, we are interested in employing such a technique for our weakly supervised action localization problem in videos. However, the large domain shift between images and videos will incur the following challenges:

A. Videos Are Much More Complex Than Images

In [19], to train the classification network, the authors obtain a feature vector for each input image by adding a global mean pooling layer upon the last convolution layer. As shown in Fig. 2 (a) and (c), the mean pooling is performed over pixels spatially. Thus, the dimension of the feature vector equals the number of channels in the last convolutional layer. To apply this idea to action localization, one can treat the segments (or frames) in videos as the pixels in images, and derive the

Manuscript received August 1, 2018; revised April 23, 2019; accepted May 23, 2019. Date of publication June 17, 2019; date of current version August 30, 2019. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61602185, Grant 61836003, and Grant 61876208, in part by the Program for Guangdong Introducing Innovative and Enterpreneurial Teams under Grant 2017ZT07X183, in part by the Guangdong Provincial Scientific and Technological Funds under Grant 2018B010107001, Grant 2017B090901008, and Grant 2018B010108002, in part by the Pearl River S&T Nova Program of Guangzhou under Grant 201806010081, and in part by the CCF-Tencent Open Research Fund under Grant RAGR20190103. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Lucio Marcenaro. *(Corresponding author: Mingkui Tan.)*

R. Zeng, P. Chen, Q. Wu, and M. Tan are with the School of Software Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: runhaozeng.cs@gmail.com; phchencs@gmail.com; qyw@gmail.com; mingkuitan@gmail.com).

C. Gan is with the MIT-IBM Watson AI Lab, Cambridge, MA 02142 USA (e-mail: ganchuang1990@gmail.com).

W. Huang is with Tencent AI Laboratory, Shenzhen 518000, China (e-mail: hwenbing@126.com).

Digital Object Identifier 10.1109/TIP.2019.2922108

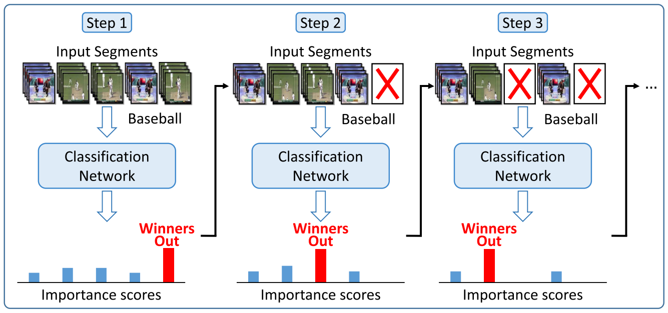


Fig. 1. Illustration of the proposed winners-out strategy. For an input video, we divide it into multiple segments evenly. Note that there often contain some segments which may strongly correspond to the video label, preventing us from localizing all the related actions (*i.e.*, the winner-takes-all issue). To break this issue, we propose to iteratively localize the most discriminative segments and remove them from the training segment set. Specifically, at each step, we select segments with the largest importance scores attained from a classification network w.r.t the video label (*e.g. Baseball*). The selected segments (see the red bars) will be removed from the video. Thus, these "winners" segments are "out" in the next training step. Then, the classification network is re-trained to localize other discriminative segments. With this iterative process, the network learns to localize all the action instances in one video, including the most discriminative and the less discriminative ones.

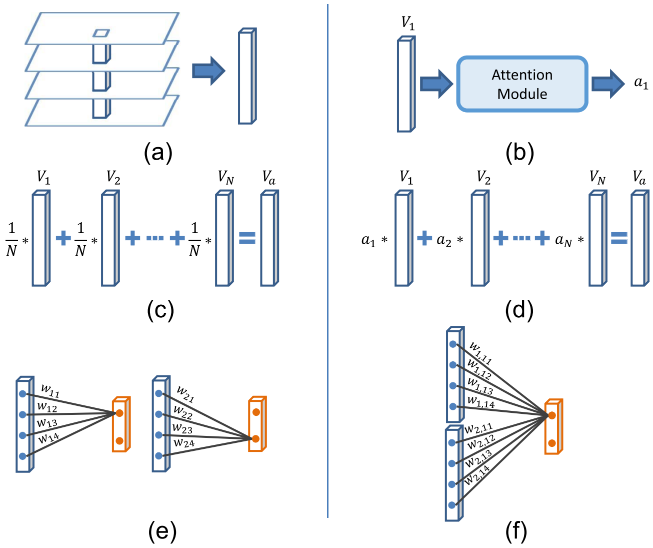

(a)
V_1
Attention Module
a_1
(b)
V_1 V_2 V_N V_a
$\frac{1}{N}*$ + $\frac{1}{N}*$ + ··· + $\frac{1}{N}*$ =
(c)
V_1 V_2 V_N V_a
a_1* + a_2* + ··· + a_N* =
(d)
w_{11} w_{12} w_{13} w_{14}
w_{21} w_{22} w_{23} w_{24}
(e)
$w_{1,11}$ $w_{1,12}$ $w_{1,13}$ $w_{1,14}$
$w_{2,11}$ $w_{2,12}$ $w_{2,13}$ $w_{2,14}$
(f)

Fig. 2. Toy models of our proposed method and [19]. (a) In [19], since the layer before the final fc layer outputs multiple feature maps, the feature of each spatial location can be viewed as a vector. (b) Our attention module assigns each feature vector with a weight. (c) and (d) illustrate the different processes of calculating global feature. Here, V_i is the pixel/segment level feature vector and V_a is the aggregated image/video level feature. [19] applies mean pooling over the feature vectors, while we apply weighted sum using the attention weights. (e) [19] finally outputs one global feature vector. (f) We concatenate multiple global features to obtain the final feature. The vector in blue is the final global feature and the orange one is the classification probability. Here, the number of classes is set to two for simplicity.

video-level feature by performing mean pooling over segment features. However, as videos are far more complex than images, mean pooling is unable to produce good features for classifying videos. Because mean pooling treats each segment equally, and it fails to select the discriminative segments for representing the video. We explore the attention mechanism to address this issue, given the success of attention mechanism in natural language processing [20] and computer vision [21]. Particularly, we first assign each segment-level feature with an attention weight and then perform weighted sum to obtain the final global feature accordingly (see Fig. 2 (b) and (d)). With the enhanced features, we train a classification network using the winners-out strategy, which will be introduced in § III-B.

B. The Criteria for Selecting the Segments (or Frames) to Remove Requires to Be Developed Specifically if the Attention Mechanism Is Considered

To select which segment to remove, we should first analyze the importance of each segment. In [19], the authors employ CAM [22] to analyze the correlation between each pixel and the target class. In CAM, the correlation of each pixel only depends on the activation values (*i.e.,* the feature vector) and the weights in the last fully connected layer. Directly using CAM in our method will neglect the attention weights, making the correlation value not precise enough. To address this issue, we propose a class-discriminative localization technique, namely **Class-specific Score Computing**. In this technique, the correlation for each segment is related to its attention weight. Besides the attention weights, we also use the segment feature and the weights that connect the video-level feature and the action class in the final fc layer. Therefore, this technique makes good use of category information learned from the training stage and provides a better reference for selecting the most discriminative action instances. Furthermore, we empirically find that employing multiple attention modules will boost the performance. The final output of our model is obtained by concatenating the outputs of all attention modules (see Fig. 2 (f)). In this way, our proposed technique is able to determine the removed segments by considering attention weights from multiple attention modules, leading to a more robust decision.

The main contributions of our paper are as follows:

(1) To the best of our knowledge, it is the first attempt to employ the adversarial erasing mechanism in video analysis. To narrow the gap between image segmentation and video action localization, we introduce an iterative-winners-out network with a training strategy, winners-out. Given the video-level labels only, we successfully train the network to localize actions in untrimmed videos.

(2) We propose Class-specific Score Computing (CSC), a class-discriminative localization technique that can analyze the importance of each action instance in the video. Compared to CAM, CSC is able to localize actions in a more precise way.

(3) We integrate winners-out and CSC into an end-to-end network and yield the state-of-the-art performance on two datasets, *i.e.* THUMOS14 [23] and ActivityNet1.3 [24]. We significantly improve the current best results from 16.9% to 20.5% on THUMOS14.

The remainder of this paper is organized as follows. In Section II, we review recent works related to our paper, which provides background knowledge to understand the motivation of designing our iterative-winners-out network. In Section III, we present details of the proposed winners-out training strategy and the Class-specific Score

Computing (CSC) technique. Also, we discuss how our proposed iterative-winners-out network is related to existing methods. The training and testing details are illustrated in Section IV. Experiments and ablation studies are presented in Section V and Section VI, respectively, before the conclusion is drawn in Section VII.

