Movement Enhancement toward Multi-Scale Video Feature Representation for Temporal Action Detection

Zixuan Zhao Dongqi Wang Xu Zhao*
Department of Automation, Shanghai Jiao Tong University, China
{zhaozixuan, wangdq0124, zhaoxu}@sjtu.edu.cn

Abstract

Boundary localization is a challenging problem in Temporal Action Detection (TAD), in which there are two main issues. First, the submergence of movement feature, i.e. the movement information in a snippet is covered by the scene information. Second, the scale of action, that is, the proportion of action segments in the entire video, is considerably variable. In this work, we first design a Movement Enhance Module (MEM) to highlight movement feature for better action location, and then, we propose a Scale Feature Pyramid Network (SFPN) to detect multi-scale actions in videos. For Movement Enhance Module, firstly, Movement Feature Extractor (MFE) is designed to get the movement feature. Secondly, we propose a Multi-Relation Enhance Module (MREM) to grasp valuable information correlation both locally and temporally. For Scale Feature Pyramid Network, we design a U-Shape Module to model different scale actions, moreover, we design the training and inference strategy of different scales, ensuring that each pyramid layer is only responsible for actions at a specific scale. These two innovations are integrated as the Movement Enhance Network (MENet), and extensive experiments conducted on two challenging benchmarks demonstrate its effectiveness. MENet outperforms other representative TAD methods on ActivityNet-1.3 and THUMOS-14.

1. Introduction

Temporal Action Detection (TAD) is a significant video understanding task, which aims to extract video segments with specific action labels from untrimmed long videos. It remains a challenging problem because action boundaries are difficult to determine precisely, which adversely affects the performance in most works.

Two critical issues stand in the way of detecting action boundaries robustly from untrimmed videos. The first issue, which could be called as movement feature submergence, exists in where either context but not movement itself dominate feature expression, or movement is small in pixel size.

*Corresponding author

Figure 1. Movement feature submergence and multi-scale actions. (a) Strong class-specific context. (b) Small movement in pixel size. (c) Multi-scale actions.

For the first case, as shown in Fig 1(a), the action of Playing Accordion can be easily judged by Accordion, while this strong class-specific context causes the real movement information in action area to be submerged. For the second case, as shown in Fig 1(b), for Futsal, the stadium scene takes up a large proportion of pixels in the frame, while the movement information is ignored. This movement feature submergence leads to the feature in background to be similar with the feature in action area, which blues the action boundaries. If the movement information can be highlighted, the boundary will be easier to be located. However, this problem has been ignored by prior works [12, 29, 22].

The second issue is the multi-scale of actions in an
Temporal Action Detection (TAD) task. Its contributions are summarized as follows. Firstly, compared with small-scale action, large-scale action contains more obviously action process (i.e., start phase, action phase and end phase). Some researches [13, 7, 11] resort feature pyramid network (FPN) to solve the problem. However, there are two aspects that are ignored. Firstly, these works detect actions at different feature scales, but, they do not adaptively learn action features of different scales. Secondly, the information flow in FPN is insufficient, lacking interactions between different pyramid layers.

In this paper, as an effort to overcome the above challenges, we investigate the feature of TAD. (1) In order to overcome the movement feature submergence, and highlight the difference between foreground and background snippets. We design the Movement Enhance Module (MEM) to establish the movement information in a video snippet, which includes two crucial parts. First, we propose a siamese network named Movement Feature Extractor (MFE), which uses the dynamic information of the frame sequence and the static information of the frame to extract movement feature from video snippets. And then, we propose the Multi-Relation Enhance Module (MREM) to establish temporal and local correlations between video snippets. (2) In order to learn specific representations for different scale actions, we propose the Scale FPN (SFPN), which uses a U-shaped network to produce multi-scale video features and facilitate the information flow between different layers. Moreover, for each layer in SFPN to take charge of detecting action segments of the corresponding scale, we design a two-stage learning strategy. In the former Generalization stage, each layer is trained with all action segments; in the latter Specialization stage, each layer is biased toward actions in a specific scale range. During inference, every layer takes charge of detecting action segments of the corresponding scale.

In summary, this work explores feature representation and action segment representation that are more suitable for TAD task. Its contributions are summarized as follows.

1. To alleviate the movement feature submergence, we design the Movement Enhance Module (MEM) to highlight the movement feature in a video snippet, and explore local and temporal relations between snippets.

2. For the multi-scale actions in an untrimmed video with different feature patterns, we design the Scale FPN (SFPN) to learn different scale actions respectively, where targeted training and inference strategies are adopted. Consequently, each layer in the SFPN specializes in actions at a certain scale range.

3. Extensive experiments conducted on two datasets verify the effectiveness of our proposed method. On ActivityNet1.3, MENet promotes the best average mAP from 36.6% to 37.7%, and boosts the mAP@0.7 from 31.8% to 34.0% on THUMOS-14.

