# Improving Action Localization by Progressive Cross-stream Cooperation

Rui Su[1] Wanli Ouyang[1,2] Luping Zhou[1] Dong Xu[1]
[1] School of Electrical and Information Engineering, The University of Sydney
[2] SenseTime Computer Vision Research Group, Australia
{rui.su,wanli.ouyang,luping.zhou,dong.xu}@sydney.edu.au

## Abstract

*Spatio-temporal action localization consists of three levels of tasks: spatial localization, action classification, and temporal segmentation. In this work, we propose a new Progressive Cross-stream Cooperation (PCSC) framework to use both region proposals and features from one stream (i.e. Flow/RGB) to help another stream (i.e. RGB/Flow) to iteratively improve action localization results and generate better bounding boxes in an iterative fashion. Specifically, we first generate a larger set of region proposals by combining the latest region proposals from both streams, from which we can readily obtain a larger set of labelled training samples to help learn better action detection models. Second, we also propose a new message passing approach to pass information from one stream to another stream in order to learn better representations, which also leads to better action detection models. As a result, our iterative framework progressively improves action localization results at the frame level. To improve action localization results at the video level, we additionally propose a new strategy to train class-specific actionness detectors for better temporal segmentation, which can be readily learnt by focusing on "confusing" samples from the same action class. Comprehensive experiments on two benchmark datasets UCF-101-24 and J-HMDB demonstrate the effectiveness of our newly proposed approaches for spatio-temporal action localization in realistic scenarios.*

## 1. Introduction

Deep learning has significantly improved performance in various computer vision tasks [6; 16; 15; 14; 26; 13; 30] including human action detection. Human action detection, also known as the spatio-temporal action localization, has attracted increasing research interests due to its wide spectrum of applications in video surveillance (a brief review of the related works is provided in Section 2). It consists of three levels of tasks: i) spatial localization, i.e., finding the spatial locations of persons within each frame, ii) action classification, i.e., identifying the action categories, and iii) temporal segmentation, i.e., determining the beginning and the end of actions. Action detection in videos is a very challenging task due to cluttered background, occlusion and large intra-class variance, etc., especially when targeting the three levels of tasks together.

Our work builds upon the existing observations that appearance and motion information are often complementary to each other in recognizing and localizing human actions at the feature level [21; 20; 17]. In addition, we further observe that the two types of information are also complementary to each other at the region proposal level, so it is beneficial to fully exploit the two types of information at both region proposal and feature levels in order to further improve spatial-temporal action localization results, which is our first motivation.

Specifically, existing region proposals using either appearance or motion clues are not perfect, and they often succeed or fail to detect region proposals in different scenarios. Therefore, they can help each other by providing region proposals to each other. For example, when the motion clue is noisy because of subtle movement or cluttered background, region proposal detectors based on motion information will fail, but region proposal detectors based on appearance information may still successfully find candidate action regions and remove a large amount of background or non-action regions. In another example, it is difficult to detect region proposals based on appearance information when certain type human actions exhibit extreme poses, but human actions could be captured by motion information from human movements. Therefore, we can use the bounding boxes detected from the motion as the region proposals for improving action detection results based on the appearance information, and vice versa.

On the other hand, the detected bounding boxes for actions in individual frames need to be linked in order to form action tubes and temporally segmented out from the entire video clip. Current works [4; 17; 20; 23] for temporal segmentation are mainly based on data association methods that depend on the temporal overlap and smoothness,

as well as action class scores. It is observed that it is often difficult for such methods to precisely identify the temporal boundaries of actions. When the appearance and motion gradually change across temporal boundaries, the frames near boundaries may have only subtle difference. In such a case, it is extremely hard to precisely decide the temporal boundary, and produces a large room for further improvement of the existing temporal refinement methods, which is our second motivation.

Based on the first motivation, in this paper, we propose a progressive framework called Progressive Cross-stream Cooperation (PCSC) to iteratively use both region proposals and features from one stream to progressively help learn better action detection models for another stream. To exploit the information from both streams at the region proposal level, we propose to combine the latest region proposals from both streams in order to collect a larger set of training samples. At the feature level, we propose a new message passing approach to pass information from one stream to another stream in order to learn better representations. As a result, we can progressively learn better action detection models and improve action localization results at the frame-level by leveraging both region proposals and features from one stream to help another stream. Based on the second motivation, we also propose a new temporal segmentation method for training a set of class-specific binary classifiers (also known as actionness detectors) to detect the happening of a certain type of actions. These actionness detectors are trained by focusing on "confusing" samples from the action tube of the same class, and therefore can learn critical features that are good at discriminating the subtle changes across the action boundaries.

Our contributions are briefly summarized as follows:

- We propose the Progressive Cross-stream Cooperation (PCSC) framework to iteratively use both features and region proposals from one stream to help learn better action detection models for another stream, which includes a new message passing approach and a simple region proposal combination strategy.
- We also propose to learn class-specific actionness detectors to improve temporal segmentation results.
- Comprehensive experiments on two benchmark datasets UCF-101-24 and J-HMDB demonstrate that our approach outperforms the state-of-the-art methods for localizing human actions both spatially and temporally in realistic scenarios.