II. Related Work

A. Action Localization

Early works on this task employ hand-crafted features and mainly focus on classifying proposals generated by sliding windows [1], [2]. In recent years, deep learning based methods have been extensively studied, and they can be grouped into four categories. In the first category, methods perform frame or segment-level classification, which needs merging steps to obtain the temporal boundaries [5], [25], [26]. For instance, Shou *et al.* [5] proposed to predict the category of each frame using a convolution-deconvolution network and merge the results using a greedy Gaussian-based strategy. Another category of approaches employs a multiple-stage framework involving proposal generation, classification and boundary refinement [3], [4], [6]. Typically, Shou *et al.* [3] proposed a multi-stage approach involving three segment-based 3D ConvNets, one for generating proposals and the other two for classifying proposals. Apart from the methods that neglect the context information of the proposals, Dai *et al.* [27] constructed features using the context around proposals. Zhao *et al.* [6] introduced the completeness of proposals and designed a pyramid structure to classify the proposals. In the third category, the methods devise end-to-end architectures integrating the proposal generation and classification [28]. Another class of approaches apply a recurrent neural network to learn temporal features for localizing actions [7]. Though these methods have achieved great results, they heavily rely on temporal annotations to train the neural networks. Our method is distinct from these approaches, for we only need video labels rather than temporal annotations for training.

B. Weakly Supervised Action Localization

There are only a few approaches that tackle the action localization problem with weak supervision. The work that most related to ours are [29]–[31]. UntrimmedNet [29] uses a network with an attention module for untrimmed video classification and performs action localization using the attention weights (*i.e.,* action instances have large attention scores and background instances have small scores). However, this method has two main shortcomings. 1) Due to relying on only a classification objective, it often tends to choose the action instances that strongly correspond to the video label, and thus neglects other less discriminative ones. Specifically, only the most discriminative instance is assigned with a high attention score and other instances are with small scores. This is a severe issue since only the most discriminative action instances will be localized, which can be regarded as a "winner-takes-all" phenomenon. In the meantime, using the softmax function will further suppress the scores of the less discriminative instances. 2) Only one attention module is trained, while it is employed to assign scores for action instances across all action classes. Since this attention module is class-agnostic, it is difficult to capture the action patterns for all classes. Directly using attention scores to localize action instances for all classes is both imprecise and inefficient. To address the first issue of [29], Nguyen *et al.* [31] replaced the soft-max function with the sigmoid function, which predicts the attention weight of each segment independently. HaS [30] proposed to hide each segment with a fixed probability before feeding it into an action localizer network to avoid the only concentration of the most discriminative segment. But this mechanism does not work well in videos. The reason may be that the most discriminative segments cannot be hidden efficiently due to the random probability, preventing the network from learning the less discriminative features as much as possible. Instead, we propose a training strategy to hide segments in the video in a goal-oriented way.

C. Weakly Supervised Object Detection

Our work is related to weakly supervised object detection [22], [32]–[36]. Bilen and Vedaldi *et al.* [32] solved the weakly supervised object detection problem using an objective for classification. Zhou *et al.* [22] proposed a technique called CAM to perform object detection in an interesting way. In light of the above methods, we tackle the weakly supervised action localization problem by training a video classification network and propose a class-discriminative localization technique in video, which is related to CAM [22] and [37] in spirit.

III. Our Approach

A. Problem Definition

We denote an untrimmed video set as $\{V_i, \boldsymbol{y}_i\}_{i=1}^{N}$, where N is the number of videos. Let $\boldsymbol{y}_i \in \{0, 1\}^C$ be the label vector for the i-th video with C being the total number of action classes in the video set. Here, each video may belong to one class or multiple classes depending on how many types of actions are included in the video. Each untrimmed video V_i may contain a set of action instances $\Phi_{V_i} = \{\phi_{q_i} = (\varphi_{q_i}, \varphi'_{q_i}, k_{q_i})\}_{q_i=1}^{Q_i}$, where Q_i is the number of action instances in V_i, and $\varphi_{q_i}, \varphi'_{q_i}, k_{q_i}$ are starting time, ending time and category of action instance ϕ_{q_i}. The task of weakly supervised action localization is to identify the positions of action instances and recognize their categories simultaneously in the input video. Note that only the video-level action labels are available in the training while the temporal annotation of each action instance is no longer required.

B. Winners-Out Strategy

With the video-level action labels at hand, we first train a network using an objective for classification. Now we clarify the formulation of the classification problem. Given an untrimmed video V_i, we divide it into K segments with even temporal interval and obtain a segment set $\mathcal{S}_i = \{s_{i,k}\}_{k=1}^{K}$. For video classification, we aim to integrate the segment-level local feature to the video-level global feature and classify it correctly. To this end, we use an attention module to

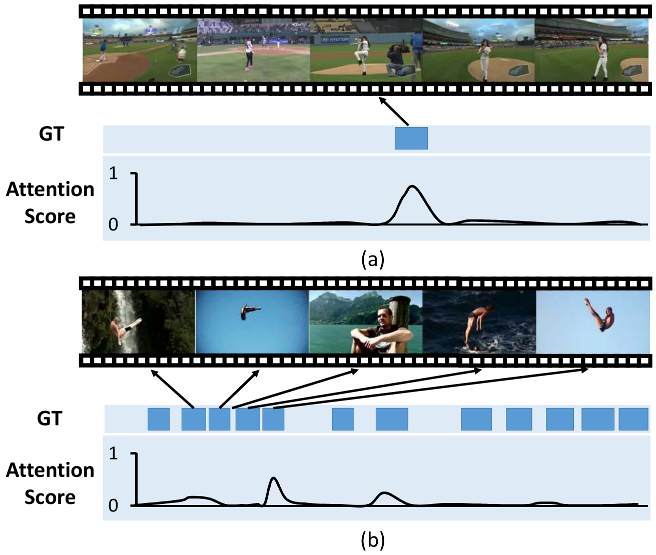


Fig. 3. Examples of the "winner-takes-all" phenomenon. In each example, we show the ground-truth on the top and the attention score of each segment at the bottom. The scores of segments are originally discrete, but we interpolate the scores between segments for better visualization quality. As shown in the first example, in the case where the video contains only one action instance, the attention score is able to localize the ground-truth via attention score. However, in the case with multiple action instances, the most discriminative segments will be assigned with the highest scores and the scores of the less discriminative ones will be suppressed. Thus, this phenomenon prevents us from localizing all the action instances in the video. (a) An example of single ground-truth instance. (b) An example of multiple ground-truth instances.

assign different weights to different segment-level local features. Specifically, we assign an attention weight $a_{i,k}$ to each segment-level feature $\boldsymbol{h}_{i,k}$. Then, the video-level global feature is calculated by

$$\boldsymbol{v}_i = \sum_{k=1}^{K} a_{i,k}\boldsymbol{h}_{i,k}. \tag{1}$$

In our preliminary experiments, we find that the attention module works well for video classification but not for selecting all the action instances in the untrimmed video, which is also empirically found in [29]. During training, only the most discriminative segment features will be selected to represent the video, and neglecting the less discriminative segments will only slightly hurt the classification performance. In this case, the attention module is prone to focus only on the segments with highly-discriminative features in order to optimize classification accuracy instead of identifying all relevant segments, which leads to the "winner-takes-all" phenomenon (Fig. 3).

To break "winner-takes-all", we propose a training strategy, namely winners-out (WO). The core idea is to let the "winners" (*i.e.,* the most discriminative segments) be "out" (*i.e.,* removed from the training video) in the next training step. In this way, the attention module is trained to focus on not only the most discriminative segments but also the less discriminative ones. Note that our "winners-out" strategy is different from the "winners out rule" in [38], which is a rule in the basketball game—giving the ball back to the team that just scored. Meanwhile, our "winners-out" also differs from "adversarial erasing" [19], regardless of their similar goal of removing the most discriminative elements. The "adversarial erasing" replaces the selected pixels with the mean value of training images, while our "winners-out" directly removes the selected segments from the video.