2. Related Work

Video Feature for TAD. The pre-trained TAC model is often used as the feature extractor for Temporal Action Detection which receives several frames as a snippet to distill both appearance and motion information from raw video frames. TSN [26], I3D [3], and SlowFast [6] are the most common feature extractors. However, they only focus on the action categories, which results in some actions that are strongly related to the scene having quite similar representations to background segments. To overcome the drawback, TSP [1] adds additional background supervision in the pre-training model to generate the background-sensitive feature. BSP [28] artificially synthesizes different video boundaries and conducts boundary learning in the pretraining model to generate boundary-sensitive feature. However, these works cannot address the problem of movement feature submergence fundamentally. In this paper, MENet uses the Movement Enhance Module (MEM) to magnify the movement feature and the difference between action and background.

Multi-scale Action Detection. The scale of the action varies dramatically in a video. Compared with the large actions, those small action segments have fewer samples and insufficient features. Some works [13, 7, 11] focus on feature pyramid network (FPN) to generate different temporal resolution features. However, the FPN structure does not learn specific feature representation for a certain action scale, and also lacks sufficient information flow intrinsically. TSI [15] proposes a scale-invariant loss, which balances large and small actions in quantity, but fails to fundamentally solve the scarce feature of small actions. In this work, we propose SFPN to cope with these problems, which establishes associations between different scale features and learns scale-specific feature representations.

Temporal Action Detection. Benefiting from the successful practice of image object detection and trimmed video action recognition, the two-stage pipeline prevails in TAD task. Specifically, the first stage is to locate candidate action segments in the video, and the second stage uses an action classifier to classify these proposals. The first stage has three main paradigms. (1) Anchor-Based method: Methods [5, 13, 21] put anchors of various scales on the video feature, and then give the final confidence of anchors. However, they cannot produce flexible boundaries. (2) Boundary-Base method: Methods [14, 31, 16]...
predict boundary scores on the video feature, which are combined to generate proposals. However, the confidence of the proposal lacks global information. (3) Combined method: Methods [12, 10, 4] combine these two methods to generate precise confidence and flexible boundaries. In this work, we follow the combined method to integrate the start point and end point as a proposal and generate multi-scale anchor maps to predict anchor confidence.

3. Method

3.1. Overview

**Problem Definition.** Input of the Movement Enhance Network (MENet) is an untrimmed video denoted as \( V \). The Problem Definition is to predict boundary scores on the video feature, which are combined to generate proposals. However, the confidence of the proposal lacks global information.

**Input of the Movement Enhance Network (MENet).** The input of MENet is an untrimmed video denoted as \( V = \{ v_i \}_{i=1}^{L_v} \), where \( v_i \) represents the \( i \)-th frame of the video and \( L_v \) is the total frames. The duration of the video is \( T_v \). Due to redundancy by video frames, several consecutive frames are usually regarded as a snippet. With the sampling interval \( \sigma \), the whole video can be defined as snippet sequence \( S = \{ v_i \}_{i=1}^{L_s} \), where \( L_s = L_v / \sigma \) representing the number of total snippets. The output of the MENet is \( \{ \psi_i \} = \{ t_{i,s}, t_{i,e}, c_i, \text{score}_i \} \) where \( t_{i,s} \) and \( t_{i,e} \) are start time and end time, action category and confidence score, respectively. The annotations of untrimmed video are action instances \( \Psi_i = (t_{i,s}, t_{i,e}, c_i) \).

**Framework Architecture.** The framework architecture of MENet is shown in Fig 2. MENet is mainly composed of two parts: Movement Enhance Module (MEM) and Scale FPN (SFPN). MEM is used to extract movement enhanced feature and explore different video relations between snippets. SFPN is proposed to generate feature pyramid and learn specific expression patterns of different action scales. Finally, SFPN gives action boundaries and confidence scores of different scale actions, separately.

3.2. Movement Enhance Module

To obtain the feature that contains action category information and is sensitive to foreground and background simultaneously, we propose the Movement Enhance Module, which contains two vital stages: Movement Feature Extractor (MFE) and Multi-Relation Enhance Module (MREM).

**Movement Feature Extractor.** In order to amplify the movement information, MFE is proposed. As shown in Fig 3, MFE uses a siamese network to highlight the movement information in a video snippet. We select R(2+1)D-34 [24] as backbone with the TAC pretrained weight on Kinetics-400 [9], and two networks share the weight. MFE has two input paths. One path is video snippet path composed of consecutive frames, and the other is frame duplication path, which selects a frame from the snippet and copies it to the length of the snippet. Here, the middle frame is selected for duplication. The feature from the video snippet path contains both static scene and dynamic movement information, which is \( F_{Total} \in \mathbb{R}^{C \times T} \), but feature from the frame duplication path only contains static scene information, which is \( F_{Static} \in \mathbb{R}^{C \times T} \). And then, MFE extracts the movement information in the snippet using the difference between the two features, which is defined as \( F_{Movement} \in \mathbb{R}^{C \times T} \). Finally, \( F_{Movement} \) and \( F_{Total} \) are concatenated as \( F_{MT} \in \mathbb{R}^{2C \times T} \).

