# LoGo: Local-Global Context Modeling and Cross-Level Regression for Temporal Action Localization

Li Xinxin[1,*], Yang Zhe[2]

[1]School of Computer and Software, Chengdu Jincheng College, Chengdu, 611731, China
[2]Tangshan Research Institute, Southwest Jiaotong University, Tangshan 063000, China

**Abstract**

The explosive growth of user-generated videos has driven the demand for automated video understanding in applications such as retrieval, surveillance, and human-computer interaction. Temporal Action Localization (TAL), a critical task in this domain, aims to identify temporal boundaries and categories of actions in untrimmed videos. However, existing methods struggle with challenges including large variations in action duration, ambiguous boundaries, and strong background noise. This paper proposes LoGo, an end-to-end framework that unifies Local-Global Context Modeling and a Cross-Level Feature Fusion Regression Head (CLFF-Head) to significantly improve localization accuracy. The key innovations include: 1) The LoGo Block, which integrates depthwise separable convolutions for local structural modeling with channel attention mechanisms for global semantic awareness, achieving balanced local-global dependency learning through residual fusion; 2) The CLFF-Head, which enhances boundary regression stability and accuracy via adaptive multi-scale feature fusion. Extensive experiments on THUMOS14 and ActivityNet1.3 demonstrate that LoGo outperforms state-of-the-art methods, achieving new SOTA performance on THUMOS14 and competitive results on ActivityNet1.3, validating its effectiveness and generalizability.

**Keywords**

Temporal Action Localization, Video Understanding, Local-Global Context Modeling

## 1. Introduction

In recent years, the explosive growth of user-generated videos on internet platforms has fueled the demand for automatic video understanding in applications such as retrieval, surveillance, and human-computer interaction. Temporal Action Localization (TAL), a crucial task in this domain, aims to identify the temporal boundaries and categories of actions in untrimmed videos. Despite recent advances, TAL remains challenging due to large variations in action duration, ambiguous boundaries, and strong background noise.

To tackle these challenges, many methods have been proposed using convolutional [1, 2], recurrent [3], or graph-based networks[4, 5, 6] to model temporal dependencies. However, the uneven distribution of action durations—ranging from brief gestures to prolonged activities—demands simultaneous modeling of both short- and long-term dependencies. CNNs offer strong local modeling but struggle with long-range context, while Transformers [7, 8] capture global semantics effectively but lack sensitivity to local details, which are critical for precise boundary detection.

Accurate boundary localization remains a bottleneck. While [8] introduces an efficient regression head, its expressiveness is limited in complex scenes. [9] improves regression with a stronger head, yet its limited cross-level feature fusion results in suboptimal localization under challenging backgrounds.

To address these issues, we propose LoGo, an end-to-end TAL framework that unifies Local-Global Context modeling with a Cross-Level Feature Fusion Regression Head (CLFF-Head). The LoGo Block integrates depthwise separable convolutions for local structure modeling with channel attention for global context, connected via a residual fusion mechanism to balance precision and long-range awareness. Furthermore, the CLFF-Head adaptively fuses multi-scale semantic features from the feature pyramid, significantly enhancing boundary regression stability and accuracy without compromising efficiency.

The main contributions of this paper are summarized as follows:

- We introduce the LoGo Block module, which combines the local modeling capability based on depthwise separable convolutions with the global modeling capability driven by channel attention mechanisms, achieving efficient integration through a residual structure. This module effectively captures both local structural details and long-range contextual dependencies.
- We design the Cross-Level Feature Fusion Regression Head (CLFF-Head), which introduces a multi-scale feature adaptive fusion mechanism, significantly improving the stability and accuracy of boundary localization.

Section 2 outlines relevant prior work. Section 3 details the proposed methodology. Section 4 presents and discusses the results of our experiments. Lastly, Section 5 summarizes our main conclusions.

## 2. Related Work

**Temporal Action Localization(TAL)** In Temporal Action Localization (TAL), two-stage and single-stage methods are employed to detect actions in videos. Two-stage methods involve generating action proposals and classifying them, which can be achieved through anchor windows[10, 11], action boundary localization[12, 13], graph representation[4], or Transformers[7, 8]. On the other hand, single-stage TAL performs both proposal generation and classification in a single pass, without a separate proposal generation step. Pioneering work[14] in this field developed anchor-based single-stage TAL using convolutional networks, inspired by single-stage object detectors[15]. Additionally, there have been anchor-free single-stage models proposed[16], incorporating a saliency-based refinement module.