As shown in Fig. 4, WO iteratively performs two operations: training a classification network for localizing the discriminative segments and removing the localized segments in the videos. In particular, in the first operation, we train the classification network with video segments. In the second operation, we fix the parameters of the trained model and calculate an importance score set $\boldsymbol{\lambda}_i^j = \{\lambda_{i,k}^j\}_{k=1}^K$ for each class j of the i-th video. Note that $\boldsymbol{\lambda}_i^j$ is produced by Class-specific Score Computing rather than the attention weight, which will be introduced in § III-D. For each video, we select the segments whose scores belong to top-p of the largest value in $\boldsymbol{\lambda}_i^j$ as "winners" and remove them from the training segment set. Thus, the "winners" segments carrying the most discriminative features of the action class are "out" in the next training step. Then, the processed segment set will be used to train the classification network in the next step. As the discriminative segments have been removed and no longer contribute to the classification prediction, the attention module is naturally driven to select new discriminative segments to maintain its classification accuracy level. We repeat the WO process for several steps until the network cannot converge well on the produced training segment set, *i.e.* no more discriminative segments are left for performing reasonably good classification.

We now illustrate the winners-out strategy more formally. Given the training video set with N videos, we generate N segment sets $\{\mathcal{S}_i\}_{i=1}^N$, where $\mathcal{S}_i = \{s_{i,k}\}_{k=1}^K$ is the segment set of the i-th video. $\mathcal{F} = \{\mathcal{F}_i\}_{i=1}^N$ represents the segment sets selected by the winners-out approach. Let $\mathcal{C}$ be the set of action categories. We iteratively produce the segment set $\mathcal{F}_i$ for each training video V_i at each learning step. Concretely, for $\mathcal{S}_i$, we calculate the j-th importance score set $\boldsymbol{\lambda}_i^j$, in which $j \in \boldsymbol{y}_i$ and $\boldsymbol{y}_i \subseteq \mathcal{C}$ is the video-level label set of V_i. To enforce the classification network to select less discriminative features, we remove the segments whose scores belong to top-p of the largest value in $\boldsymbol{\lambda}_i^j$ in the next training step. The whole procedure is summarized in Algorithm 1.

C. Network Architecture With Group Attention Modules

The proposed winners-out strategy selects the most discriminative segments *w.r.t.* the importance scores. Adopting the attention weights as the importance scores are not precise enough since it will neglect the knowledge learned from the training stage. To this end, we propose a class-discriminative localization technique to better evaluate the importance of the video segments. Since this technique is related to the attention module, we first illustrate the attention module employed in our network. Formally, for an untrimmed video V (the video index i is omitted for simplicity) and the corresponding segment set $\mathcal{S} = \{s_k\}_{k=1}^K$, we randomly extract L frames in each segment and obtain the segment-level local feature vector by $\boldsymbol{h}_k = \frac{1}{L}\sum_{l=1}^{L}\varphi(f_{k,l})$, where $\varphi(\cdot)$ is the feature extractor and $f_{k,l}$ is the l-th extracted frame of the k-th segment. Then, we obtain a set of segment-level features

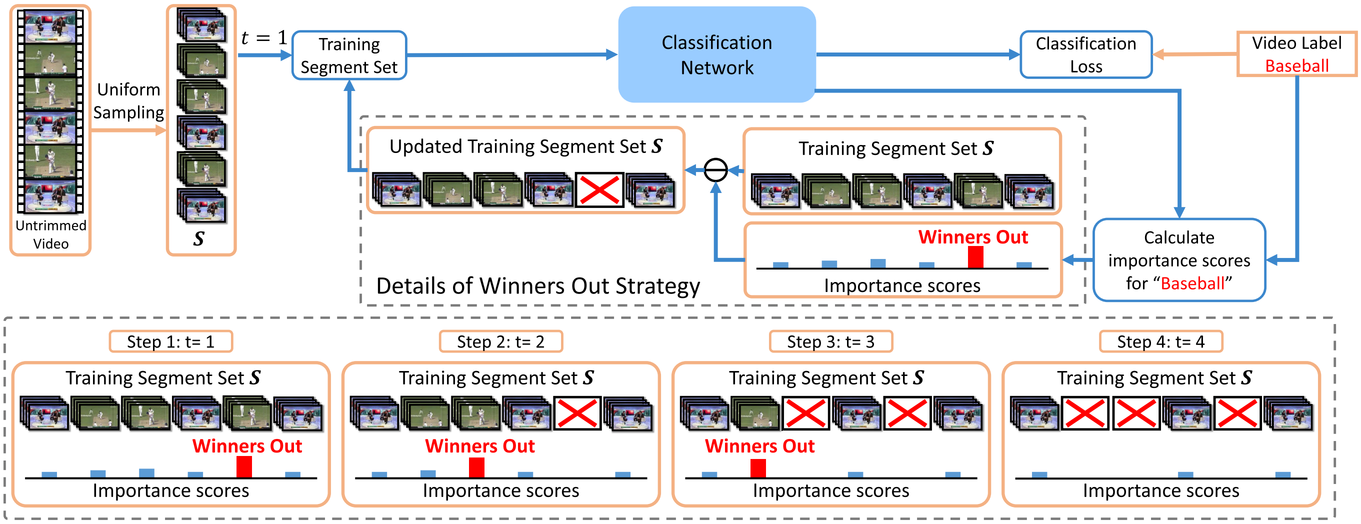


Fig. 4. The proposed winners-out strategy for training the classification network. At Step t, we first train the classification network with the current training segment set $\mathcal{S}$. Then the category information learned from the training stage is employed to assign each segment with a class-specific importance score. The segments whose scores belong to top-p of the largest value (see the red bars) will be selected as "winners" and removed from $\mathcal{S}$, thus leading to the updated training segment set. The updated segment set is then used to re-train the classification network and the segments selected at step t are "out" in the next training step. The training segment set and the corresponding importance scores at each step are shown in the bottom.

Algorithm 1 Winners-Out Strategy for Training Neural Networks

```
Input: Untrimmed video set $\mathcal{V} = \{V_i, \boldsymbol{y}_i\}_{i=1}^N$, segment sets $\{\mathcal{S}_i\}_{i=1}^N$, number of iteration steps $N_{step}$
Let $t = 1$.
while $t < N_{steps}$ do
    Train the classification network $M_c$ on $\{\mathcal{S}_i\}_{i=1}^N$.
    for $\mathcal{S}_i$ in $\{\mathcal{S}_i\}_{i=1}^N$ do
        Set $\mathcal{F}_i = \emptyset$.
        for $j$ in $\boldsymbol{y}_i$ do
            Calculate the importance scores $\boldsymbol{\lambda}_i^j$.
            Select the top-$p$ segments $\mathcal{R}$ according to $\boldsymbol{\lambda}_i^j$.
            Update selected segments set $\mathcal{F}_i = \mathcal{F}_i \bigcup \mathcal{R}$.
        end for
        Remove selected segments from the segment set
        $\mathcal{S}_i = \mathcal{S}_i \setminus \mathcal{F}_i$.
    end for
    $t = t + 1$.
end while
```

$\boldsymbol{H} = (\boldsymbol{h}_1, \boldsymbol{h}_2, \cdots, \boldsymbol{h}_K)$. $\boldsymbol{H}$ has the size K-by-D, where D is the dimensionality of the segment-level feature. The attention module takes $\boldsymbol{H}$ as input and outputs a weight vector by

$$\boldsymbol{a} = \mathrm{softmax}(\boldsymbol{w}\boldsymbol{H}^T + \boldsymbol{b}), \tag{2}$$

where $\boldsymbol{w}$ and $\boldsymbol{b}$ are the learn-able parameters of dimensionality D and K. This vector representation $\boldsymbol{a}$ usually focuses on a specific feature of action. However, there can be multiple types of feature for one action. For example, the number of people who are playing basketball in the video can be different and the color of the playground also varies. To select more types of action features, we need multiple attention modules that focus on different characteristics of actions. As a result, we employ group attention modules in our network. Specifically, we extend $\boldsymbol{w}$ in Eq. (2) into an M-by-D matrix, note it as $\boldsymbol{W}$, and the attention vector $\boldsymbol{a}$ becomes a matrix $\boldsymbol{A}$ with size M-by-K. Formally,

$$\boldsymbol{A} = \mathrm{softmax}(\boldsymbol{W}\boldsymbol{H}^T + \boldsymbol{B}), \tag{3}$$

where $\boldsymbol{B}$ is a learn-able parameter with size M-by-K. Here, the soft-max function is performed along the second dimension of its input. We denote $\boldsymbol{A}_m$ the m-th row of $\boldsymbol{A}$, representing the output attention weights of the m-th attention module. To better initialize the parameters of the attention modules, we use the shifting operation introduced in [39]. The weighted feature vector produced by the m-th attention module is

$$\boldsymbol{v}_m = \frac{\alpha_m \boldsymbol{A}_m \boldsymbol{H}}{\sqrt{M}\|\alpha_m \boldsymbol{A}_m \boldsymbol{H}\|_2}, \tag{4}$$

where α_m is a learn-able scalar. Lastly, we obtain the video-level feature by concatenating the outputs of all attention modules:

$$\boldsymbol{g} = [\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_M]. \tag{5}$$

