An Adaptive Dual Selective Transformer for Temporal Action Localization

Qiang Li, Guang Zu, Hui Xu, Jun Kong, Yanni Zhang, and Jianzhong Wang

***Abstract*—Temporal action localization (TAL), which aims to identify and localize actions in long untrimmed videos, is a challenging task in video understanding. Recent studies have shown that the Transformer and its variants are effective at improving the performance of TAL. The success of the Transformer can be attributed to the use of multi-head self-attention (MHSA) as a token mixer to capture long-term temporal dependencies within the video sequence. However, in the existing Transformer architecture, the features obtained by multiple token mixing (i.e., self-attention) heads are treated equally, which neglects the distinct characteristics of different heads and hampers the exploitation of discriminative information. To this end, we present a new method called the adaptive dual selective Transformer (ADSFormer) for TAL in this paper. The key component in ADSFormer is the dual selective multi-head token mixer (DSMHTM), which integrates multiple feature representations from different token mixing heads by adaptively selecting important features across both the head and channel dimensions. Moreover, we also incorporate our ADSFormer into a pyramid structure so that the multi-scale features obtained can be effectively combined to improve TAL performance. Benefiting from the dual selective multi-head token mixer (DSMHTM) and pyramid feature combination, ADSFormer outperforms several state-of-the-art methods on four challenging benchmark datasets: THUMOS14, MultiTHUMOS, EPIC-KITCHENS-100 and ActivityNet-1.3.**

***Index Terms*—Temporal action localization, action recognition, vision transformers, video understanding.**

I. Introduction

TEMPORAL action localization (TAL) is a computer vision task that focuses on identifying and localizing specific actions in video sequences. The objective of this task is to recognize the categories of continuous actions in the video and determine their corresponding start and end time. TAL plays a significant role in various video analysis problems. Thus, it has been used in a wide range of applications [1], [2], [3], such as event detection, action retrieval, behavior analysis and video summarization.

Currently, numerous methods have been proposed to address the TAL task. In early studies, most TAL methods were built based on handcrafted features, such as static features [4], [5], dynamic optical flow features [5], spatial-temporal features [6] and trajectory features [7]. By extracting these features from videos and combining them with other techniques (such as the hidden Markov model [4], support vector machines [5], [7] and latent topic models [6]), TAL tasks can be implemented. Although the methods based on handcrafted features have strong interpretability, it is difficult for these methods to meet the demands of various scenarios due to their lack of generalizability.

With the continuous development of deep learning, many deep learning-based TAL methods have emerged. These methods can be divided into two categories: two-stage TAL and one-stage TAL. Two-stage TAL methods [8], [9], [10], [11], [12] first generate candidate video segments as action proposals, followed by categorizing the proposals into different action classes and refining their temporal boundaries. Although two-stage approaches typically yield superior performances, the proposals in them are susceptible to handcrafted anchors and confidence thresholds. One-stage methods [13], [14], [15] achieve temporal action localization and classification via a single network without requiring a separate proposal generation step. Thus, they are less complex than two-stage methods and can be easily trained in an end-to-end manner. In typical deep learning-based TAL methods, some deep networks, such as I3D [16], SlowFast [17] and TSN [18] are first adopted to extract the features of each frame in the video. Then, the features of the entire video are mapped to a latent space by a backbone network, i.e., an encoder. Finally, a decoder is utilized to classify the actions and estimate their corresponding boundaries. The encoder is crucial in deep learning-based TAL since it is responsible for temporal context modeling. Thus, many techniques, such as 1D convolutional neural networks (CNNs) [13], graph convolution networks (GCNs) [19], cross-scale graph pyramid networks (xGPNs) [20] and recurrent networks [21], have been leveraged to serve as encoders in TAL.

Recently, Transformer [22] has attracted much attention due to its successful application to natural language processing and computer vision problems. Compared with other techniques (such as CNNs), the advantage of the Transformer

Manuscript received 5 August 2023; revised 5 December 2023 and 17 January 2024; accepted 6 February 2024. Date of publication 20 February 2024; date of current version 24 April 2024. This work was supported in part by NSFC under Grant 62272096 and in part by Jilin Provincial Science and Technology Department under Grant 20210201077GX. The Associate Editor coordinating the review of this manuscript and approving it for publication was Prof. Sanghoon Lee. *(Qiang Li and Guang Zu contributed equally to this work.) (Corresponding authors: Jun Kong; Jianzhong Wang.)*

Qiang Li, Guang Zu, Hui Xu, Yanni Zhang, and Jianzhong Wang are with the School of Information Sciences and Technology, Northeast Normal University, Changchun 130117, China (e-mail: liq782@nenu.edu.cn; zug727@nenu.edu.cn; xuh504@nenu.edu.cn; zhangyn500@nenu.edu.cn; wangjz019@nenu.edu.cn).

Jun Kong is with the School of Information Sciences and Technology and Key Laboratory of Applied Statistics of MOE, Northeast Normal University, Changchun 130117, China (e-mail: kongjun@nenu.edu.cn).

We publicly released our codes and pretrained models at: https://github.com/LiQiang0307/ADSFormer.