## 2. Related Work

### 2.1. Spatial temporal localization methods

Spatio-temporal action localization involves three types of tasks: spatial localization, action classification, and temporal segmentation. A huge amount of efforts have been dedicated to improve the three tasks from different perspectives. First, for spatial localization, the state-of-the-art human detection methods are utilized to obtain precise object proposals, (including the use of fast and faster R-CNN in [3; 20] as well as Single Shot Multibox Detector (SSD) in [23; 10]).

Second, discriminant features are also employed for both spatial localization and action classification. For example, to remove the ambiguity of actions in each single frame, some methods [10; 3] stack neighbouring frames near one key frame to extract more discriminant features in order to better represent this key frame. Other methods [22; 18; 3] utilize recurrent neural networks to link individual frames or use 3D CNNs [2; 27] to exploit temporal information. Meanwhile, complementary information from multi-modalities is also utilized to improve feature extraction results. For example, a number of works [20; 10; 17; 3] fuse the appearance and motion clues to extract more robust features for action classification.

Third, temporal segmentation is to form action tubes from per-frame detection results. Methods for this task are largely based on the association of per-frame detection results, such as the overlap, continuity and smoothness of objects, as well as the action class scores. To improve segmentation accuracies, a variety of temporal refinement methods have been proposed, e.g., the traditional temporal sliding windows [17], dynamic programming [23; 20], tubelets linking [10], and thresholding-based refinement [3], etc.

Finally, several methods were also proposed to improve action detection efficiency. For example, without requiring time-consuming multi-stage training process, the works in [20; 17] proposed to train a single CNN model by simultaneously performing action classification and bounding box regression. More recently, an online real-time spatio-temporal localization method is also proposed in [23].

### 2.2. Two-stream R-CNN

Based on the observation that appearance and motion clues are often complementary to each other, several state-of-the-art action detection models [20; 10; 17; 3] followed the standard two-stream R-CNN approach. The features extracted from the two streams are fused to improve action detection performance. For example, in [20], the softmax score of each motion bounding box is used to help the appearance bounding boxes with largest overlap. In [3], three types of fusion strategies are discussed: i) simply averaging the softmax outputs of the two streams, ii) learning per-class weights to weigh the original pre-softmax outputs and applying softmax on the weighted sum, and iii) training a fully connected layer on top of the concatenated output from each stream. It is reported in [3] that the third fusion strategy achieves the best performance.

Please note that the appearance and motion fusion approaches in the existing works as discussed above are all based on the late fusion strategy. They are only trained (if there is any training process) on top of the detection networks of the two streams. In contrast, in this work we iteratively use both features and bounding boxes from one stream to progressively help learn better action detection models for another stream, which is intrinsically different with these existing approaches [23; 10] that fuse two-stream information only at the feature level in a late fusion fashion.

# 3. Action Detection Model

Building upon the two-stream framework [17], we propose a Progressive Cross-stream Cooperation (PCSC) model for action detection at the frame level. In this model, the RGB (appearance) stream and the flow (motion) stream iteratively help each other at both features level and region proposal level in order to achieve better localization results. Finally, the action tube refinement module introduced in Section 3.4 is used to link the detection boxes at each frame.

## 3.1. PCSC Overview

The overview of our PCSC model (i.e., the frame-level detection model) is given in Fig. 1. As shown in Fig. 1(a), our PCSC is composed of a set of "stages". Each stage refers to one round of cross-stream cooperation, in which the features and region proposals from one stream will help improve action localization performance for another stream. Specifically, each stage comprises of two cooperation modules and a detection head module. Our **detection head** module is a standard one, which consists of several layers for region classification and regression.

The two cooperation modules include region-proposal-level cooperation and feature-level cooperation, which are introduced in details in Section 3.2 and Section 3.3, respectively. For region-proposal-level cooperation, the detection results from one stream (e.g, the RGB stream) are used as the additional region proposals, which are combined with the region proposals from the other stream (e.g, the flow stream) to refine the region proposals and improve the action localization results. Based on the refined region proposals, we also perform feature-level cooperation by first extracting RGB/flow features from these ROIs and refine these RGB/flow features via a message-passing module shown in Fig. 1(b), and Fig. 1(c), which will be introduced in Section 3.3. The refined ROI features in turn lead to better region classification and regression results in the detection head module, which benefits the subsequent action localization process in the next stage. By performing the aforementioned processes for multiple rounds, we can progressively improve the action detection results. The whole network is trained in an end-to-end fashion by minimizing the overall loss, which is the summation of the losses from all stages.

After performing frame-level action detection, our approach links the per-frame detection results to form action tubes, in which the temporal boundary is further refined by using our proposed class-specific actionness detectors. The details are provides in Section 3.4.