The resulting global feature is a vector of dimensionality MD. By feeding the global feature to the classifier, we obtain the video-level prediction $\hat{\boldsymbol{y}}$. In the case of video containing action instances from multiple classes, we normalize the video label vector $\boldsymbol{y}$ by $\bar{\boldsymbol{y}} = \boldsymbol{y}/\|\boldsymbol{y}\|_1$ to ensure that the sum of probabilities is 1. Since we have multiple attention modules, the attention matrix $\boldsymbol{A}$ can suffer from redundancy problems if some modules focus on the same action pattern. Thus, we use the penalization term suggested by [40] to encourage diversity across different attention modules. Then the objective function is formulated as:

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{i=1}^{N}\sum_{j=1}^{C} \bar{y}_i^j \log(\hat{y}_i^j) + \beta \|\boldsymbol{A}\boldsymbol{A}^T - \boldsymbol{I}\|_F^2, \quad (6)$$

where $\boldsymbol{\theta}$ is the parameter of the neural network. Here, $\|\cdot\|_F$ is the Frobenius norm of a matrix and β is a hyper-parameter to trade-off the cross-entropy loss and the penalization term. We set $\beta = 0.1$ in all our experiments and find that it works well across all of them.

D. Class-Specific Score Computing

With the network introduced above, we are able to train it using the video-level labels. By training such a network for untrimmed video classification, the attention modules learn to select the discriminative features among segments. However, as we claimed in § II, the attention weight is class-agnostic and thus it is impractical to directly use the attention weight for our winners-out strategy.

To better select the segments carrying discriminative features of action, we propose a class-discriminative localization technique, namely **Class-specific Score Computing (CSC)**. We denote $\boldsymbol{z}_m^j$ the weight vector connecting the output of the m-th attention module and the score of class j in the fully-connected layer, and thus the input to the final soft-max layer for class j is

$$\begin{aligned}\hat{y}^j &= \sum_{m=1}^{M} \boldsymbol{v}_m \boldsymbol{z}_m^j \\ &= \sum_{m=1}^{M} \frac{\alpha_m \boldsymbol{A}_m \boldsymbol{H}}{\sqrt{M}\|\alpha_m \boldsymbol{A}_m \boldsymbol{H}\|_2} \boldsymbol{z}_m^j \\ &= \sum_{m=1}^{M} (\frac{\alpha_m}{\sqrt{M}\|\alpha_m \boldsymbol{A}_m \boldsymbol{H}\|_2} \sum_{k=1}^{K} a_{m,k} \boldsymbol{h}_k) \boldsymbol{z}_m^j \\ &= \sum_{k=1}^{K}\sum_{m=1}^{M} \frac{\alpha_m a_{m,k}}{\sqrt{M}\|\alpha_m \boldsymbol{A}_m \boldsymbol{H}\|_2} \boldsymbol{h}_k \boldsymbol{z}_m^j, \quad (7)\end{aligned}$$

where $a_{m,k}$ is the weight of the k-th segment generated by the m-th attention module. From Eq. (7), we obtain the importance score of the k-th segment to the j-th class:

$$\lambda_k^j = \text{ReLU}\left(\sum_{m=1}^{M} \frac{\alpha_m a_{m,k}}{\sqrt{M}\|\alpha_m \boldsymbol{A}_m \boldsymbol{H}\|_2} \boldsymbol{h}_k \boldsymbol{z}_m^j\right). \quad (8)$$

We apply a ReLU to the importance score since we are only interested in the segment that has a positive influence on the class of interest, *i.e.*, segment which have a negative contribution to the classification score $\hat{y}^j$ will be ignored. Although we have multiple attention modules and use penalization term in Eq. (6) to encourage each attention module to learn different types of action features, we still lack control over which attention module learns information about which category of action. As a result, we sum up the importance scores provided by all the attention modules in Eq. (8).

With Eq. (8), our proposed method can be used to analyze the importance of each segment and generate CSC scores to the specific action class j via

$$\boldsymbol{\lambda}^j = \left(\lambda_1^j, \lambda_2^j, \cdots, \lambda_K^j\right). \quad (9)$$

Then, at each winners-out step, the segments whose CSC scores belong to top-p of the largest value will be removed. In contrast to the attention weights, our CSC makes full use of the category information and provides a better reference for selecting the most discriminative segments.

E. Discussion

Contribution beyond [29], [30]. The core idea of [29] is Eq. (1), where the simplest form of attention module is adopted. Note that only one attention module is trained to assign weight λ to each video segment $\boldsymbol{h}$, which represents the probability of that segment containing actions. In our network, we use multiple attention modules as discussed in § III-C and empirically show that using multiple attention modules outperforms that with only a single attention module.

It is worth noting that, in [29] the final attention weight of each segment is normalized to [0,1] by applying softmax function to the original attention weights of all the segments in that video (Eq. (2)). In this way, only a few segments will be assigned with large weights and other segments will get scores that are much smaller, and thus leading to the "winner-takes-all" issue. In our method, though we use softmax function to yield the normalized weight, we apply the winners-out strategy to remove the winner segments after each training step, preventing the network from only concentrating on the most discriminative segment. Though [30] also tries to solve the "winner-takes-all" problem, due to the lack of evaluating the importance of each video segment, it hides the segments randomly. However, this random strategy may bring two drawbacks: 1) The hidden segments can be action or background instances. It may force the network to output high response on the background instances. 2) Randomly hiding is extremely inefficient. It takes much more time to train the network. In contrast, our proposed CSC technique is able to assign the discriminative segment with a large score and thus the most discriminative ones can be removed from the training video. With this goal-oriented operation, we tackle the "winner-takes-all" problem in both efficient and effective way.

IV. Training and Testing

A. Training Details

The two key components, winners-out strategy and Class-specific Score Computing, are finally integrated into an end-to-end iterative-winners-out network. We extract multi-modal features including appearance feature (RGB frames) $\boldsymbol{H}_{\text{rgb}}$ and motion feature (Optical flow) $\boldsymbol{H}_{\text{flow}}$ from the input video. We use a classical two-stream architecture and the overview of our network is shown in Fig. 5.

Multi-modal Concatenation. By feeding features into the network, each stream outputs a video-level feature by Eq. (3)-(5). Then, the RGB feature $\boldsymbol{g}_{\text{rgb}}$ and the flow feature $\boldsymbol{g}_{\text{flow}}$ are concatenated into the final video-level feature $\boldsymbol{g}_{\text{final}} = [\boldsymbol{g}_{\text{rgb}}, \boldsymbol{g}_{\text{flow}}]$.

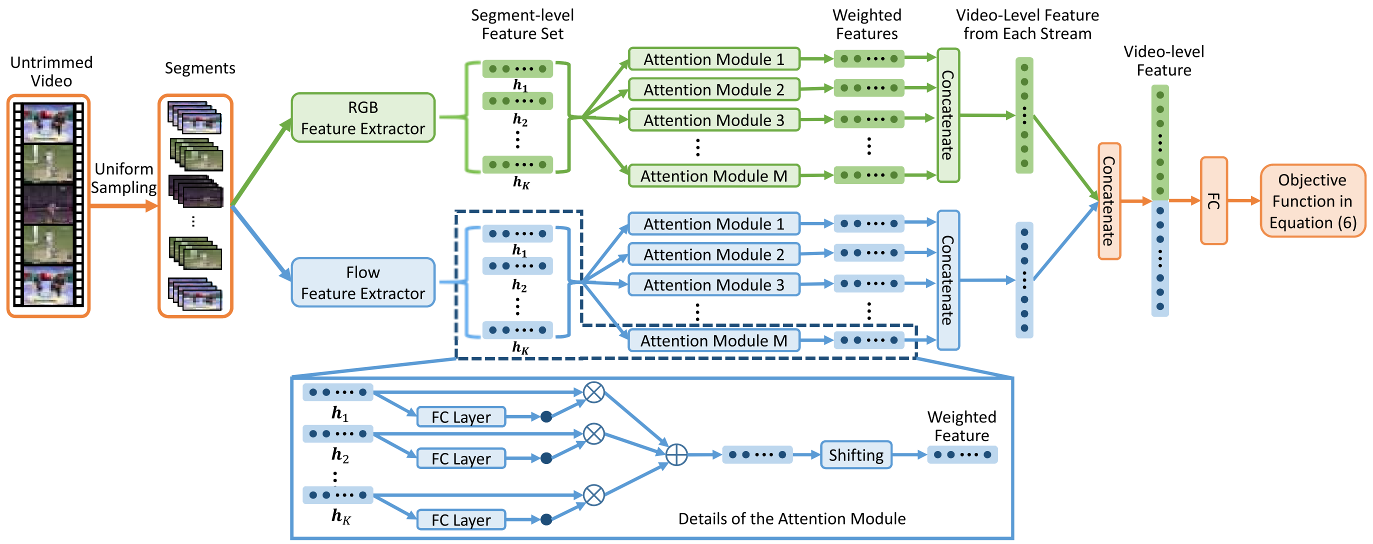


Fig. 5. The network architecture of the two-stream iterative-winners-out network with group attention modules. We employ the two-stream architecture with multiple attention modules for each stream. We first divide the input video into segments evenly and then obtain the segment-level feature set via the feature extractor. Each attention module with the shifting operation takes the feature set as input and will output a weighted feature vector. We then concatenate the outputs of all the attention modules in each stream. Finally, the outputs of two streams will be concatenated into a final video-level feature vector.