**Multi-Relation Enhance Module.** To establish relations between snippets and fuse the information at semantic dimension, the Multi-Relation Enhance Module is proposed. As shown in Fig 2, firstly, we use a 1D convolution to fuse information at semantic dimension. Secondly, there are three information paths in MREM. (1) Local Path: 1D convolution can directly establish relations in a local scale \( k \), where \( k \) is the kernel size. However, it is difficult for 1D convolutions to establish temporal relationship between snippet features. (2) Forward Path: The LSTM processes each snippet successively according to the input order, with an outstanding ability to model the temporal relation. In forward path, we use LSTM to establish forward information. (3) Backward Path: In backward path, we flip the video sequence, and establish backward information. Finally, the three paths are integrated.

3.3. Scale Feature Pyramid Network

We contend different scale actions have different patterns. Therefor, the SFPN is designed to learn patterns of specific scales. SFPN contains U-Shape Module and Detector Head, which aims to generate feature pyramid and final detection results, respectively.

**U-Shape Module.** For feature pyramid generating, as shown in Fig 2, SFPN uses a U-shaped architecture inspired by [20] to generate feature sequences in different temporal resolution. From top to down, SFPN uses multiple temporal convolution layers followed by AvgPooling to downsample the temporal resolution. From bottom to up, to restore the lower layer information, SFPN uses several 1D transpose convolution to restore the feature resolution and features in the same resolution are concatenated.

**Detector Head.** As demonstrated in Fig 2, each detector head consists of a Boundary Predictor, an Anchor Feature Align Layer and two Anchor Score Predictor (ASP). The output of the Boundary Predictor and Anchor Score Predictor is used to generate proposals and confidence scores, and the Anchor Feature Align Layer is to generate the feature representation for each anchor.

**Boundary Predictor.** Boundary Predictor aims at predicting boundary probabilities \( \bar{P}_s = \{ p_{s,i} \}_{i=1}^{L_s}, P_e = \{ p_{e,i} \}_{i=1}^{L_s} \) of each snippet by recognizing the boundary pattern.

**Anchor Feature Align Layer.** The area between any two snippets can form an anchor, and each anchor consists of several consecutive snippets. Supposing \( N_i \) anchors are randomly sampled to participate in the forward. The Anchor Feature Align Layer adopts SGAlign designed in [29]
Figure 2. Overview of our MENet and the structure of some sub-modules. There are two crucial parts in the framework (Top) of MENet: Movement Enhance Module (MEM) and Scale FPN (SFPN). MEM is designed to generate movement enhanced feature. SFPN is designed to generate multi-scale feature pyramid, and learn action expressions in different layers. The structure of the sub-modules is shown below (Bottom) with the same color in the framework.

Figure 3. Illustration of MFE. This module constructs the movement feature in a snippet by the difference between dynamic video snippet and static picture.

**Anchor Score Predictor.** IoU-ASP and Offset-ASP have the same structure to recognize the action pattern within an anchor. For the $i$-th layer in SFPN, IoU-ASP generates two maps $M_{d_{i}}^{c_{i}} \in \mathbb{R}^{D_{i} \times L_{i}}$, $M_{r_{i}}^{c_{i}} \in \mathbb{R}^{D_{i} \times L_{i}}$, where $D_{i}$ is predefined maximum anchor duration. In Offset-ASP, two output maps are denoted as $M_{e_{i}}^{c_{i}} \in \mathbb{R}^{D_{i} \times L_{i}}$, $M_{d_{i}}^{c_{i}} \in \mathbb{R}^{D_{i} \times L_{i}}$. These two maps indicate each anchor’s center offset and duration offset respectively.

### 3.4. Specific action pattern Learning

Actions of different scales have different feature representations. Correspondingly, different layers in SFPN have different feature granularity and temporal resolution. Therefore, we advocate applying a targeted training strategy for actions at different scales, assuring that each layer in SFPN is only responsible for actions at a specific scale range. Specifically, define the scale $S \in [0, 1]$ as the ratio of action length to video length. Then the first layer (the top layer of the pyramid) with the longest sequence is responsible for small-scale actions whose scale in $[S_{\text{min}}^{1}, S_{\text{max}}^{1}]$, and the second layer (the middle layer of the pyramid) with a moderate sequence length is responsible for medium-scale actions whose scale in $[S_{\text{min}}^{2}, S_{\text{max}}^{2}]$, the remaining third layer with the shortest sequence is responsible for large-scale actions whose scale belongs to $[S_{\text{min}}^{3}, S_{\text{max}}^{3}]$. To implement this idea, a two-stage training strategy is designed.