## 3.2. Cross-stream Region Proposal Cooperation

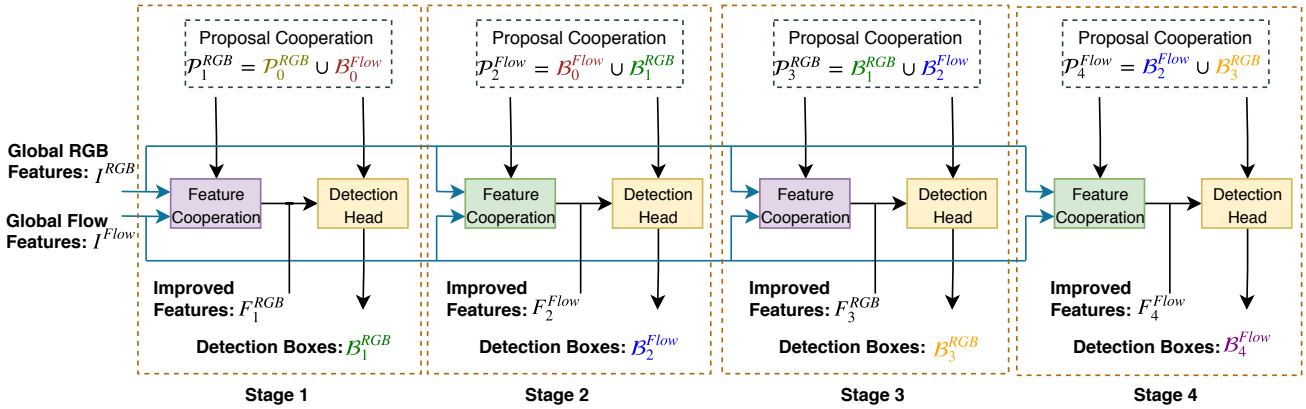
We employ the two-stream Faster R-CNN method [19] for frame-level action localization. Each stream has its own Region Proposal Network (RPN) [19] to generate candidate action regions, and these candidates are then used as training samples to train a bounding box regression network for action localization. Based on our observation, the region proposals generated by either stream can only partially cover the true action regions, which degrades the detection performance. Therefore, in our model, we use the region proposals from one stream to help another stream.

In this paper, the bounding boxes from RPN are called **region proposals**, while the bounding boxes from the detection head are called **detection boxes**.

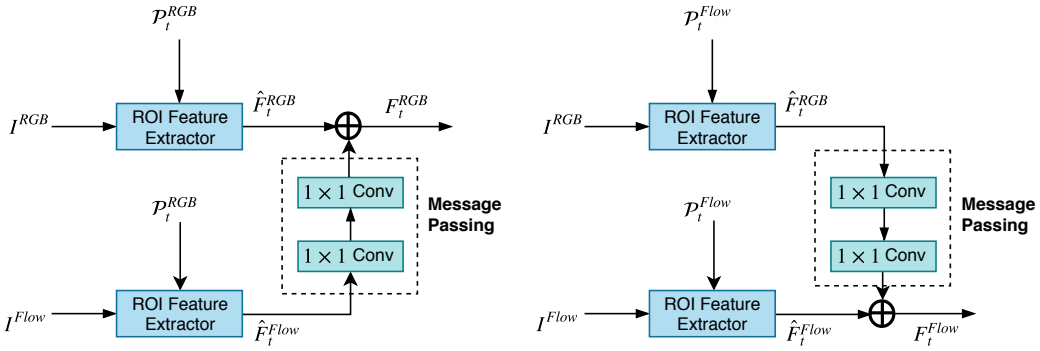
The region proposals from the RPNs of the two streams are first used to train their own detection head separately in order to obtain their own corresponding detection boxes. The set of region proposals for each stream is then refined by combining two subsets. The first subset is from the detection boxes of its own stream (e.g, RGB) at the previous stage. The second subset is from the detection boxes of another stream (e.g, flow) at the current stage. To remove more redundant boxes, a lower NMS threshold is used when the detection boxes in another stream (e.g. flow) is used for the current stream (e.g. RGB).

Mathematically, let $\mathcal{P}^{(i)}$ and $\mathcal{B}^{(i)}$ denote the set of region proposals and the set of the detection boxes, respectively, where $i$ indicates the $i$th stream, $t$ denotes the $t$th stage. The region proposal $\mathcal{P}_t^{(i)}$ is updated as $\mathcal{P}_t^{(i)} = \mathcal{B}_{t-2}^{(i)} \cup \mathcal{B}_{t-1}^{(j)}$, and the detection box $\mathcal{B}_t^{(j)}$ is updated as $\mathcal{B}_t^{(j)} = \mathcal{G}(\mathcal{P}_t^{(j)})$, where $\mathcal{G}(\cdot)$ denotes the mapping function from the detection head module. Initially, when $t = 0$, $\mathcal{P}_0^{(i)}$ is the region proposal from RPN for the $i$th stream. This process is repeated between the two streams for several stages, which will progressively improve the detection performance of both streams.