*STRL'25: Third International Workshop on Spatio-Temporal Reasoning and Learning, 16 August 2025, Montreal, Canada*
*Corresponding author.

**Object detection** Object detection is closely related to Temporal Action Localization (TAL), with both tasks sharing similar challenges. General Focal Loss [17] enhances bounding box regression by transforming it from learning a Dirac delta distribution to a more general distribution function. Several methods [18, 19, 20] leverage Depthwise Convolution to model network structures, while certain branched designs [21, 22] have demonstrated strong generalization capabilities. These approaches offer valuable insights for designing the architecture of TAL systems.

# 3. Methodology

**Temporal Action Localization** Given an input video $X$, we make the assumption that $X$ can be represented by a collection of feature vectors $X = \{x_1, x_2, \ldots, x_T\}$ defined at discrete time steps $t = \{1, 2, \ldots, T\}$, where the total duration $T$ varies across different videos. For instance, $x_T$ could represent the feature vector of a video clip extracted from a 3D convolutional network at time $t$. The objective of temporal action localization (TAL) is to predict the action label $Y = \{y_1, y_2, \ldots, y_N\}$ based on the input video sequence $X$. Here, $Y$ comprises $N$ action instances $y_i$, and the number of instances $N$ can vary across videos. Each instance $y_i = (s_i, e_i, a_i)$ is characterized by its starting time $s_i$ (onset), ending time $e_i$ (offset), and the corresponding action label $a_i$. The starting time $s_i$ lies in the range of $[1, T]$, the ending time $e_i$ lies in the range of $[1, T]$, and the action label $a_i$ belongs to the set of pre-defined categories $1, .., C$, where $C$ represents the total number of categories. Additionally, it is required that $s_i$ is less than $e_i$ for each instance. Therefore, the task of TAL presents a challenging problem of predicting structured outputs.

## 3.1. Method Overview

The overall architecture of LoGo is shown in Fig. 1. Our model consists of three parts: a video feature extractor, a Multi-scale LoGo Encoder, and two sub-task heads. Concretely, for a given video clip, we extract video features using a pre-trained 3D-CNN model. Then, the extracted features are passed through the Multi-scale LoGo Encoder, which performs downsampling operations to better represent features at different temporal scales. Finally, the pyramid features produced by the Multi-scale LoGo Encoder are processed by two task-specific heads to generate the final predictions. In the following, we will describe the details of our model.

## 3.2. Multi-scale LoGo Encoder

The input feature $X$ is first encoded into multi-scale temporal feature pyramid $Z = \{Z^1, Z^2, ..., Z^L\}$ using Multi-scale LoGo Encoder $e$. The encoder $e$ simply contains two 1D convolutional neural network layers as feature projection layers, followed by $L - 1$ Local-Global Context Modeling (LoGo) blocks to produce feature pyramid $Z$.

First, the input features $Z^{l-1}$ from the previous layer of the pyramid are passed through a Layer Normalization (LN) operation to stabilize the feature distribution. These normalized features are then fed into the LoGo block, which jointly captures local and global temporal information. Through a residual connection, the output of the LoGo block is added to the original input features, resulting in the new feature representation $\bar{Z}^{l-1}$.

Next, the feature $\bar{Z}^{l-1}$ is processed through a Group Normalization (GN) operation to further enhance the training stability of the model. Then, a Multi-Layer Perceptron (MLP) is applied to perform nonlinear transformation, yielding the feature $\hat{Z}^{l-1}$. Again, through a residual connection, the output of the MLP is added to $\bar{Z}^{l-1}$, producing the updated feature representation.

Finally, the processed feature $\hat{Z}^{l-1}$ undergoes a downsampling operation. Downsampling is implemented via a 1D max-pooling operation with a window size of 3 and a stride of 2, reducing the temporal dimension of the features and passing them to the next layer of the pyramid.

Here are the mathematical formulas corresponding to these steps:

$$\bar{Z}^{l-1} = LoGo(LN(Z^{l-1})) + Z^{l-1}, \tag{1}$$

$$\hat{Z}^{l-1} = MLP(GN(\bar{Z}^{l-1})) + \bar{Z}^{l-1}, \tag{2}$$

$$Z^{l} = downsample(\hat{Z}^{l-1}) \tag{3}$$

where $l \in [1, L]$, $LN$ is the LayerNorm operation, $GN$ is the GroupNorm operation, $downsample$ is implemented by a 1D max-pooling with a window size of 3 and stride of 2.