**Two-stage training strategy.** The two-stages of training ASP are Generalization and Specialization, respectively. Due to the different lengths of feature sequences, the to-
Figure 4. The two-stage training strategy (left column and middle column) and the inference strategy (right column) of ASP. Each square represents an anchor map of a layer, the grey area is the invalid region. Other colors represent different sampling ratios in a certain scale range, with white, green, yellow and red representing 0%, r1, 2r1, and 100% respectively.

Inference

The number of dense anchors contained in each layer varies. In order to ensure that each layer is trained equally and reduce the computational cost, different anchor sampling ratios r1, r2, r3 are set for the three layers, where 0% < r1 ≤ r2 ≤ r3 ≤ 100%. (1) In the former Generalization stage, given the sampling ratio ri, randomly sample N_i = (r_i/2) · r_i anchors from valid region to train the IoU-ASP and Offset-ASP of i-th layer, as shown in the left column of Fig 4. The purpose of this stage is to let ASP learn a general pattern of all actions. (2) In the latter Specialization stage, for the i-th layer in SFPN, the training samples come from two parts, as shown in the middle column in Fig 4. The first part are randomly sampled according to the sampling rate of min(100%, 2 - r_i) from anchors whose scale in [S_{min}, S_{max}] (yellow area in Fig 4). The second part are sampled from the remaining anchors whose scale in (0, S_{min}) or (S_{max}, 100%), with the sampling rate of r_i (green area in Fig 4). This Specialization stage highlights the effect of anchors at the certain scale range, making the detection head of each layer more specialized.

3.5. Training & Inference

Label Assignment. Assuming the video duration is L_v, each snippet corresponds to a video period. In annotation, a ground-truth action which starts at t_s^* and ends at t_e^* is expanded from moment to region, the start region is defined as R_s = [t_s^* - 1.5 T_v, t_s^* + 1.5 T_v] and end region is defined as R_e = [t_e^* - 1.5 T_v, t_e^* + 1.5 T_v]. For Boundary Predictor, compute the overlap between each snippet period and R_s as the label of start probabilities G_s ∈ R^{L_v}. Similarly, compute the overlap between snippet period and R_e as the label of end probabilities G_e. For IoU-ASP, following [12], IoU between each anchor and all actions are calculated and then arranged into a map G_{i\text{IoU}} ∈ R^{D_i × L_v}. Anchors with IoU score greater than 0.9 participate in the training of Offset-ASP. For anchor start at t_s and end at t_e, assuming its corresponding ground-truth action is [t_s^*, t_e^*], the center offset ∆c and duration offset ∆d are calculated as Eq. (1)-(2), and arranged into maps G_{i\text{cent}} ∈ R^{D_i × L_v} and G_{i\text{dura}} ∈ R^{D_i × L_v}.

\[
c = \frac{t_s + t_e}{2}, d = t_e - t_s, c^* = \frac{t_s^* + t_e^*}{2}, d^* = t_e^* - t_s^*
\]

Basic Loss. (1) For the i-th layer, the loss of Boundary-Predictor is the re-weighted binary cross-entropy loss \( L_B \) Eq. 3, where \( P = \{p_i\}_{i=1}^{N_i} \) and \( G = \{g_i\}_{i=1}^{N_i} \) represent the predicted boundary probabilities and its label of all snippets, respectively. Snippets with \( g_i > 0.5 \) serve as positive samples (i.e., \( \delta\{g_i > 0.5\} = 1 \)) and others are negative samples. \( T^i_s \) and \( T^i_e \) are the number of positive and negative samples in this layer respectively. (2) The loss of IoU-ASP is the re-weighted binary cross-entropy loss \( L_{\text{asp}} \) Eq. 4 and L2 loss \( L_2 \), where \( M = \{m_j\}_{j=1}^{N_j} \) and \( G = \{g_j\}_{j=1}^{N_j} \) represent the predicted value and ground-truth value of each sampled anchor in the i-th layer. \( N_i, N_i^+, N_i^- \) represent the number of sampled anchors, positive anchors whose IoU greater than 0.9 and negative anchors of this layer, respectively. (3) The loss of Offset-ASP is Smooth L1 loss \( L_1 \). Note that only those sampled anchors with IoU greater than 0.9 participate in the training of Offset-ASP. The re-weights in \( L_B \) and \( L_{\text{asp}} \) are used to balance the number between positive and negative samples.

\[
L_B(P, G) = -\frac{1}{L_i} \sum_{i=1}^{L_i} \left( \frac{L_i}{T_i} \right) \cdot \delta\{g_i > 0.5\} \cdot \log p_i + \frac{L_i}{T_i} \cdot (1 - \delta\{g_i > 0.5\}) \cdot \log(1 - p_i)
\]

\[
L_{\text{asp}}(M, G) = -\frac{1}{N_j} \sum_{j=1}^{N_j} \left( \frac{N_j}{N_i} \right) \cdot \delta\{g_j > 0.9\} \cdot \log m_j + \frac{N_j}{N_i} \cdot (1 - \delta\{g_j > 0.9\}) \cdot \log(1 - m_j)
\]