As can been seen, with our approach, the diversity of region proposals from one stream will be enhanced by using the complementary boxes from another stream. This could help reduce the missing bounding boxes. Moreover, only bounding boxes with high confidence are utilized in our approach, which increases the chances to add more precisely detected bounding boxes and thus further help improve the detection performance. These aforementioned strategies, together with the cross-modality feature cooperation strategy that will be described in Section 3.3, effectively im-




(a)




(b) (c)

Figure 1: (a) Overview of our Cross-stream Cooperation framework (four stages are used as an example for better illustration). The region proposals and features from the flow stream help improve action detection results for the RGB stream at Stage 1 and Stage 3, while the region proposals and features from the RGB stream help improve action detection results for the flow stream at Stage 2 and Stage 4. Each stage comprises of two cooperation modules and the detection head. The region-proposal level cooperation module refines the region proposals $\mathcal{P}_t^i$ and the feature-level cooperation module improves the features $\mathbf{F}_t^i$, where the superscript $i \in \{RGB, Flow\}$ denotes the RGB/Flow stream and the subscript $t$ denotes stage number. The detection head is used for estimating the action location and the class label. For region-proposal level cooperation, we combine the most recent region proposals from the two streams. (b) (c) Details of our feature-level cooperation modules through message passing from one stream to another stream. The flow features are used to help the RGB features in (b), while the RGB features are used to help the flow features in (c).

prove the frame-level action detection results.

Our cross-stream cooperation strategy at the region proposal level shares similar high-level ideas with the two-view learning method co-training [1], as both methods make use of predictions from one stream/view to generate more training samples (i.e., the additional region proposals in our approach) to improve the prediction results for another stream/view. However, our approach is intrinsically different with co-training in the following two aspects. As a semi-supervised learning method, the co-training approach [1] selects unlabelled testing samples and assigns pseudo-labels to those selected samples to enlarge the training set. In contrast, the additional region proposals in our work still come from training videos instead of testing videos, so we know the labels of the new training samples by simply comparing these additional region proposals with the ground-truth bounding boxes. Also, in co-training [1], the learnt classifiers will be directly used for predicting the labels of testing data in the testing stage so the complementary information from the two views for the testing data is not exploited. In

contrast, in our work, the same pipeline used in the training stage will also be adopted for testing samples in the testing stage, and thus we can further exploit the complementary information from the two streams for the testing data.

### 3.3. Cross-stream Feature Cooperation

To extract spatio-temporal features for action detection, similar to [5], we use the I3D network as the backbone network for each stream. Moreover, we follow Feature Pyramid Network (FPN) [12] to build feature pyramid with high-level semantics, which has been found to be useful to improve bounding box proposals and object detection. This involves a bottom-up pathway and a top-down pathway and lateral connections. The bottom-up pathway uses the feed-forward computation along the backbone I3D network, which produces a feature hierarchy with increasing semantic levels but decreasing spatial resolutions. Then these features are upsampled by using the top-down pathway, which are merged with the corresponding features in the bottom-up pathway through lateral connections.

Following [12], we use the feature maps at the layers of Conv2c, Mixed3d, Mixed4f, Mixed5c in I3D to construct the feature hierarchy in the bottom-up pathway, and denote these feature maps as $\{C_2^i,\ C_3^i,\ C_4^i,\ C_5^i\}$, where $i \in \{\text{RGB}, \text{Flow}\}$, indicating the RGB and the flow streams, respectively. Accordingly, the corresponding feature maps in the top-down pathway are denoted as $\{U_2^i, U_3^i, U_4^i, U_5^i\}$.

Most two-stream action detection frameworks [23; 17; 10] only exploit the complementary RGB and flow information by fusing softmax scores or concatenating the features from the final classifiers, which are insufficient for the features from the two streams to exchange information from one stream to another and benefit from such information exchange. Based on this observation, we develop a message-passing module to bridge these two streams, so that they help each other for feature refinement.

We pass the messages between the feature maps in the bottom-up pathway of the two streams. Denote $l$ as the index for the set of feature maps in $\{C_2^i,\ C_3^i,\ C_4^i,\ C_5^i\}$, $l \in \{2, 3, 4, 5\}$. Let us use improvement of the RGB features as an example (the same method is applied to improvement of the flow features). Our message-passing module improves the features $C_l^{\text{RGB}}$ with the help of the features $C_l^{\text{Flow}}$ as follows:

$$C_l^{\text{RGB}} = f_\theta(C_l^{\text{Flow}}) \oplus C_l^{\text{RGB}}. \quad (1)$$

where $\oplus$ denotes the element-wise addition of the feature maps, $f_\theta(\cdot)$ is the mapping function (parameterized by $\theta$) of our message-passing module. The function $f_\theta(C_l^{\text{Flow}})$ nonlinearly extracts the message from the feature $C_l^{\text{Flow}}$, and then use the extracted message for improving the features $C_l^{\text{RGB}}$.

The output of $f_\theta(\cdot)$ has to produce the feature maps with the same number of channels and resolution as $C_l^{\text{RGB}}$ and $C_l^{\text{Flow}}$. To this end, we design our message-passing module by stacking two $1 \times 1$ convolutional layer with relu as the activation function. The first $1 \times 1$ convolutional layers reduces the channel dimension and the second convolutional layer restores the channel dimension back to its original number. This design saves the number of parameters to be learnt in the module and exchange message by using two-layer non-linear transform. Once the feature maps $C_l^{\text{RGB}}$ in the bottom-up pathway are refined, the corresponding features maps $U_l^{\text{RGB}}$ in the top-down pathway are generated accordingly.