Algorithm 2 Winners-Out Testing

```
Input: Segment set $\mathcal{S} = \{s_k\}_{k=1}^{K}$, trained model $M_c$,
number of iteration steps $N_{step}$
Video Prediction: Choose top-$q$ classes as $\hat{\boldsymbol{y}}$
for $j$ in $\hat{\boldsymbol{y}}$ do
    Initialize: $\mathcal{F}^j = \emptyset$, $\mathcal{S}^j = \mathcal{S}$; Let $t = 1$.
    while $t < N_{step}$ do
        Obtain importance scores $\boldsymbol{\lambda}^j$ via CSC$(\mathcal{S}^j, M_c, j)$.
        Select the top-$p$ segments $\mathcal{R}$ according to $\boldsymbol{\lambda}^j$.
        Update the selected segment set $\mathcal{F}^j = \mathcal{F}^j \bigcup \mathcal{R}$.
        Update the testing segment set $\mathcal{S}^j = \mathcal{S}^j \setminus \mathcal{R}$.
        $t = t + 1$.
    end while
end for
Output: $\{\mathcal{F}^j\}(j \in \hat{\boldsymbol{y}})$
```

Multi-modal CSC scores. For each stream, we calculate CSC scores via Eq. (8) and (9). Then, the final CSC scores are obtained by the weighted sum of scores from two streams:

$$\boldsymbol{\lambda}^j_{\text{final}} = \boldsymbol{\lambda}^j_{\text{rgb}} + \gamma\, \boldsymbol{\lambda}^j_{\text{flow}}, \tag{10}$$

where γ is a trade-off parameter.

B. Testing and Post-Processing

1) Winners-Out Testing: In the testing stage, for each video V and the corresponding segment set $\mathcal{S} = \{s_k\}_{k=1}^{K}$, we feed segment-level features to the trained network and obtain the video-level prediction scores. Since one video may have multiple action labels, we choose q action classes with the largest prediction scores as the predicted labels and note it as $\hat{\boldsymbol{y}}$. For each class $j \in \hat{\boldsymbol{y}}$, we first initialize the testing segment set $\mathcal{S}^j = \mathcal{S}$ and use CSC to calculate the importance scores $\boldsymbol{\lambda}^j$ at step t. We then select segments $\mathcal{R}$ whose scores belong to top-p of the largest value in $\boldsymbol{\lambda}^j$ and the selected segments will be added to a segment set $\mathcal{F}^j$. In the meantime, these segments will be removed from the testing segment set and thus leading to $\mathcal{S}^j = \mathcal{S}^j \setminus \mathcal{R}$ for the next testing step. Then, at step $t+1$, $\mathcal{S}^j$ will be fed into the network and repeat the aforementioned process. Finally, we obtain the selected segment sets $\{\mathcal{F}^j\}(j \in \hat{\boldsymbol{y}})$. Since this process builds upon winners-out, we name it Winners-out Testing. The whole procedure is summarized in Algorithm 2.

2) Post-Processing: For each class j, we connect the adjacent segments in $\mathcal{F}^j$ to form the action proposals. In action localization, the proposal is a one-dimensional time interval that potentially contains multiple segments, with the class labels and scores. We define the proposal by $[k_{\text{start}}, k_{\text{end}}]$, where k_{start} and k_{end} denote for the index of the first segment and the last segment within the proposal, respectively. Thus, each proposal is composed of $(k_{\text{end}} - k_{\text{start}} + 1)$ segments. We assign each proposal with a score as the mean CSC score of all the segments within the proposal:

$$\sum_{k=k_{\text{start}}}^{k=k_{\text{end}}} \frac{\lambda^j_{k,\text{final}}}{k_{\text{end}} - k_{\text{start}} + 1}. \tag{11}$$

This value corresponds to the proposal score for class j. Finally, we take the generated proposals as our action localization results.

V. Experiments

A. Datasets and Evaluation Metrics

1) THUMOS14 [23]: is composed of four parts: training, validation, testing and background sets. The training set is the UCF-101 dataset with 13320 trimmed videos of 101 categories. The validation set and test set contain 1010 and

1574 untrimmed videos, respectively. The background set contains 2500 videos. The temporal action localization task of THUMOS14 dataset, which contains videos over 20 hours from 20 sports classes, is challenging and widely used. Following the common setting in [23], we apply 200 videos in the validation set for training. Here, we only adopt the video-level labels to ensure the weakly supervision contraction. The performance is evaluated on the testing set that contains 213 videos.

2) ActivityNet [24]: is a standard benchmark for action localization in untrimmed videos and it provides rich and diverse action categories. We evaluate our method on ActivityNet1.3, which is the largest and also the most challenging version. It involves 200 activities and contains around 10K training videos and 5K validation videos. Each video has an average of 1.65 action instances with temporal annotations. As a standard practice, we train on the training videos and test on the validation videos. Note that we only use video-level labels for the weakly supervised setting.

3) Evaluation Metrics: We use the mean Average Precision (mAP) as the comparison metric. Following the conventional evaluation set-ups, we report the mAP at different IoU thresholds. A predicted proposal is correct if it gets the same category as ground-truth and its temporal IoU with this ground-truth instance is larger than the IoU threshold. On THUMOS14, the IOU thresholds are chosen from [0.1, 0.2, 0.3, 0.4, 0.5]; on ActivityNet1.3, the IOU thresholds are determined over [0.5, 0.75, 0.95].

B. Implementation Details

For each input video, we set K to 400 for a fair comparison with [29]. We also conduct ablation study in § VI-E to explore other choices of K. We set L to 3 in our experiments. We choose the Inception architecture [41] with Batch Normalization, namely BN-Inception, to be the feature extractor. The feature extractor is pre-trained on Kinetics dataset [42] and is froze when training on the target dataset. The input for the RGB and Flow stream is 1-frame RGB image and 5-frame stacks of TV-L1 [43] optical flows, respectively. M and p are set to 18 and 30, respectively. The initial learning rate is 0.001 and will be decreased to its $\frac{1}{10}$ every 20 epochs. The dropout ratio is set as 0.8. γ in Eq. (10) is set to 10. Through experiments, we find that removing 30 segments at each winners-out step produces the best results. In the testing stage, we choose the top-2 classes to perform action localization and set the number of winners-out steps to 10 for all the videos. Our network is trained by back-propagation and stochastic gradient descent (SGD) [44].

C. Comparison With State-of-the-Arts

1) ActivityNet1.3 Dataset: We compare our method with state-of-the-arts, including supervised and weakly supervised methods. Ours-A uses the same feature extractor (*i.e.,* I3D network [14] pre-trained on Kinetics dataset) as [31] and K is set to 400 for fair comparisons. Ours-B and Ours-C use BN-Inception network (denoted by 2D) pre-trained on Kinetics as the feature extractor with $K = 400$ and $K = 200$, respectively. The results are shown in Table I. Ours-A outperforms Nguyen *et al.* [31] in most IoU thresholds under the same experimental settings. When IoU = 0.95, we improve the result from 2.6% to 4.8%. Ours-B achieves results that are comparable to [31] with a weaker feature extractor. Ours-C achieves the state-of-the-art performance in all IoU thresholds. Interestingly, even with weak supervision only, our method outperforms [25] (22.5%) and R-C3D (26.8%) [4] when IoU is set to 0.5. Note that R-C3D [4] and [25] apply frame-level annotations for training while we only use the video-level labels as supervision.