Total Loss. The loss of the i-th layer is composed of three parts: boundary loss, IoU loss and offset loss, as shown in Eq.(5) (6) (7) respectively. The total training objective is the sum of all three layers, formulated as Eq. 8. Besides, in order to balance the value between different terms, the coefficient \( \lambda_1 \) and \( \lambda_2 \) are set as 5 and 10.

\[
L_{\text{bound}} = L_B(P_s^i, G_s^i) + L_B(P_e^i, G_e^i)
\]

\[
L_{\text{IoU}} = L_{\text{asp}}(M_{clst}^i, G_{clst}^i) + \lambda_1 \cdot L_2(M_{reg}^i, G_{reg}^i)
\]

\[
L_{\text{off}} = L_1(M_{cent}^i, G_{cent}^i) + L_1(M_{dura}^i, G_{dura}^i)
\]
\[ L_{\text{total}} = \sum_{i=1}^{3} (L_B^i + L_{\text{IoU}}^i + \lambda_2 \cdot L_{\text{aff}}^i) \]  

(8)

**Inference.** Each layer outputs \( p_s^i, p_e^i, M_{dx}^i, M_{reg}^i, M_{cent}^i \) and \( M_{\text{dura}} \) from the detector head in inference. Following [14, 12, 15], snippet in boundary probability \( P_s \) is screened out as candidate start point if it is local peak or its probability is greater than \( 0.5 \cdot \max(P_s) \). And snippets can be selected as candidate end point from \( P_e \) in the same way. Then the candidate start and end points are combined into proposals. In order to produce the more reasonable proposals, as shown in Fig 4, each layer of SFPN only outputs anchors in the corresponding scale range. Specifically, for the \( i \)-th layer, its responsible scale range is \([S_{\text{min}}^i, S_{\text{max}}^i]\). Furthermore, in the proposals set, for any start snippet whose index is \( sdx \) and centered at time \( t_1 \), and the end snippet whose index is \( edx \) and centered at time \( t_2 \), we can get its start probability \( p_s = P_s[\text{sdx}] \), end probability \( p_e = P_e[\text{edx}] \) and tIoU score \( p_{\text{IoU}} = M_{\text{cls}}[\text{edx} - \text{sdx}, \text{sdx}] \cdot M_{\text{reg}}[\text{edx} - \text{sdx}, \text{sdx}] \). Its center is \( c = (\text{sdx} + \text{edx})/2L_i \) and duration is \( d = (\text{edx} - \text{sdx})/L_i \). Subsequently, to refine the boundary, we can obtain center offset \( \Delta c = M_{\text{cent}}[\text{edx} - \text{sdx}, \text{sdx}] \) and duration offset \( \Delta d = M_{\text{dura}}[\text{edx} - \text{sdx}, \text{sdx}] \), and adjust the boundary to \((t_1', t_2')\) as Eq. 9:

\[
\begin{align*}
    c' &= c - \Delta c + c, \\
    d' &= d + \Delta d \\
    t_1 &= c' - \frac{d'}{2}, \\
    t_2 &= c' + \frac{d'}{2}
\end{align*}
\]  

(9)

Finally, proposals generated by each layer are merged together, and the action classifier assigns every proposal with a certain label and a classification score \( P_{\text{label}} \). The final score for the proposal \((t_1', t_2')\) is shown as Eq. 10. Then Soft-NMS [32] is adopted to remove redundant segments.

\[
\text{score}_{t_1', t_2'} = \frac{p_s \cdot p_e \cdot p_{\text{IoU}} \cdot P_{\text{label}}}{\sum \text{score}_{t_1', t_2'}}
\]  

(10)

**4. Experiments**

**4.1. Dataset and Settings**

**Dataset.** In order to verify the effectiveness of MENet, we experiment on two challenging datasets. ActivityNet-1.3 [2] contains 200 categories of daily life, sports, etc. We use the training set for model training and report the performance on the validation set. THUMOS-14 [8] consists of 413 videos with 20 action classes which is almost sports. In THUMOS-14, the validation set contains 200 long videos, including 3007 action segments, and the test set contains 213 videos, including 3358 action segments. Following the standard practice of THUMOS-14, we train MENet on the validation set and valid it on the test set.

**Metric.** In order to fully demonstrate the advantages of our proposed method, we use two main evaluation metrics: mAP under a certain temporal tIoU threshold (mAP@tIoU) and average recall at a specified average number of proposals (AR@AN). Besides, on ActivityNet, the area under the AR-AN curve (AUC) also serves as a metric.