The above process is for image-level messaging passing only. The image-level message passing is only performed from the Flow stream to the RGB stream once. This message passing provides good features to initialise the message-passing stages.

Denote the image-level feature map sets for the RGB and flow streams by $I^{\text{RGB}}$ and $I^{\text{Flow}}$ respectively. They are used to extract features $\hat{F}_t^{\text{RGB}}$ and $\hat{F}_t^{\text{Flow}}$ by ROI pooling in each stage $t$, as shown in Fig. 1. At Stage 1 and Stage 3, the ROI-feature $\hat{F}_t^{\text{Flow}}$ of the flow stream is used to help improve the ROI-feature $\hat{F}_t^{\text{RGB}}$ of the RGB stream, as illustrated in Fig. 1 (b). Specifically, the improved RGB feature $F_t^{\text{RGB}}$ is obtained by applying the same method in Eqn. (1). Similarly, at Stage 2 and Stage 4, the ROI-feature $\hat{F}_t^{\text{RGB}}$ of the RGB stream is also used to help improve the ROI-feature $\hat{F}_t^{\text{Flow}}$ of the flow stream, as illustrated in Fig. 1 (c). The message passing between ROI-features aims to provide better features for action box detection and regression, which benefits the next cross-stream cooperation stage.

### 3.4. Action Tube Refinement

After frame-level detection results are generated, we then build action tubes by linking them. Here we use the same linking strategy as in [23], except that we do not apply temporal labeling. Although this linking strategy is robust to missing detection, it is still difficult to accurately determine the start and the end of each action tube, which is a key factor that degrades video-level performance of our action detection framework.

To solve this problem, we develop a class-specific actionness detector to detect the actionness (i.e.the happening of an action) at each given location (spatially and temporally). To simplify the action tube refinement process, the input features for actionness detection are the same features for frame-level action detection. Specifically, we use the features after feature level cooperation (see Section 3.3). Taking advantage of the predicted action class labels from

the frame-level detection results, we construct our actionness detector by using $N$ binary classifiers, where $N$ is the number of action classes. Each classifier addresses the actionness of a specific class. This strategy is more robust than learning a general actionness classifier for all action classes. Specifically, after frame-level detection, each bounding box has a class label, based on which, the bounding boxes from the same class are traced and linked to form action tubes [23]. To train the binary actionness classifier for each action class $i$, the bounding boxes that are within the action tubes predicted as class $i$ by the frame-level detector are used as the training samples. Each bounding box is labeled either as 1 when its overlap with the ground-truth box is greater than 0.5, or as 0 otherwise. Note the training samples may include those bounding boxes falsely detected near a temporal boundary and included into the action tubes of Class $i$. Therefore, they are useful for the actionness classifier to learn the subtle but critical features that determines the begining and the end of this action. The output of the actionness classifier is a probability of actionness of class $i$.

At the testing stage, given an action tube formed using [23], we apply the class-specific actionness detector at every frame-level bounding box in this tube to predict its actionness probablity (called actionness score). Then a median filter over multiple frames is employed to smooth the actionness scores of all bounding boxes in this tube. If a bounding box has a smoothed score lower than a preset threshold, it will be filtered out from this action tube, and then we can refine the action tubes so that they have more accurate temporal boundaries. Note that when a non-action region near a temporal boundary is falsely detected, it is included in the training set to train our class-specific actionness detector. Therefore, our approach takes advantage of the "confusing samples" across temporal boundary to obtain better action tubes at the testing stage.

### 3.5. Training Details

For better spatial localization at the frame-level, we follow [10] to stack neighbouring frames to exploit temporal context and improve action detection performance for key frames. A key frame is a frame containing the ground-truth actions. Each training sample, which is used to train the RPN in our PCSC method, is composed of $k$ neighouring frames with the key frame in the middle. The region proposals generated from the RPN are assigned with positive labels when they have an intersection-over-union (IoU) overlap higher than 0.5 with any ground-truth bounding box, or negative labels if their IoU overlap is lower than 0.5 with all ground-truth boxes. This label assignment strategy also applies to the additional bounding boxes from the assistant stream during the region proposal-level cooperation process.

## 4. Experimental results

We introduce our experimental setup and datasets in Section 4.1, and then compare our method with the state-of-the-art methods in Section 4.2, and conduct ablation study in Section 4.3.

### 4.1. Experimental Setup

**Datasets.** We evaluate our PCSC model on two benchmarks: UCF-101-24 [24] and J-HMDB-21 [9]. **UCF-101-24** contains 3207 untrimmed videos from 24 sports classes, which is a subset of the UCF-101 dataset, with spatio-temporal annotations provided by [23]. Following the common practice, we use the predefined "split 1" protocol to split the training and test sets, and report the results based on this split. **J-HMDB-21** contains 928 videos from 21 action classes. All the videos are trimmed to contain the actions only. We experiment on three predefined training-test splits, and report the averaged results on this dataset.