Finally, the encoded feature pyramid is constructed by combining the features of all the LoGo blocks as $Z = \{Z^1, Z^2, ..., Z^L\}$

**LoGo Block** To simultaneously capture local structural details and global semantics in action instances, this study proposes the LoGo Block module, which integrates depthwise separable convolution (for local modeling) and channel attention mechanisms (for global modeling) to achieve efficient fusion of multi-scale temporal features. Specifically, the LoGo Block first applies LayerNorm normalization to stabilize the feature distribution and improve training effectiveness. It then employs depthwise separable convolution to model local patterns, effectively capturing short-term dependencies and fine-grained variations in temporal sequences.

Global modeling is implemented through two pathways: Pathway 1 generates channel attention weights (global modulation factor) via global average pooling followed by a fully connected layer, while Pathway 2 extracts salient features through max pooling and multiplies them with linearly transformed features for dynamic channel weighting. These pathways respectively focus on global context and salient features to enhance multi-scale action characteristics.

The final output combines three components: local features multiplied by the global modulation factor, dynamically weighted salient features, and the original input through residual connections. This design strengthens feature representation capabilities while facilitating stable gradient propagation.

Overall, the LoGo Block adopts a lightweight structure to enable multi-level modeling of temporal action information, significantly improving the model's ability to recognize action boundaries and understand semantics in complex backgrounds.

Mathematically, the LoGo can be written as:

$$f_{LoGo} = Conv(x) f_{avg} + f_{max} FC(x) + x \tag{4}$$

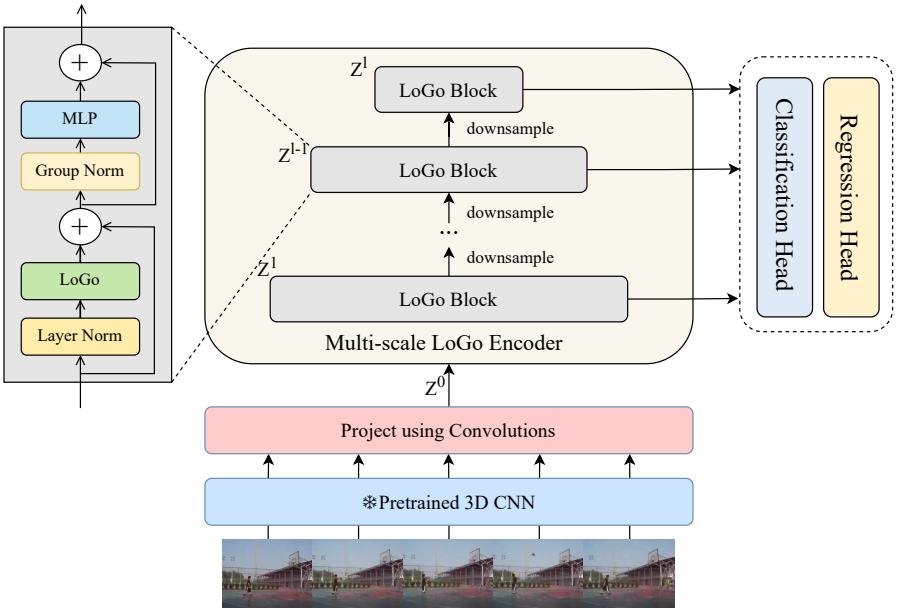


**Figure 1:** Illustration of LoGo framework. We build the pyramid features with LoGo Block. The corresponding features in each level are fed into a lightweight classification head and a CLFF-Head to obtain the result.

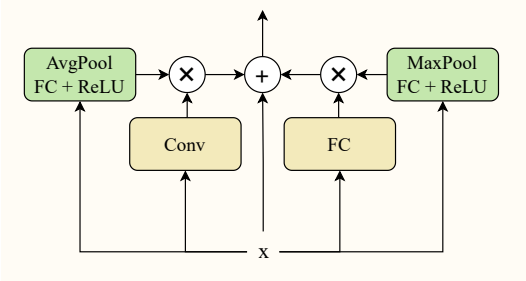


**Figure 2:** Illustration of LoGo. LoGo integrates local convolutional features, global modulation factors, and dynamically weighted features through residual connections to stabilize training.

where $FC$ and $Conv$ denotes fully-connected layer and the 1-D depth-wise convolution layer over temporal dimension. $f_{avg}$ and $f_{max}$ are given as:

$$f_{avg} = ReLU(FC(AvgPool(x))), \quad (5)$$

$$f_{max} = ReLU(FC(MaxPool(x))), \quad (6)$$

where $AvgPool(x)$ is the average pooling for all features over the temporal dimension and $MaxPool(x)$ is the max pooling for all features over the temporal dimension.