TABLE I

COMPARISON WITH STATE-OF-THE-ART ACTION LOCALIZATION METHODS ON ACTIVITYNET1.3 VALIDATION SET. WE SHOW THE EXPERIMENTAL SETTINGS OF OUR MODEL IN THE FORM OF (FEATURE EXTRACTOR-NUMBER OF SEGMENTS-PRETRAINING DATASET)

Supervision	Method	mAP@IoU		
		0.5	0.75	0.95
Fully supervised	Montes *et al.* [25]	22.5	-	-
	Xu *et al.* [4]	26.8	-	-
Weakly supervised	Nguyen *et al.* [31]	29.3	16.9	2.6
	Ours-A (I3D-400-Kinetics)	27.2	17.5	4.8
	Ours-B (2D-400-Kinetics)	27.0	15.7	4.0
	Ours-C (2D-200-Kinetics)	**29.8**	**17.6**	**4.7**

TABLE II

COMPARISON WITH STATE-OF-THE-ART ACTION LOCALIZATION METHODS ON THUMOS14 TESTING SET. WE SHOW THE EXPERIMENTAL SETTINGS OF OUR MODEL IN THE FORM OF (FEATURE EXTRACTOR-NUMBER OF SEGMENTS-PRETRAINING DATASET)

Supervision	Method	mAP@IoU				
		0.1	0.2	0.3	0.4	0.5
Fully supervised	Richard *et al.* [46]	39.7	35.7	30.0	23.2	15.2
	Yeung *et al.* [47]	48.9	44.0	36.0	26.4	17.1
	Yuan *et al.* [48]	51.0	45.2	36.5	27.8	17.8
	S-CNN [3]	47.7	43.5	36.3	28.7	19.0
	CDC [5]	-	-	40.1	29.4	23.3
	R-C3D [4]	54.5	51.5	44.8	35.6	28.9
	SSN [6]	66.0	59.4	51.9	41.0	29.8
Weakly supervised	UntrimmedNet [29]	44.4	37.7	28.2	21.1	13.7
	Hide and Seek [30]	36.4	27.8	19.5	12.7	6.8
	Nguyen et al. [31]	52.0	44.7	35.5	25.8	16.9
	Ours-A (I3D-400-Kinetics)	51.3	43.1	34.1	26.3	18.0
	Ours-B (2D-400-Kinetics)	54.5	45.0	35.1	25.2	17.0
	Ours-C (2D-800-Kinetics)	**57.6**	**48.9**	**38.9**	**29.3**	**20.5**
	Ours-D (2D-400-ImageNet)	49.5	42.1	32.0	22.5	15.4

2) THUMOS14 Dataset: We compare our proposed iterative-winners-out network with state-of-the-art supervised and weakly supervised action localization methods in Table II. Ours-A, B, C have the same settings as those on ActivityNet except that K in Ours-C is set to 800. For fair comparisons, we add a model Ours-D in which the feature extractor is the same as that is used in UntrimmedNet [29], *i.e.,* a model pre-trained on ImageNet [45]. With the same I3D feature and number of segments, Ours-A significantly outperforms Nguyen *et al.* [31] when the IoU thresholds are set to 0.5. Although the feature extractor in Ours-B is weaker than that in [31], Ours-B still achieves a comparable performance. Ours-C improves the previous state-of-the-art result from 16.9% to 20.5% when IoU = 0.5. Ours-D achieves better performance than UntrimmedNet under the same settings, verifying the effectiveness of our winners-out strategy.

TABLE III

COMPARISON OF MAP (IOU = 0.5) IN DIFFERENT WINNERS-OUT STEPS

Setting	THUMOS14	Gain	ActivityNet1.3	Gain
WO-Step1	14.8	-	24.3	-
WO-Step2	**17.0**	**2.2**	**27.0**	**2.7**
WO-Step3	15.8	1.0	26.2	1.9

TABLE IV

COMPARISON OF MAP (IOU = 0.5) USING WINNERS-OUT IN THE TRAINING AND TESTING STAGES

Setting	THUMOS14	ActivityNet1.3
Train + Test	14.2	23.1
Train + WO Test	14.8	25.6
WO Train + Test	14.4	24.3
WO Train + WO Test	**17.0**	**27.0**

Note that UntrimmedNet uses 1010 videos in the validation set for training, while we use 200 videos of the validation set.

VI. ABLATION STUDIES

A. Understanding of Winners-out Steps

Here, we perform experiments to understand the contribution of each WO step. From Table III, the action localization performance indeed increases as more WO steps are performed. As the most discriminative segments have been removed from the video, the classification network is encouraged to select other discriminative segments. However, performing WO for too many steps will hurt the performance since it may select negative segments.

We visualize the segments selected by each WO step in Fig. 6. In the example of "Mowing the lawn", the network tends to focus on the scene containing the human, the lawn mower, and the grass in the first WO step. In the second WO step, since the most discriminative segments are removed from the video, the network concentrates on the segments with incomplete visual components (*e.g.,* the other half of the actor is outside the scope of the camera). In the third WO step, the network chooses the segment where only the grass is contained to represent the whole video. It is obvious that the negative segments have been selected due to over removing. Other examples in Fig. 6 are consistent with this observation.

B. The Effectiveness of Winners-Out Strategy

To evaluate the effectiveness of WO strategy, we conduct several contrast experiments: (1) Train and test regularly; (2) Train regularly and test with WO; (3) Train with WO and test regularly; (4) Train and test with WO. The results are listed in Table IV. The performance is significantly improved from 14.2% to 17.0% and 23.1% to 27.0% on two datasets when WO strategy is used in the training and testing stages simultaneously. It is evident that the winners-out strategy serves as a very important component in our proposed method and brings significant gains. (2) and (3) outperform (1) by 0.6% and 0.2% on THUMOS14, respectively. The improvement is not significant when using WO in the training and testing stages individually. The reason may be that the "winner-takes-all" phenomenon exists in both the training and testing stages.

TABLE V

COMPARISON BETWEEN CAM AND CSC ON THUMOS14, MEASURED BY ACTION LOCALIZATION MAP

IoU	0.1	0.2	0.3	0.4	0.5
CAM	48.6	41.5	32.5	24.6	16.2
CSC	**54.5**	**45.0**	**35.1**	**25.2**	**17.0**

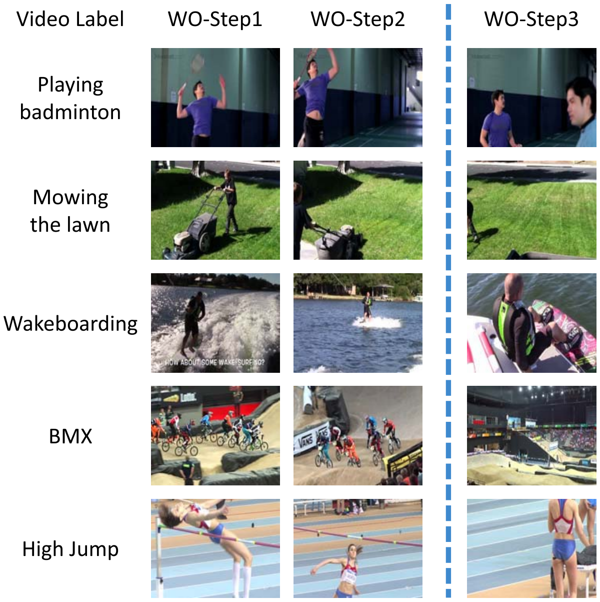


Fig. 6. The "winner" segments at different winners-out steps. Each row shows an example of the training videos and each column shows the segment with the highest CSC score at different WO steps. Here, we choose the center frame of the segment for visualization. Segments in WO-Step3 are the failure cases due to performing WO for too many steps.

In setting (2), training without our WO strategy leads to a network that only focuses on the most discriminative segments. Even if we employ WO in the testing stage, the difference between the less discriminative action features and the background features are not significant enough to separate them. In setting (3), the network is able to break the "winner-takes-all" phenomenon with the help of WO strategy in the training stage. However, performing testing for only one iteration could only select the most discriminative segments and thus produce a 0.2% improvement only. Overall, these results demonstrate that the winners-out strategy is able to break the "winner-takes-all" effectively, and employing it in both stages simultaneously leads to the state-of-the-art results.