**Network Details.** We use R(2+1)D-34 as backbone with the TAC pretrained weight on Kinetics-400 serving as the primary feature extractor. The sampling interval \( \sigma \) set as 16 frames and 5 frames for ActivityNet-1.3 and THUMOS-14, respectively. In this way, for the input video with \( L_v \) frames, video snippet length \( L_s = L_v/\sigma \). For ActivityNet-1.3, we use linear interpolation to resize the length of video features \( T = 200 \). For a long video in THUMOS-14, a sliding window with a length of 256 and an overlap of 50\% is used to truncate the original video feature. Each window is detected independently, and finally the results of all windows are concatenated as the total result of the entire long video. The scale ranges of each layer in the pyramid are set as: \([S_{\text{min}}^1, S_{\text{max}}^1] = [0, 15\%]\), \([S_{\text{min}}^2, S_{\text{max}}^2] = [10\%, 75\%]\) and \([S_{\text{min}}^3, S_{\text{max}}^3] = [50\%, 100\%]\). As for the action classifier, following the routine of most two-stage TAD methods, Untrimmed Net [25] serves as the classifier for THUMOS-14, and the recognition model by [32] for ActivityNet-1.3.

**Implementation Details.** MENet is trained by Adam algorithm with batch size 8 and learning rate \( 10^{-3} \). The sampling ratio of each layer \( r_1, r_2, r_3 \) are set as \( 50\%, 40\%, 90\% \), respectively. For ActivityNet-1.3, the training process takes 10 epochs, where the first 7 epochs is the Generalization stage and the rest is the Specialization stage. As for THUMOS-14, the total training epoch is 7, and only the first epoch is the Generalization stage. Besides, the learning rate is decayed to \( 10^{-4} \) after 5 epochs on THUMOS-14 and 7 epochs on ActivityNet-1.3.

**4.2. Performance Evaluation On Detection**

We compare MENet with other state-of-the-art methods on ActivityNet-1.3 and THUMOS-14. Table 1 shows the comparison on the validation set of ActivityNet-1.3. The mAP at tIoU thresholds of \{0.5, 0.75, 0.95\} are reported, as well as the Avg. mAP which is calculated at tIoU thresholds between 0.5 and 0.95 with the step of 0.05. Impressively, MENet outperforms other representative methods at tIoU thresholds 0.75 and 0.95. MENet promotes the Avg. mAP from 36.6\% to 37.7\%, with an increase of more than 1.1\%. Comparing the methods of different backbones, MENet exceeds others.

Table 2 presents the detection performance comparison on the test set of THUMOS-14. MENet achieves comparable performance with the best method [30] at low tIoU thresholds. But, MENet outperforms other representative methods at tIoU thresholds 0.6 and 0.7. MENet improves the mAP@0.7 from the current best 31.8\% to 34.0\%. To locate the boundaries more accurately, a higher threshold is usually used to separate the positive and negative samples in training. Therefore MENet has a better response at high tIoU thresholds. Higher tIoU means more precise action boundaries, which is more difficult and more important for action localization. Meanwhile, it is noteworthy
that MENet shows obvious superiority compared with other FPN-based methods like [11], which substantiates the effectiveness of the SFPN proposed in MENet.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>mAP@tIoU (%)</th>
<th>0.5</th>
<th>0.75</th>
<th>0.95</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GTAN [16]</td>
<td>P3D</td>
<td>52.6</td>
<td>34.1</td>
<td>8.9</td>
<td>34.3</td>
<td></td>
</tr>
<tr>
<td>BSN [14]</td>
<td>TSN</td>
<td>46.5</td>
<td>30.0</td>
<td>8.0</td>
<td>30.0</td>
<td></td>
</tr>
<tr>
<td>BMN [12]</td>
<td>TSN</td>
<td>50.1</td>
<td>34.8</td>
<td>8.3</td>
<td>33.9</td>
<td></td>
</tr>
<tr>
<td>G-TAD [29]</td>
<td>TSN</td>
<td>50.4</td>
<td>34.6</td>
<td>9.0</td>
<td>34.1</td>
<td></td>
</tr>
<tr>
<td>PCMNet [18]</td>
<td>TSN</td>
<td>51.4</td>
<td>36.1</td>
<td>9.5</td>
<td>35.3</td>
<td></td>
</tr>
<tr>
<td>TCA-Net [19]</td>
<td>TSN</td>
<td>52.3</td>
<td>36.7</td>
<td>6.9</td>
<td>35.5</td>
<td></td>
</tr>
<tr>
<td>RTD-Net [23]</td>
<td>I3D</td>
<td>47.2</td>
<td>30.7</td>
<td>8.6</td>
<td>30.8</td>
<td></td>
</tr>
<tr>
<td>ContextLoc [33]</td>
<td>I3D</td>
<td>56.0</td>
<td>35.2</td>
<td>3.6</td>
<td>34.2</td>
<td></td>
</tr>
<tr>
<td>AFSD [11]</td>
<td>I3D</td>
<td>52.4</td>
<td>35.8</td>
<td>6.5</td>
<td>34.4</td>
<td></td>
</tr>
<tr>
<td>STPT [27]</td>
<td>STPT</td>
<td>51.4</td>
<td>33.7</td>
<td>6.8</td>
<td>33.4</td>
<td></td>
</tr>
<tr>
<td>ActionFormer</td>
<td>[30]</td>
<td>54.7</td>
<td>37.8</td>
<td>8.4</td>
<td>36.6</td>
<td></td>
</tr>
<tr>
<td>MENet (ours)</td>
<td>R(2+1)D</td>
<td>54.7</td>
<td>38.4</td>
<td>10.5</td>
<td>37.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 1. The temporal action detection performance comparison with state-of-art methods on ActivityNet-1.3. Bold text indicates the best results.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>AR@100</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSN [14]</td>
<td></td>
<td>74.2</td>
<td>66.2</td>
</tr>
<tr>
<td>GTAN [16]</td>
<td></td>
<td>74.8</td>
<td>67.1</td>
</tr>
<tr>
<td>BMN [12]</td>
<td></td>
<td>75.0</td>
<td>67.1</td>
</tr>
<tr>
<td>RTD-Net [23]</td>
<td></td>
<td>73.2</td>
<td>65.8</td>
</tr>
<tr>
<td>TCA-Net [19]</td>
<td></td>
<td>76.1</td>
<td>68.1</td>
</tr>
<tr>
<td>MENet (ours)</td>
<td></td>
<td>77.0</td>
<td>70.2</td>
</tr>
</tbody>
</table>