## 3.3. Temporal Action Localization Decoder

Next, our model uses a decoder to decode the feature pyramid $Z = \{Z^1, Z^2, ..., Z^l\}$ extracted by the multi-path temporal feature encoder into sequence labels $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_l\}$ . The decoder of this model consists of a classification head and a cross-level feature fusion regression head, both of which are lightweight convolutional networks.

**Classification Head** Based on the feature pyramid $Z$ , the task of the classification head is to predict the action category probabilities $p(a_t)$ for each moment $t$ at different levels of the pyramid. The classification head in this chapter adopts a simple and efficient 1D convolutional network, with shared parameters across different levels to reduce model complexity. Specifically, the network consists of three 1D convolutional layers, with a kernel size of 3. ReLU activation and LayerNorm normalization are applied in the first two layers to enhance training stability. Finally, after a Sigmoid function mapping, the output is the probability distribution for each action category.

**Cross-Level Feature Fusion Regression Head** Inspired by the Trident-head structure in TriDet, the regression head in this paper emphasizes the role of relative boundary feature information at different levels of the feature pyramid in action localization. However, Trident-head relies solely on a single feature layer for boundary estimation, which may limit its performance in boundary localization. To address this, we propose a Cross-Level Feature Fusion Regression Head (CLFFHead). This method not only utilizes the features of the current pyramid layer when predicting boundary offsets but also incorporates the feature information from the previous layer, thus enhancing the complementarity of features at different scales and improving the stability and accuracy of boundary regression.

Given the feature sequence $F \in \mathbb{R}^{T \times D}$ output from the feature pyramid, we first obtain three feature sequences from three branches: $F_s \in \mathbb{R}^T$ , $F_e \in \mathbb{R}^T$ and $F_c \in \mathbb{R}^{T \times 2 \times (B+1)}$. $F_s$ and $F_e$ represent the response values for the start and end boundaries of an action at each moment, respectively. Both $F_s$ and $F_e$ are obtained through 1D convolutions. For $F_c \in \mathbb{R}^{T \times 2 \times (B+1)}$ , during its derivation, we not only utilize the features of the current layer but also incorporate the feature information from the previous pyramid layer. This cross-level feature fusion allows the

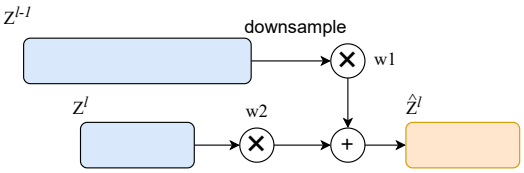


**Figure 3:** Illustration of cross-level feature fusion progress. Cross-level feature fusion with learnable weights $w_1$ and $w_2$ dynamically adjusts fusion ratios.

model to leverage richer contextual information, enhancing the robustness of boundary prediction. In the feature fusion process, we introduce two learnable parameters, $w_1$ and $w_2$, to ensure that the fusion weights can be dynamically adjusted. This process is illustrated in Fig. 3, where the blue blocks denote the features processed by the LoGo Encoder. This mechanism enables the model to adaptively select the appropriate feature proportion, optimizing boundary prediction performance under different tasks and data distributions. For more details about Trident-head decoder, the readers can refer to [9]. All heads are modeled using a three-layer convolutional neural network, with shared parameters across all feature pyramid layers to reduce the number of parameters.

### 3.4. Loss Function

The loss function plays a crucial role in model training by providing an optimization objective that measures the error between predictions and ground truth, thereby evaluating model performance. A well-designed loss function not only impacts learning effectiveness but also directly influences convergence speed. Thus, the selection and optimization of the loss function are critical aspects of model training that cannot be overlooked.

For training, our model adopts a hybrid optimization strategy combining classification loss and regression loss to improve classification accuracy and bounding box localization precision.

Classification Loss: We employ Focal Loss, a loss function specifically designed to address class imbalance. By introducing a modulating factor, Focal Loss assigns higher weights to hard-to-classify samples, enhancing the model's focus on challenging categories during training and improving overall classification performance.

Regression Loss: We utilize GIoU Loss (Generalized IoU). Compared to traditional IoU, GIoU not only considers the overlapping area between predicted and ground-truth bounding boxes but also accounts for differences in their minimum enclosing rectangles. This improvement addresses the limitations of IoU when bounding boxes are not fully aligned, enabling more precise positional optimization and enhancing temporal localization accuracy for action regions.