**Metrics.** We evaluate the action detection performance at both frame-level and video-level by mean Average precision (mAP). To calculate mAP, we consider a detection box is correct when its overlap with a ground-truth box or tube is greater than a threshold $\delta$. The overlap between our detection results and the ground truth is measured by the intersection-over-union (IoU) at the frame-level and the spatio-temporal tube overlap at the video-level. In addition, we also report the results based on COCO evaluation metric [11], which averages the mAPs over 10 different IoU thresholds from 0.5 to 0.95 with an interval of 0.05.

**Implementation Details.** We use the I3D features [2] for both streams, and the I3D model is pretrained with Kinetics. The optical flow images are extracted from FlowNet v2 [8]. The mini-batch size used to train the RPN and the detection head is 256 and 512, respectively. Our PCSC model is trained for 6 epochs by using three 1080Ti GPUs. The initial learning rate is set as 0.01, which drops $10\%$ at the 5th epoch and another $10\%$ at the 6th epoch.

### 4.2. Comparison with the State-of-the-art methods

We compare our PCSC method with the state-of-the-art methods. The results of the existing methods are quoted directly from their original papers. In addition, we also evaluate the object detection model in [12] with the I3D network as its backbone network. Specifically, the model in [12] (denoted as "Faster R-CNN + FPN") and our PCSC only differ in that the PCSC method has the cross-stream cooperation modules while the work in [12] does not have them. Therefore, the comparison between the two approaches can better demonstrate the benefit of our cross-stream cooperation strategy.

#### 4.2.1 Results on the UCF-101-24 Dataset

The results on the UCF-101-24 dataset are reported in Table 1 and Table 2.

Table 1: Comparison (mAPs % at the frame level) of different methods on the UCF-101-24 dataset when using the IoU threshold $\delta$ at 0.5.

| | mAPs |
|---|---|
| Weinzaepfel et al. [29] | 35.8 |
| Peng and Schmid [17] | 65.7 |
| Kalogeiton et al. [10] | 69.5 |
| Gu et al. [5] | 76.3 |
| Faster R-CNN + FPN [12] | 75.5 |
| PCSC (Ours) | **79.2** |

Table 2: Comparison (mAPs % at the video level) of different methods on the UCF-101-24 dataset when using different IoU thresholds.

| IoU threshold $\delta$ | 0.2 | 0.5 | 0.75 | 0.5:0.95 |
|---|---|---|---|---|
| Weinzaepfel et al. [29] | 46.8 | - | - | - |
| Peng and Schmid [17] | 73.5 | 32.1 | 02.7 | 07.3 |
| Saha et al. [20] | 66.6 | 36.4 | 0.79 | 14.4 |
| Singh et al. [23] | 73.5 | 46.3 | 15.0 | 20.4 |
| Kalogeiton et al. [10] | 76.5 | 49.2 | 19.7 | 23.4 |
| Gu et al. [5] | - | 59.9 | - | - |
| Faster R-CNN + FPN [12] | 80.1 | 53.2 | 15.9 | 23.7 |
| PCSC + TR (Ours) | **84.3** | **61.0** | **23.0** | **27.8** |

Table 1 shows the mAPs from different methods at the frame-level on the UCF-101-24 dataset. All mAPs are calculated based on the IoU threshold $\delta = 0.5$. As can be seen, our PCSC model achieves an mAP of 79.2%, outperforming all the existing methods by a large margin. Especially, PCSC performs better than [12] by an improvement of 3.7%. This improvement can be fully due to the proposed cross-stream cooperation framework, which is the only difference between [12] and our PCSC. It is interesting to observe that both methods [10] and [5] additionally utilize the temporal context for per frame-level action detection. They do not fully exploit the complementary of appearance and motion information, and therefore are worse than our method. As can be seen, our method outperforms [10] and [5] by 9.7% and 2.9%, respectively, in terms of frame-level mAPs.

Table 2 reports the video-level mAPs at various IoU thresholds (0.2, 0.5, and 0.75) on the UCF-101-24 dataset. The results based on the COCO evaluation metrics [11] are reported in the last column of Table 2. Our method is denoted as "PCSC + TR" in Table 2, where the proposed temporal refinement (TR) method is applied to refine the action tubes generated from our PCSC model. Consistent with the observations on the frame-level, our method outperforms all the state-of-the-art methods under all evaluation metrics. When using the IoU threshold $\delta = 0.5$, we achieve an mAP of 61.0% on the UCF-101-24 dataset. This result beats [10] and [5], which only achieve the mAPs of 49.2% and 59.9%, respectively. Moreover, as the IoU threshold increases, we observe that the performance of our method drops less when compared with other state-of-the-art methods. This demonstrates that our detection method achieves higher localization accuracy than other competitive methods.