Table 3. Temporal action proposal performance comparison with other representative methods on ActivityNet-1.3. Bold text indicates the best results.

<table>
<thead>
<tr>
<th>Method</th>
<th>AR@50</th>
<th>AR@100</th>
<th>AR@200</th>
<th>AR@500</th>
</tr>
</thead>
<tbody>
<tr>
<td>BSN [14]</td>
<td>37.5</td>
<td>46.1</td>
<td>53.2</td>
<td>60.6</td>
</tr>
<tr>
<td>BMN [12]</td>
<td>39.4</td>
<td>47.7</td>
<td>54.7</td>
<td>62.1</td>
</tr>
<tr>
<td>RTD-Net [23]</td>
<td>41.5</td>
<td>49.3</td>
<td>56.4</td>
<td>62.9</td>
</tr>
<tr>
<td>TCA-Net [19]</td>
<td>42.1</td>
<td>50.5</td>
<td>57.1</td>
<td>63.6</td>
</tr>
<tr>
<td>MENet (ours)</td>
<td>44.2</td>
<td>52.3</td>
<td>58.6</td>
<td>66.0</td>
</tr>
</tbody>
</table>

Table 4. Temporal action proposal performance comparison with other representative methods on THUMOS-14, measured by AR@AN. Bold text indicates the best results.

1). Effectiveness of the MEM: As discussed before, when constructing the movement enhance feature, MFE and MREM are the crucial designs in MEM. To certify the function of these two modules, we design a Base model for comparison, in which the Movement Feature is removed and MREM is replaced by temporal 1D convolution. As shown in Table 5, we can find that compared with the Base model, Movement Feature and MREM both can promote the performance. Besides, the best results emerge when combining them together as the intact MEM.

<table>
<thead>
<tr>
<th>Construction</th>
<th>mAP@tIoU (%)</th>
<th>0.5</th>
<th>0.75</th>
<th>0.95</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base</td>
<td>54.2</td>
<td>37.3</td>
<td>10.4</td>
<td>37.0</td>
<td></td>
</tr>
<tr>
<td>Base + Movement Feature</td>
<td>54.5</td>
<td>37.8</td>
<td>10.6</td>
<td>37.3</td>
<td></td>
</tr>
<tr>
<td>Base + MREM</td>
<td>54.4</td>
<td>37.9</td>
<td>10.4</td>
<td>37.3</td>
<td></td>
</tr>
<tr>
<td>MEM</td>
<td>54.7</td>
<td>38.4</td>
<td>10.5</td>
<td>37.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 5. Ablation study on the MEM. The Detection performance is reported on ActivityNet-1.3. And The best result can be achieved with these two modules are integrated.

2). Effectiveness of the U-shape Module: As discussed before, when constructing the feature pyramid, U-shape Module is designed to establish interactions between different layers. To reveal the validity of this idea, we use 1D convolution with the step of 2 and AvgPooling to downsample the feature sequence respectively. From Table 6, the U-shape Module can promote the performance, which substantiates the importance of this design.

3). Effectiveness of feature pyramid structure: For better detection accuracy, MENet makes use of the feature pyramid. To verify the impact of this structure, we only use a single layer feature sequence to conduct experiments.
and compare it with the SFPN, keeping other components unchanged. On ActivityNet-1.3, single-layer feature with lengths of 200, 100 and 50 are used. Corresponding results are shown in Table 7. The experimental results exhibit that SFPN performs better than any other single-layer structure, thus justifying the feature pyramid structure is indeed beneficial to the performance.