Each layer $l$ in the feature pyramid outputs a temporal feature $F^l \in \mathbb{R}^{2^{l-1}T \times D}$ , which is then processed by a classification head and a cross-layer feature fusion regression head for temporal action localization. The output of layer $l$ at time $t$ is denoted as $\hat{o}_t^l = (\hat{c}_t^l, \hat{d}_{st}^l, \hat{d}_{et}^l)$ . The overall loss function is formulated as:

$$\mathcal{L} = \frac{1}{N_{\text{pos}}} \sum_{l,t} \mathbb{1}_{\{c_t^l > 0\}} (\sigma_{\text{IoU}} \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{reg}}) + \frac{1}{N_{\text{neg}}} \sum_{l,t} \mathbb{1}_{\{c_t^l = 0\}} \mathcal{L}_{\text{cls}}, \tag{7}$$

where $N_{\text{pos}}$ and $N_{\text{neg}}$ represent the numbers of positive and negative samples, respectively.

## 4. Experiments

### 4.1. Dataset and Experimental Setup

**Datasets** We conduct experiments on two datasets, including *THUMOS14*, and *ActivityNet-1.3*. These datasets have been widely adopted as standard benchmarks in the temporal action localization task.

*THUMOS14* is a large-scale video dataset, which contains a large number of open-source videos capturing human actions from 20 classes in real environments. Among all the videos, there are 220 (3,007 action instances) and 213 (3,358 action instances) untrimmed videos with temporal annotations in validation and test set, respectively. Following the common setting in THUMOS14, we use the validation set for training and report results on the test set.

*ActivityNet-1.3* is another popular large-scale dataset for TAL. It includes around 20,000 videos (more than 600 hours) with 200 action categories. The dataset has three subsets:10,024 videos for training, 4,926 for validation, and 5,044 for testing. On average, each video comprises approximately 1.5 actions. Following the common practice, we train our model on the training set and report the performance on the validation set.

**Evaluation Metric** We use the mean Average Precision (mAP) at various temporal Intersection over Unions (tIoU) thresholds to evaluate the TAL performance of different methods. For THUMOS14 datasets, we report the results at IoU thresholds [0.3:0.7:0.1]. For ActivityNet-1.3 dataset, we report the results at IoU thresholds [0.5,0.75,0.95].

**Implementation Details** Our model is trained end-to-end with AdamW [23] optimizer. The initial learning rate is set to $10^{-4}$ for THUMOS14 and $10^{-3}$ for ActivityNet. We detach the gradient before the start boundary head and end boundary head and initialize the CNN weights of these two heads with a Gaussian distribution $N(0, 0.1)$ to stabilize the training process. The learning rate is updated with Cosine Annealing schedule. We train 40 and 15 epochs for THUMOS14, ActivityNet (containing warmup 20, 10 epochs). We conduct our experiments on a single NVIDIA A100 GPU.

### 4.2. Main Results

**THUMOS14** We adopt the commonly used I3D [33] as our backbone feature and Tab. 1 presents the results. Our method achieves an average mAP of 69.2%, outperforming all previous methods including one-stage and two-stage methods. Notably, our method also achieves better performance than recent Transformer-based methods [7, 8], which demonstrates that the simple design can also have impressive results. Its performance improvement is more evident at higher IoU thresholds (e.g., 0.6 and 0.7), highlighting the model's strength in accurate localization. Although our regression head is inspired by TriDet, the overall architecture

**Table 1**
Comparison with the state-of-the-art methods on THUMOS14 dataset.

| Method | Backbone | 0.3 | 0.4 | 0.5 | 0.6 | 0.7 | Avg. |
|---|---|---|---|---|---|---|---|
| BMN[12] | TSN | 56.0 | 47.4 | 38.8 | 29.7 | 20.5 | 38.5 |
| TCANet[14] | TSN | 60.6 | 53.2 | 44.6 | 36.8 | 26.7 | 44.3 |
| ContextLoc[24] | I3D | 68.3 | 63.8 | 54.3 | 41.8 | 26.2 | 50.9 |
| AFSD[16] | I3D | 67.3 | 62.4 | 55.5 | 43.7 | 31.1 | 52.0 |
| ReAct[25] | TSN | 69.2 | 65.0 | 57.1 | 47.8 | 35.6 | 55.0 |
| TadTR[26] | I3D | 74.8 | 69.1 | 60.1 | 46.6 | 32.8 | 56.7 |
| TALLFormer[7] | Swin | 76.0 | - | 63.2 | - | 34.5 | 59.2 |
| ActionFormer[8] | I3D | 82.1 | 77.8 | 71.0 | 59.4 | 43.9 | 66.8 |
| TemporalMaxer[27] | I3D | 82.8 | 78.9 | 71.8 | 60.5 | 44.7 | 67.7 |
| ASL[28] | I3D | 83.1 | 79.0 | 71.7 | 59.7 | 45.8 | 67.9 |
| K.Xia et al.[29] | I3D | 81.6 | 78.4 | 72.2 | 59.0 | 44.5 | 67.1 |
| TransGMC[30] | I3D | 82.3 | 78.8 | 71.4 | 60.0 | 45.1 | 67.5 |
| FAM[31] | I3D | 82.8 | 79.1 | 71.1 | 59.8 | 44.2 | 67.4 |
| DualH[32] | I3D | 83.6 | 79.5 | 72.2 | 60.0 | 44.9 | 68.0 |
| **Ours** | I3D | **83.9** | **80.3** | **72.7** | **62.5** | **46.4** | **69.2** |