#### 4.2.2 Results on the J-HMDB Dataset

For the J-HMDB dataset, the results in terms of frame-level mAPs and video-level mAPs are reported in Table 3 and Table 4, respectively. Since the videos in J-HMDB dataset are trimmed to only contain actions, the temporal refinement process is not required, so we do not apply our TR method when generating action tubes on the J-HMDB dataset.

Table 3: Comparison (mAPs % at the frame level) of different methods on the J-HMDB dataset when using the IoU threshold $\delta$ at 0.5.

| | mAPs |
|---|---|
| Peng and Schmid [17] | 58.5 |
| Kalogeiton et al. [10] | 65.7 |
| Hou et al. [7] | 61.3 |
| Gu et al. [5] | 73.3 |
| Sun et al. [25] | **77.9** |
| Faster R-CNN + FPN [12] | 70.2 |
| PCSC (Ours) | 74.8 |

Table 4: Comparison (mAPs % at the video level) of different methods on the J-HMDB dataset when using different IoU thresholds.

| IoU threshold $\delta$ | 0.2 | 0.5 | 0.75 | 0.5:0.95 |
|---|---|---|---|---|
| Gkioxari and Malik [4] | - | 53.3 | - | - |
| Wang et al. [28] | - | 56.4 | - | - |
| Weinzaepfel et al. [29] | 63.1 | 60.7 | - | - |
| Saha et al. [20] | 72.6 | 71.5 | 43.3 | 40.0 |
| Peng and Schmid [17] | 74.1 | 73.1 | - | - |
| Singh et al. [23] | 73.8 | 72.0 | 44.5 | 41.6 |
| Kalogeiton et al. [10] | 74.2 | 73.7 | 52.1 | 44.8 |
| Hou et al. [7] | 78.4 | 76.9 | - | - |
| Gu et al. [5] | - | 78.6 | - | - |
| Sun et al. [25] | - | 80.1 | - | - |
| Faster R-CNN + FPN [12] | 79.1 | 78.5 | 57.2 | 47.6 |
| PCSC (Ours) | **82.6** | **82.2** | **63.1** | **52.8** |

We have similar observation as in the UCF-101-24

dataset. At the video-level, our method is again the best performer under all evaluation metrics on the J-HMDB dataset (see Table 4). When using the IoU threshold $\delta = 0.5$, our PCSC method outperforms [5] and [25] by 3.6% and 2.1%, respectively.

At the frame-level (see Table 3), our PCSC method performs the second best, which is only worse than a very recent work [25]. However, the work in [25] uses S3D-G as the backbone network, which provides much stronger features when compared with the I3D features used in our method. In addition, please note that, our method outperforms [25] in terms of mAPs at the video level (see Table 4), which demonstrates promising performance of our PCSC method. Moreover, as a general framework, our PCSC method could also take advantage of strong features provided by the S3D-G model to further improve the results, which will be explored in our future work.

### 4.3. Ablation Study

To investigate the contributions of different components in our PCSC model, we construct a mini-UCF-101-24 dataset by sampling every 10 frames from the videos in UCF-101-24, which is only used as the training data in our ablation study. The test data of UCF-101-24 for evaluation in all experiments are kept unchanged.

Table 5: Ablation study for our PCSC method at different training stages on the mini-UCF-101-24 dataset.

| Stage | PCSC w/o feature cooperation | PCSC |
|---|---|---|
| 0 | 72.7 | **75.7** |
| 1 | 73.3 | **76.1** |
| 2 | 73.5 | **76.4** |
| 3 | 73.9 | **76.6** |
| 4 | 73.9 | **76.7** |

Table 6: Ablation study for our temporal refinement method on the mini-UCF-101-24 dataset.

| | video mAP |
|---|---|
| Faster R-CNN + FPN [12] | 53.2 |
| PCSC | 55.8 |
| PCSC + TR | **59.4** |

**Progressive cross-stream cooperation.** In Table 5, we report the results of an alternative approach of our PCSC (called PCSC w/o feature cooperation approach). In the second column, we remove the feature-level cooperation module from Fig. 1, and only use the region-proposal-level cooperation module. As our PCSC is conducted in a progressive manner, we also report the performance at different stages to verify the benefit of this progressive strategy. It is worth mentioning that the output at each stage is obtained by combining the detected bounding boxes from both the current stage and all previous stages, and then we apply non-maximum suppression (NMS) to obtain the final bounding boxes. For example, the output at stage 4 is obtained by applying NMS to the union set of the detected bounding boxes from Stages 0, 1, 2, 3 and 4. At Stage 0, the detection results from both RGB and the Flow streams are simply combined. From Table 5, we observe that the detection performance of our PCSC method with or without the feature-level cooperation module is improved as the number of stages increases. However, such improvement seems to become saturated when reaching Stage 4, as indicated by the marginal performance gain from Stage 3 to Stage 4. Meanwhile, when comparing our PCSC with the alternative approach PCSC w/o feature cooperation at every stage, we observe that both region-proposal-level and feature-level cooperation contributes to performance improvement.