<table>
<thead>
<tr>
<th>Length</th>
<th>mAP@tIoU (%)</th>
<th>0.5</th>
<th>0.75</th>
<th>0.95</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>53.0</td>
<td>37.1</td>
<td>10.8</td>
<td>36.8</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>54.1</td>
<td>38.1</td>
<td>10.4</td>
<td>37.1</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>54.2</td>
<td>38.3</td>
<td>10.5</td>
<td>37.3</td>
<td></td>
</tr>
<tr>
<td>SFPN</td>
<td>54.7</td>
<td>38.4</td>
<td>10.5</td>
<td>37.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 7. Ablation study on the SFPN. “Length” means the temporal length of single layer feature sequence. The Detection performance is reported on ActivityNet-1.3.

4). **Effectiveness of two-stage training**: For efficient training of MENet and ensuring that the $i$-th layer in SFPN is specialized in actions at a specific scale range $[S^i_{\text{min}}, S^i_{\text{max}}]$, the two-stage training strategy is applied. When training ASP of the $i$-th layer, in the first Generalization stage, the anchors are randomly selected from all scales according to the sampling ratio $r_i$, in the second Specialization stage, the sampling ratio inside and outside the range $[S^i_{\text{min}}, S^i_{\text{max}}]$ are $2 \cdot r_i$ and $r_i$, respectively. In this ablation study, we train MENet using the fully Generalization stage, the fully Specialization stage and the complete two-stage strategy, respectively. In addition, to demonstrate the effectiveness of this training method, we also test a new Bias training strategy that only samples from the range $[S^i_{\text{min}}, S^i_{\text{max}}]$ with the sampling ratio of $r_i$.

As shown in Table 8, the performance of the Bias strategy is the worst, indicating that in addition to those samples in the specific scale range, other samples outside this range also play an important role for a better understanding of actions. Further, the training effect is impeded when removing either the Generalization or the Specialization stage, which verifies the effectiveness of the two-stage training strategy.

5). **Qualitative Comparison**: To illustrate the edge of MENet more intuitively, we select videos of Javelin Throw and High Jump from ActivityNet-1.3 and compare our results qualitatively with the Base, in which we remove the movement enhanced feature and SFPN. Results with the highest confidence score are visualized in Fig 5. Firstly, Compared with the Base method, MENet uses the movement feature, which enlarges the difference between action and background, making the model more accurate in boundary localization. Secondly, it can be seen the High Jump action contains a large-scale action and a small-scale action. Due to the SFPN in MENet, MENet can better capture actions at different scales. Results of qualitative comparison once again prove the superiority of MENet.

<table>
<thead>
<tr>
<th>Training Strategy</th>
<th>mAP@tIoU (%)</th>
<th>0.5</th>
<th>0.75</th>
<th>0.95</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>only Bias</td>
<td>45.8</td>
<td>29.1</td>
<td>7.4</td>
<td>29.4</td>
<td></td>
</tr>
<tr>
<td>only Generalization</td>
<td>54.3</td>
<td>38.2</td>
<td>11.0</td>
<td>37.4</td>
<td></td>
</tr>
<tr>
<td>only Specialization</td>
<td>54.6</td>
<td>38.0</td>
<td>10.8</td>
<td>37.4</td>
<td></td>
</tr>
<tr>
<td>two-stage</td>
<td>54.7</td>
<td>38.4</td>
<td>10.5</td>
<td>37.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 8. Ablation study on the two-stage training strategy. The proposed two-stage training leads to better performance than the single stage method or the Bias method.

5. Conclusion

Temporal action detection (TAD) is aimed at localizing the action segment and classifying the action class. Despite the considerable attention and rapid development, TAD is hindered by two critical problems: the movement feature submergence and multi-scale action detection. In an effort to overcome these two challenges, we propose the Movement Enhance Network (MENet) with two crucial designs. Firstly, we propose the Movement Enhance Module (MEM) to extract the movement feature in video snippets and explore the multi-relations between the snippets. Secondly, we propose the Scale FPN (SFPN) to handle the different scale actions, we design the U-shape Module to facilitate the information flow between different scale features. For better performance, we design the training and inference stage, ensuring that each layer in MENet is specialized at a specific scale range. Extensive experiments demonstrate that MENet considerably outperforms other representative TAD methods on ActivityNet-1.3 and THUMOS-14.

**Acknowledgements.** This work was supported by National Natural Science Foundation of China (NSFC) No. 62176156.
References


[32] Yue Zhao, Bowen Zhang, Zhirong Wu, Shuo Yang, Lei Zhou, Sijie Yan, Limin Wang, Yuanjun Xiong, D Lin, Y Qiao, et al. Cuhk & ethz & siat submission to activitynet...