C. The Effectiveness of CSC

1) CSC v.s. CAM: As discussed in § I, compared to CAM, our proposed CSC takes the attention weight of each segment as additional information. To verify this, we compare CSC with CAM under the same setting. Here, we remove the attention modules and perform average pooling over segment features to obtain the global feature. Following [39], we also replicate the global feature for M times. From Table V, CSC outperforms CAM under all IoU thresholds. The CAM method performs average pooling to obtain the global representation,

TABLE VI

ACTION LOCALIZATION RESULTS ON THUMOS14 WITH A DIFFERENT NUMBER OF ATTENTION MODULES, MEASURED BY mAP@IoU = 0.5

M	1	5	10	16	18	20	22	24	30
mAP	14.30	15.26	16.10	16.75	**17.02**	16.96	16.54	16.51	16.54

and each pixel/segment is treated equally. In contrast, the attention module in our CSC predicts a higher attention weight on the segment that is more discriminative. In this way, CSC is able to select the discriminative segments more precisely and boost the action localization performance eventually.

2) CSC v.s. Attention: Compare to [29], our proposed CSC further integrates the activation of each segment and the weights of fc layer besides the attention weights. Here, we perform experiments to verify this. For a fair comparison, the only modification is replacing CSC scores with attention weights. We find that WO with CSC significantly outperforms that with attention weights (17.0% v.s. 15.7%). We also visualize action localization results on THUMOS14 and ActivityNet 1.3 in Fig. 7 to figure out how this improvement is obtained. Fig. 7 (a) shows an example with many action instances. Despite this huge challenge, our proposed CSC is able to localize most of the action instances and their boundaries. Though the model using the attention weights is able to find some of the action instances, it always fails to recognize the boundaries of actions. This demonstrates that considering the activation of each segment and the weights of fc layer makes it easier to identify the start time and end time of the action instance, which is important in this task. In Fig. 7 (c), we show an example where the appearance of the video frames are similar; nevertheless, our CSC still localizes the target action precisely. When employing attention weights, one of the detection results matches the ground-truth action instance. However, the attention weights fail to distinguish the action and background instances which are similar in appearance and thus produce some negative results. We can see that benefiting from the category information learned from the training stage, CSC provides a more precise reference for selecting discriminative action instances.

D. The Effectiveness of Group Attention Modules

1) The Number of Attention Modules: Since we employ group attention modules in our model, we conduct an ablation study to explore how does M affect the performance. We show the results in TableVI. The action localization mAP first increases with more attention modules. This is consistent with the results in [39] that using more attention modules is able to capture more types of action patterns. Our model performs well when M is 18 and 20. However, when M is larger than 20, the mAP decreases. A possible reason is that the network will be hard to train with more parameters. In our experiments, we set M to 18 unless otherwise specified.

2) Group Attention v.s. Single Attention: From Table VI, our method achieves better results when using group attention modules ($M > 1$). The reason may be that training only one attention module to learn the action patterns for multiple classes is difficult. As a result, the capability of the single attention module is limited. In contrast, with our design of group attention modules, each attention module can focus on limited types of action patterns instead of learning general patterns for all actions. Furthermore, the final output of our model is obtained by concatenating the outputs of multiple attention modules. This design brings better video-level representations and enables our proposed CSC to determine the removed segments by considering attention weights from multiple attention modules, leading to more robust decision.

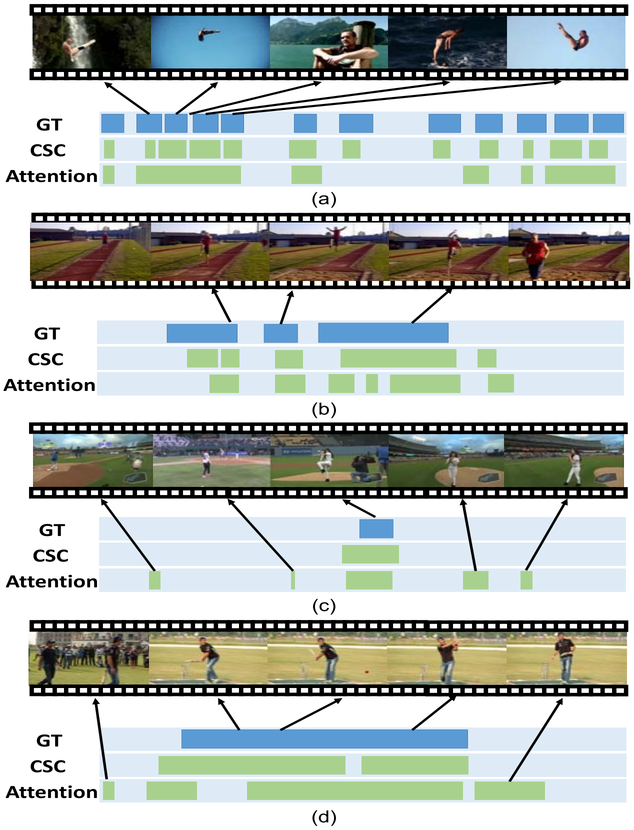
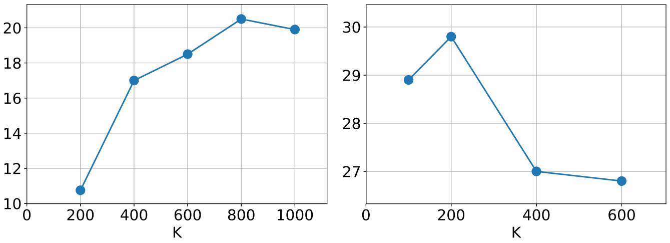


Fig. 7. Visualization of action localization results. Horizontal axis stands for time. The blue boxes denote the ground-truth action instances and the green boxes stand for the predicted results of our model using CSC and attention weights. (a) and (b): There are multiple action instances in the input video; nevertheless, our method with CSC is able to localize most of the actions and recognize the boundaries simultaneously. However, using attention weights always fails to detect the boundaries of actions. (c) and (d): The frames of the input video remain similar. Our CSC is able to localize the segments containing actions precisely while using attention weights brings negative results. (a) Example 1: Cliff Diving in THUMOS14. (b) Example 2: Long Jump in THUMOS14. (c) Example 3: Baseball Pitch in THUMOS14. (d) Example 4: Baseball Pitch in ActivityNet.

3) Comparison Between Our Attention Module and [39].: In our model, we modify the attention module in [39] by removing the parameter β. The parameter β is a learnable scalar added on the attended features and is unrelated to the segment features. Thus, it will not affect the importance of each segment in our proposed CSC. In other words, Eq. (8) will not change when β is added. We conduct an experiment where β is added to our attention module and show the results in Table VII. The performance difference is small compared to our original model, demonstrating that removing β does not harm the performance while enjoying fewer parameters.

TABLE VII
ACTION LOCALIZATION RESULTS ON THUMOS14 WITH THE PARAMETER β, MEASURED BY MAP AT DIFFERENT IOU THRESHOLDS

IoU	0.1	0.2	0.3	0.4	0.5
w/ β	54.8	45.6	35.1	24.5	16.8
w/o β	54.5	45.0	35.1	25.2	17.0



(a) (b)

Fig. 8. Action localization results with different number of segments K, measured by mAP@IoU = 0.5. (a) THUMOS14. (b) ActivityNet1.3.

E. The Number of Video Segments

The proposed iterative-winners-out network aims to select segments to form the action instances. Concerning how does K affect the action localization results, we perform experiments with different K and show the results in Fig. 8. Our model achieves the best result by using 800 segments on THUMOS14 and 200 segments on ActivityNet. As discussed in [49], each video on THUMOS14 has 15 action instances on average and 71% of actions are shorter than 2% of the video length. Therefore, more segments help locate the short actions on THUMOS14. In contrast, each video on ActivityNet has only 1.5 instances on average and more than 64% frames are actions. Fewer segments are able to alleviate the effects of false negative segments and thus boost the localization performance on ActivityNet 1.3.

VII. CONCLUSIONS

We have addressed the weakly supervised action localization problem by developing an iterative-winners-out network that is inspired by the Adversarial Erasing semantic segmentation network. Our method features two key components that address the shortcomings of existing approaches. One is the winners-out training strategy, the other one is a class-discriminative localization technique, namely Class-specific Score Computing (CSC). When given the video-level action labels only, our method learns to localize the discriminative action instances for each action class. Our proposed method yields state-of-the-art performance on two standard benchmark datasets THUMOS14 and ActivityNet1.3. One future direction to enhance our network could be considering more advanced feature representation methods.

REFERENCES

[1] D. Oneata, J. Verbeek, and C. Schmid, "Action and event recognition with Fisher vectors on a compact feature set," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 1817–1824.

[2] J. Yuan, B. Ni, X. Yang, and A. A. Kassim, "Temporal action localization with pyramid of score distribution features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3093–3102.