**Action tubes refinement.** In Table 6, we also investigate the effectiveness of our temporal refinement method by reporting the results with/without the temporal refinement module, in which video-level mAPs at the IoU threshold $\delta = 0.5$ are reported. By generating higher quality detection results at each frame, our PCSC method outperforms the work in [12] that does not use the two-stream cooperation strategy by 2.6%. After applying our action tube refinement method to further boost the video-level performance, we arrive at the video-level mAP of 59.4%, which demonstrates that it is beneficial to use our action tube refinement method to refine the temporal boundaries of action tubes.

## 5. Conclusion

In this work, we have proposed the Progressive Cross-stream Cooperation (PCSC) framework to progressively improve spatio-temporal action localization results at the frame level, which consists of several iterative stages. At each stage, we improve action localization results for one stream (i.e., RGB/flow) by the leveraging the information from another stream (flow/RGB) at both region proposal level and feature level. We additionally propose a simple but effective approach to improve temporal segmentation results by training class-specific actionness detectors based on the training samples around temporal boundaries. The effectiveness of our newly proposed approaches is demonstrated by extensive experiments on both UCF-101-24 and J-HMDB datasets.

## References

[1] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM. ISBN 1-

58113-057-0. doi: 10.1145/279943.279962. URL http://doi.acm.org/10.1145/279943.279962. 4

[2] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2, 6

[3] G. Chéron, A. Osokin, I. Laptev, and C. Schmid. Modeling spatio-temporal human track structure for action localization. *CoRR*, abs/1806.11008, 2018. URL http://arxiv.org/abs/1806.11008. 2

[4] G. Gkioxari and J. Malik. Finding action tubes. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015. 1, 7

[5] C. Gu, C. Sun, D. A. Ross, C. Vondrick, C. Pantofaru, Y. Li, S. Vijayanarasimhan, G. Toderici, S. Ricco, R. Sukthankar, C. Schmid, and J. Malik. Ava: A video dataset of spatio-temporally localized atomic visual actions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 5, 7, 8

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1

[7] R. Hou, C. Chen, and M. Shah. Tube convolutional neural network (t-cnn) for action detection in videos. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 7

[8] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *IEEE conference on computer vision and pattern recognition (CVPR)*, volume 2, page 6, 2017. 6

[9] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 3192–3199, 2013. 6

[10] V. Kalogeiton, P. Weinzaepfel, V. Ferrari, and C. Schmid. Action tubelet detector for spatio-temporal action localization. In *ICCV*, 2017. 2, 3, 5, 6, 7

[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6, 7

[12] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 5, 6, 7, 8

[13] L. Liu, H. Wang, G. Li, W. Ouyang, and L. Lin. Crowd counting using deep recurrent spatial-aware network. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 849–855. AAAI Press, 2018. 1

[14] W. Ouyang and X. Wang. Single-pedestrian detection aided by multi-pedestrian detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3198–3205, 2013. 1

[15] W. Ouyang, K. Wang, X. Zhu, and X. Wang. Chained cascade network for object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1938–1946, 2017. 1

[16] W. Ouyang, X. Zeng, X. Wang, S. Qiu, P. Luo, Y. Tian, H. Li, S. Yang, Z. Wang, H. Li, et al. Deepid-net: Object detection with deformable part based convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1320–1334, 2017. 1

[17] X. Peng and C. Schmid. Multi-region two-stream r-cnn for action detection. In *European Conference on Computer Vision*, pages 744–759. Springer, 2016. 1, 2, 3, 5, 7

[18] L. Pigou, A. van den Oord, S. Dieleman, M. V. Herreweghe, and J. Dambre. Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video. *International Journal of Computer Vision*, 126(2-4):430–439, 2018. 2

[19] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015. 3

[20] S. Saha, G. Singh, M. Sapienza, P. H. S. Torr, and F. Cuzzolin. Deep learning for detecting multiple space-time action tubes in videos. In *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*, 2016. 1, 2, 7

[21] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems 27*, pages 568–576, 2014. 1

[22] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao. A multi-stream bi-directional recurrent neural network for fine-grained action detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 2

[23] G. Singh, S. Saha, M. Sapienza, P. Torr, and F. Cuzzolin. Online real time multiple spatiotemporal action localisation and prediction. 2017. 1, 2, 3, 5, 6, 7

[24] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. 2012. 6

[25] C. Sun, A. Shrivastava, C. Vondrick, K. Murphy, R. Sukthankar, and C. Schmid. Actor-centric relation network. In *The European Conference on Computer Vision (ECCV)*, September 2018. 7, 8

[26] S. Sun, J. Pang, J. Shi, S. Yi, and W. Ouyang. Fishnet: A versatile backbone for image, region, and pixel level prediction. In *Advances in Neural Information Processing Systems*, pages 762–772, 2018. 1

[27] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, 2015. 2

[28] L. Wang, Y. Qiao, X. Tang, and L. Van Gool. Actionness estimation using hybrid fully convolutional networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016. 7

[29] P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Learning to track for spatio-temporal action localization. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015. 7

[30] W. Zhang, W. Ouyang, W. Li, and D. Xu. Collaborative and adversarial network for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3801–3809, 2018. 1