**Table 2**
Comparison with the state-of-the-art methods on ActivityNet-1.3 dataset.

| Method | Backbone | 0.5 | 0.75 | 0.95 | Avg. |
|---|---|---|---|---|---|
| BMN[12] | TSN | 50.1 | 34.8 | 8.3 | 33.9 |
| AFSD[16] | I3D | 49.6 | 33.0 | 8.6 | 32.6 |
| ReAct[25] | TSN | 49.6 | 33.0 | 8.6 | 32.6 |
| TadTR[26] | TSN | 51.3 | 35.0 | **9.5** | 34.6 |
| TALLFormer[7] | Swin | 54.1 | 36.2 | 7.9 | 35.6 |
| ActionFormer[8] | R(2+1)D | 54.7 | **37.8** | 8.4 | 36.6 |
| ASL[28] | I3D | 54.1 | 37.4 | 8.0 | 36.2 |
| K.Xia et al.[29] | I3D | 54.2 | 35.1 | 7.3 | 34.2 |
| TransGMC[30] | R(2+1)D | **54.8** | 37.6 | 8.5 | **36.7** |
| FAM[31] | I3D | 54.6 | 37.2 | 8.3 | 36.3 |
| DualH[32] | I3D | 54.4 | 37.1 | 8.1 | 36.3 |
| **Ours** | R(2+1)D | **54.8** | **37.8** | 8.5 | **36.7** |

of our framework differs significantly from TriDet. Therefore, a direct comparison may not effectively demonstrate the effectiveness of CLFFHead. Instead, we choose to validate it through ablation studies under a unified framework.

**ActivityNet** ActivityNet. For the ActivityNet v1.3 dataset, we adopt the TSP R(2+1)D [34] as our backbone feature. Following previous methods [7, 8, 16], the video classification score predicted from the UntrimmedNet is adopted to multiply with the final detection score. Tab. 2 presents the results. our method achieves the highest scores at IoU thresholds of 0.5 and 0.75, reaching 54.8% and 37.8% respectively—on par with TransGMC. The average mAP also reaches 36.7%, matching or slightly outperforming the current best methods. Although the performance at the strictest IoU threshold (0.95) is slightly lower than TadTR[26] , our method still maintains a leading overall performance, especially under the more commonly used thresholds of 0.5 and 0.75. This suggests that our method is not only effective for densely labeled and boundary-ambiguous videos (such as those in THUMOS14), but also adaptable to longer videos with large action spans in more complex scenes (as in ActivityNet-1.3).

## 4.3. Ablation Study

In this section, we mainly conduct the ablation studies on the THUMOS14 dataset.

**The effectiveness of LoGo block** To assess the contribution of the LoGo block, we conduct a set of ablation studies by replacing or simplifying its components. Specifically, we begin with a baseline temporal feature pyramid adopted from [8, 16], which consists of two 1D convolutional layers and a shortcut connection. We then progressively enhance this baseline by introducing ActionFormer (SA), the LoGo block, and the CLFF-Head.

As shown in Tab. 3, replacing the standard convolutional block with self-attention improves the average mAP from 62.1% to 66.8%. Further adding the LoGo block yields a notable gain to 68.0%, and finally, the full model with LoGo and CLFF-Head achieves the best performance with an average mAP of 69.2% on THUMOS14.