[3] Z. Shou, D. Wang, and S.-F. Chang, "Temporal action localization in untrimmed videos via multi-stage CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1049–1058.

[4] H. Xu, A. Das, and K. Saenko, "R-C3D: Region convolutional 3D network for temporal activity detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 5783–5792.

[5] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5734–5743.

[6] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin, "Temporal action detection with structured segment networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2914–2923.

[7] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2678–2687.

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

[9] L. Fan, W. Huang, C. Gan, S. Ermon, B. Gong, and J. Huang, "End-to-end learning of motion representation for video understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 6016–6025.

[10] 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.*, Dec. 2015, pp. 4489–4497.

[11] C. Gan, N. Wang, Y. Yang, D.-Y. Yeung, and A. G. Hauptmann, "DevNet: A deep event network for multimedia event detection and evidence recounting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 2568–2577.

[12] L. Wang *et al.*, "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. 14th Eur. Conf. Comput. Vis. (ECCV)*. Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 20–36. doi: 10.1007/978-3-319-46484-8_2.

[13] W. Huang *et al.*, "Toward efficient action recognition: Principal backpropagation for training two-stream networks," *IEEE Trans. Image Process.*, vol. 28, no. 4, pp. 1773–1782, Apr. 2019.

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

[15] P. Bilinski and F. Bremond, "Video covariance matrix logarithm for human action recognition in videos," in *Proc. Int. Joint Conf. Artif. Intell.*, Jul. 2015, pp. 2140–2147.

[16] C. Gan, B. Gong, K. Liu, H. Su, and L. J. Guibas, "Geometry guided convolutional neural networks for self-supervised video representation learning," in *Proc. CVPR*, Jun. 2018, pp. 5589–5597.

[17] Y. Guo, Q. Chen, J. Chen, Q. Wu, Q. Shi, and M. Tan, "Auto-embedding generative adversarial networks for high resolution image synthesis," *IEEE Trans. Multimedia*, 2019. doi: 10.1109/TMM.2019.2908352.

[18] C. Gan, T. Yao, K. Yang, Y. Yang, and T. Mei, "You lead, we exceed: Labor-free video concept learning by jointly exploiting Web videos and images," in *Proc. CVPR*, Jun. 2016, pp. 923–932.

[19] Y. Wei, J. Feng, X. Liang, M.-M. Cheng, Y. Zhao, and S. Yan, "Object region mining with adversarial erasing: A simple classification to semantic segmentation approach," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1568–1576.

[20] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[21] C. Deng, Q. Wu, Q. Wu, F. Hu, F. Lyu, and M. Tan, "Visual grounding via accumulated attention," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7746–7755.

[22] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2921–2929.

[23] Y.-G. Jiang *et al.* (2014). *THUMOS Challenge: Action Recognition with a Large Number of Classes*. [Online]. Available: http://crcv.ucf.edu/THUMOS14/

[24] F. C. Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles, "ActivityNet: A large-scale video benchmark for human activity understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 961–970.

[25] A. Montes, A. S. Aguilera, S. Pascual, and X. G. Nieto, "Temporal activity detection in untrimmed videos with recurrent neural networks," in *Proc. 1st NIPS Workshop Large Scale Comput. Vis. Syst. (LSCVS)*, 2016, pp. 1–5.

[26] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang, “Real-time action recognition with enhanced motion vector CNNs,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2718–2726.

[27] X. Dai, B. Singh, G. Zhang, L. S. Davis, and Y. Q. Chen, “Temporal context network for activity localization in videos,” in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 5793–5802.

[28] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles, “End-to-end, single-stream temporal action detection in untrimmed videos,” in *Proc. Brit. Mach. Vis. Conf.*, 2017, p. 2.

[29] L. Wang, Y. Xiong, D. Lin, and L. Van Gool, “Untrimmednets for weakly supervised action recognition and detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 4325–4334.

[30] K. K. Singh and Y. J. Lee, “Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Oct. 2017, pp. 3524–3533.

[31] P. Nguyen, T. Liu, G. Prasad, and B. Han, “Weakly supervised action localization by sparse temporal pooling network,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2018, pp. 6752–6761.

[32] H. Bilen and A. Vedaldi, “Weakly supervised deep detection networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 2846–2854.

[33] B. Lai and X. Gong, “Saliency guided end-to-end learning forweakly supervised object detection,” in *Proc. Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2053–2059.

[34] Z. Jie, Y. Wei, X. Jin, J. Feng, and W. Liu, “Deep self-taught learning for weakly supervised object localization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 1377–1385.

[35] H. Bilen, M. Pedersoli, and T. Tuytelaars, “Weakly supervised object detection with convex clustering,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1081–1089.

[36] W. Ren, K. Huang, D. Tao, and T. Tan, “Weakly supervised large scale object localization with multiple instance learning and bag splitting,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 2, pp. 405–416, Feb. 2016.

[37] Z. Zhuang *et al.*, “Discrimination-aware channel pruning for deep neural networks,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 875–886.

[38] S. C. Albright and W. Winston, “A probabilistic model of winners’ outs versus losers’ outs rules in basketball,” *Oper. Res.*, vol. 26, no. 6, pp. 1010–1019, 1978.

[39] 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 Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7834–7843.

[40] Z. Lin *et al.*, “A structured self-attentive sentence embedding,” Mar. 2017, *arXiv:1703.03130*. [Online]. Available: https://arxiv.org/abs/1703.03130

[41] 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.

[42] W. Kay *et al.*, “The kinetics human action video dataset,” May 2017, *arXiv:1705.06950*. [Online]. Available: https://arxiv.org/abs/1705.06950

[43] C. Zach, T. Pock, and H. Bischof, “A duality based approach for realtime TV-L^1 optical flow,” in *Proc. 29th DAGM Symp. Pattern Recognit.* Heidelberg, Germany: Springer, 2007, pp. 214–223. doi: 10.1007/978-3-540-74936-3_22.

[44] Y. LeCun *et al.*, “Backpropagation applied to handwritten zip code recognition,” *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[45] 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.

[46] A. Richard and J. Gall, “Temporal action detection using a statistical language model,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 3131–3140.

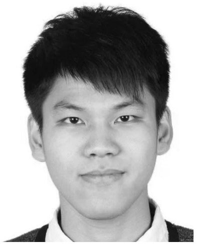
[47] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei, “Every moment counts: Dense detailed labeling of actions in complex videos,” *Int. J. Comput. Vis.*, vol. 126, nos. 2–4, pp. 375–389, 2018.

[48] Y. Yuan, X. Liang, X. Wang, D.-Y. Yeung, and A. Gupta, “Temporal dynamic graph LSTM for action-driven video object detection,” in *Proc. ICCV*, Oct. 2017, pp. 1801–1810.

[49] H. Alwassel, F. C. Heilbron, V. Escorcia, and B. Ghanem, “Diagnosing error in temporal action detectors,” in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 256–272.

Runhao Zeng received the bachelor’s degree in automation science and engineering from the South China University of Technology, Guangzhou, China, in 2015, where he is currently pursuing the Ph.D. degree with the School of Software Engineering. His research interests include machine learning, and deep learning and their applications in video understanding.

Chuang Gan is currently a Researcher with the MIT-IBM Watson AI Lab. His research interest mainly includes multi-modality learning for video understanding.



Peihao Chen received the B.E. degree in automation science and engineering from the South China University of Technology, China, in 2018, where he is currently pursuing the M.E. degree with the School of Software Engineering. His research interests include deep learning in video and audio understanding.

Wenbing Huang received the bachelor’s degree in applied mathematics from Beihang University in 2012 and the Ph.D. degree in computer science and technology from Tsinghua University in 2017. He is currently a Senior Researcher with the Tencent AI Laboratory. He has published over 20 papers in conferences including the Proceedings of NeurIPS, CVPR, ECCV, IJCAI, and AAAI. He has published in over 20 peer-reviewed journals including the IEEE TRANSACTIONS ON FUZZY SYSTEMS. His research interests include computer vision and machine learning.

Qingyao Wu received the Ph.D. degree in computer science from the Harbin Institute of Technology, China, in 2013. He is currently a Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His current research interests include computer vision, natural language processing, and big data learning.

Mingkui Tan received the bachelor’s degree in environmental science and engineering and the master’s degree in control science and engineering from Hunan University, Changsha, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from Nanyang Technological University, Singapore, in 2014. From 2014 to 2016, he was a Senior Research Associate in computer vision with the School of Computer Science, University of Adelaide, Australia. He is currently a Professor with the School of Software Engineering, South China University of Technology. His research interests include machine learning, sparse analysis, deep learning, and large-scale optimization.