**Table 3**
Analysis of the Effectiveness of three main components on THUMOS14.

| Method | SA | LoGo | CLFFHead | 0.3 | 0.5 | 0.7 | Avg. |
|---|---|---|---|---|---|---|---|
| 1 | | | | 77.3 | 65.2 | 40.0 | 62.1 |
| 2 | ✓ | | | 82.1 | 71.0 | 43.9 | 66.8 |
| 4 | | ✓ | | 83.5 | 72.3 | 45.1 | 68.0 |
| 3 | | ✓ | ✓ | **83.9** | **72.7** | **46.4** | **69.2** |

**The effectiveness of regression head** To verify the effectiveness of the proposed Cross-Level Feature Fusion Regression Head, we conduct ablation studies on three types of regression heads: (1) a lightweight regression head adopted from [8], (2) the regression head used in [9], and (3) our proposed Cross-Level Feature Fusion Regression Head. All other hyperparameters (e.g., the number of pyramid layers) are kept identical to those used in our framework. Tab. 4 presents the results. While the regression head from TriDet performs slightly better at high IoU (0.7), our proposed head shows more stable performance across thresholds and achieves the highest overall mAP (69.2%). This demonstrates the benefits of integrating multi-level features to enhance regression robustness and precision.

**The effectiveness of feature pyramid level** To investigate the impact of the number of feature pyramid layers on model performance, we conducted experiments with different pyramid depths and evaluated their performance under multiple IoU thresholds. Tab. 5 presents the results. As the number of layers increases from 3 to 6, performance steadily

**Table 4**
Analysis of the Effectiveness of Regression Head on THUMOS14.

| Method | 0.3 | 0.5 | 0.7 | Avg. |
|---|---|---|---|---|
| 1 | 83.0 | **72.9** | 45.2 | 68.0 |
| 2 | 83.5 | 72.5 | **46.8** | 68.8 |
| 3 | **83.9** | 72.7 | 46.4 | **69.2** |

improves, peaking at 69.2% mAP with 6 layers. However, further increasing to 7 layers slightly degrades performance, suggesting that while deeper pyramids can better capture multi-scale action information, excessive depth may introduce redundancy or training difficulties.

**Table 5**
Analysis of the number of feature pyramid layers.

| Levels | 0.3 | 0.5 | 0.7 | Avg. |
|---|---|---|---|---|
| 3 | 79.4 | 65.4 | 37.7 | 61.8 |
| 4 | 82.1 | 70.5 | 43.9 | 66.1 |
| 5 | 82.9 | 70.8 | 45.5 | 67.8 |
| 6 | **83.9** | **72.7** | **46.4** | **69.2** |
| 7 | 83.5 | 72.1 | 45.9 | 68.8 |

## 5. Conclusions

This paper addresses the critical challenge of jointly optimizing local detail capture and global semantic modeling in temporal action localization (TAL) by proposing the LoGo framework based on local-global context modeling. The designed LoGo Block integrates the local structural modeling capability of depthwise separable convolutions with the global semantic awareness of channel attention mechanisms, achieving efficient fusion of multi-scale temporal features. Furthermore, the proposed Cross-Level Feature Fusion Regression Head (CLFF-Head) significantly enhances boundary localization stability and accuracy through adaptive fusion of multi-level semantic information from the feature pyramid. Experiments on THUMOS14 and ActivityNet1.3 demonstrate that the LoGo framework excels in complex backgrounds and multi-scale action scenarios, outperforming existing methods. These results validate the effectiveness of the local-global collaborative modeling strategy and cross-layer feature fusion mechanism.

## Acknowledgments

This work is partially supported by National Natural Science Foundation of China (62376231), Sichuan Science and Technology Program (24NSFSC1070), Fundamental Research Funds for the Central Universities (2682025ZTPY052).

## References

[1] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, D. Lin, Temporal action detection with structured segment networks, in: Proceedings of the IEEE international conference on computer vision, 2017, pp. 2914–2923.

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

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

[4] Y. Bai, Y. Wang, Y. Tong, Y. Yang, Q. Liu, J. Liu, Boundary content graph neural network for temporal action proposal generation, in: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII 16, Springer, 2020, pp. 121–137.

[5] M. Xu, C. Zhao, D. S. Rojas, A. Thabet, B. Ghanem, G-tad: Sub-graph localization for temporal action detection, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 10156–10165.

[6] Z. Yang, J. Qin, D. Huang, Acgnet: Action complement graph network for weakly-supervised temporal action localization, in: Proceedings of the AAAI conference on artificial intelligence, volume 36, 2022, pp. 3090–3098.

[7] F. Cheng, G. Bertasius, Tallformer: Temporal action localization with a long-memory transformer, in: European Conference on Computer Vision, Springer, 2022, pp. 503–521.

[8] C.-L. Zhang, J. Wu, Y. Li, Actionformer: Localizing moments of actions with transformers, in: European Conference on Computer Vision, Springer, 2022, pp. 492–510.

[9] D. Shi, Y. Zhong, Q. Cao, L. Ma, J. Li, D. Tao, Tridet: Temporal action detection with relative boundary modeling, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 18857–18866.

[10] V. Escorcia, F. Caba Heilbron, J. C. Niebles, B. Ghanem, Daps: Deep action proposals for action understanding, in: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14, Springer, 2016, pp. 768–784.

[11] S. Buch, V. Escorcia, C. Shen, B. Ghanem, J. Carlos Niebles, Sst: Single-stream temporal action proposals, in: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, 2017, pp. 2911–2920.

[12] T. Lin, X. Liu, X. Li, E. Ding, S. Wen, Bmn: Boundary-matching network for temporal action proposal generation, in: Proceedings of the IEEE/CVF international conference on computer vision, 2019, pp. 3889–3898.

[13] G. Gong, L. Zheng, Y. Mu, Scale matters: Temporal scale aggregation network for precise action localization in untrimmed videos, in: 2020 IEEE international conference on multimedia and expo (ICME), IEEE, 2020, pp. 1–6.

[14] Z. Qing, H. Su, W. Gan, D. Wang, W. Wu, X. Wang, Y. Qiao, J. Yan, C. Gao, N. Sang, Temporal context aggregation network for temporal action proposal refinement, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 485–494.

[15] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You

only look once: Unified, real-time object detection, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.

[16] C. Lin, C. Xu, D. Luo, Y. Wang, Y. Tai, C. Wang, J. Li, F. Huang, Y. Fu, Learning salient boundary feature for anchor-free temporal action localization, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 3320–3329.

[17] X. Li, W. Wang, L. Wu, S. Chen, X. Hu, J. Li, J. Tang, J. Yang, Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection, Advances in neural information processing systems 33 (2020) 21002–21012.

[18] F. Chollet, Xception: Deep learning with depthwise separable convolutions, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1251–1258.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, H. Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861 (2017).

[20] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, S. Xie, A convnet for the 2020s, in: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2022, pp. 11976–11986.

[21] J. Hu, L. Shen, G. Sun, Squeeze-and-excitation networks, in: Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 7132–7141.

[22] C. Szegedy, S. Ioffe, V. Vanhoucke, A. Alemi, Inception-v4, inception-resnet and the impact of residual connections on learning, in: Proceedings of the AAAI conference on artificial intelligence, volume 31, 2017.

[23] I. Loshchilov, F. Hutter, Decoupled weight decay regularization, arXiv preprint arXiv:1711.05101 (2017).

[24] Z. Zhu, W. Tang, L. Wang, N. Zheng, G. Hua, Enriching local and global contexts for temporal action localization, in: Proceedings of the IEEE/CVF international conference on computer vision, 2021, pp. 13516–13525.

[25] D. Shi, Y. Zhong, Q. Cao, J. Zhang, L. Ma, J. Li, D. Tao, React: Temporal action detection with relational queries, in: European conference on computer vision, Springer, 2022, pp. 105–121.

[26] X. Liu, Q. Wang, Y. Hu, X. Tang, S. Zhang, S. Bai, X. Bai, End-to-end temporal action detection with transformer, IEEE Transactions on Image Processing 31 (2022) 5427–5441.

[27] T. N. Tang, K. Kim, K. Sohn, Temporalmaxer: Maximize temporal context with only max pooling for temporal action localization, arXiv preprint arXiv:2303.09055 (2023).

[28] J. Shao, X. Wang, R. Quan, J. Zheng, J. Yang, Y. Yang, Action sensitivity learning for temporal action localization, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 13457–13469.

[29] K. Xia, L. Wang, Y. Shen, S. Zhou, G. Hua, W. Tang, Exploring action centers for temporal action localization, IEEE Transactions on Multimedia 25 (2023) 9425–9436.

[30] J. Yang, P. Wei, Z. Ren, N. Zheng, Gated multi-scale transformer for temporal action localization, IEEE Transactions on Multimedia 26 (2023) 5705–5717.

[31] W. Wu, T. Lu, J. Wang, P. Tang, F. Gao, Temporal action detection with frequency attention mechanism, in: 2024 7th International Conference on Mechatronics, Robotics and Automation (ICMRA), IEEE, 2024, pp. 137–141.

[32] Z. Zhang, C. Palmero, S. Escalera, Dualh: A dual hierarchical model for temporal action localization, in: 2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG), IEEE, 2024, pp. 1–10.

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

[34] H. Alwassel, S. Giancola, B. Ghanem, Tsp: Temporally-sensitive pretraining of video encoders for localization tasks, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 3173–3183.