Digital Object Identifier 10.1109/TMM.2024.3367599

is that it can capture global dependencies among tokens by utilizing self-attention for token mixing. In TAL, it is well known that capturing long-term temporal dependencies among frame features can facilitate the identification and localization of complex actions over a long period. Thus, Transformer and its variants have been introduced into the TAL task and achieved impressive performance [23], [24], [25], [26]. ActionFormer [24] is a representative method that directly employs a multi-head self-attention-based Transformer for one-stage anchor-free TAL. Subsequently, TemporalMaxer [25] followed the macro-architecture of ActionFormer but replaced the Transformer encoder with simple max pooling to reduce the redundancy in features and accelerate training speed. In TriDet [26], Shi et al. utilized a convolution-based scalable-granularity perception (SGP) layer as a substitution for self-attention in Transformer architecture to improve TAL performance. Furthermore, a novel Trident-head decoder was also proposed in TriDet to obtain more accurate action boundary estimations.

Although TemporalMaxer and TriDet realized that the rank loss problem in self-attention of Transformer leads to learned features exhibiting high similarity and low discriminability [27], [28], their solutions for mitigating this limitation may not be optimal. First, TemporalMaxer simply adopts max pooling to maintain the features with the largest values in a local window. Thus, it also inherits the drawback of max pooling. That is, only the maximum feature in the receptive field is considered, and the others are all neglected. In some cases, it may lose the distinguishing information [21]. Second, TriDet proposed a two-branch convolutional network to extract multiple level features so that the feature diversity can be increased. However, high diversity does not always indicate high discriminability. For example, although principal component analysis (PCA) is a classical and very widely used dimensionality reduction technique that maximizes the diversity (i.e., variance) of low-dimensional features, a common criticism of this method is its deficiency of discriminative ability [29]. Moreover, several recent studies have shown that although the diversity of self-attention features is low, Transformer-based methods still outperform their CNN counterparts in various computer vision tasks [27], [28].

To improve the performance of TAL, we present an adaptive dual selective Transformer (ADSFormer) in this paper. Different from TemporalMaxer and TriDet, which replace the self-attention-based token mixer with max pooling and convolution operations, the aim of our ADSFormer is to exploit the discriminative information from multiple token mixing heads as well as possible. For instance, as a classical token mixer, multi-head self-attention projects the input features into multiple subspaces and mixes the tokens by self-attention within each subspace in parallel, which is considered as a critical reason for the success of Transformer [30]. However, most of the existing Transformer-based methods (including ActionFormer) combine the different features obtained by multiple token mixing heads equally, which inevitably neglects the distinct information captured by different heads [31] and impairs their performance. To address this issue, a dual selective multi-head token mixer (DSMHTM) is proposed in our ADSFormer. The dual selective mechanism enables our DSMHTM to adaptively adjust the weights of features in both the head and channel dimensions and allows the discriminative features to be well mined to boost TAL performance. Inspired by other studies [24], [25], [26], we also leverage a pyramid network structure to combine the multi-scale features obtained by ADSFormer. The experimental results on four benchmark datasets demonstrate that our proposed approach is effective for TAL and outperforms other related methods.

The contributions of this paper can be summarized as follows:

1) We propose an end-to-end temporal action localization framework called ADSFormer, which can directly predict action categories and boundaries in videos without artificial anchors or proposals.
2) We propose a novel Transformer-based backbone that uses the dual selective multi-head token mixer (DSMHTM) module to adaptively adjust the weights of features in different heads and channels, resulting in more discriminative features.
3) The dual selective mechanism is not limited to the self-attention-based token mixer in the vanilla Transformer. Thus, this approach can be seamlessly integrated with any multi-head token mixer to improve its performance.
4) In the experimental section, we incorporate the proposed DSMHTM into two representative token mixers, i.e., multi-head self-attention (MHSA) and adaptive Fourier neural operators (AFNO) [32], resulting in two variants of our method: ADSFormer_SA and ADSFormer_AFNO. The superiority and effectiveness of the two variants are demonstrated by extensive experimental results on challenging datasets including THUMOS14 [33], EPIC-KITCHENS-100 [34], MultiTHUMOS [35] and ActivityNet-1.3 [36].

The remainder of this paper is organized as follows: Section II briefly reviews some related work. The proposed ADSFormer is detailly described in Section III. Section IV provides the experimental results and ablation studies to validate the effectiveness of our method. We conclude this paper and present future work in Section V.

II. Related Work

In this section, some previous work which related to our study is reviewed.

A. Temporal Action Localization (TAL)

Temporal action localization (TAL) is the task to localize and classify all actions present in an untrimmed video. Some literatures [24], [25], [26] broadly divided the existing TAL methods into two different categories: two-stage methods and one-stage methods.

In two-stage approaches, action proposals are first generated in a category-agnostic manner, and then the classification and temporal boundary refinement are applied for each proposal. Since the proposal generation is fundamental and indispensable in two-stage based TAL, many studies [8], [9], [10], [11], [12] put their emphasis on this stage. Currently, the mainstream techniques for generating proposals include anchor windows

classification [37], action boundaries detection [10], [11], [38], [39], graph-based proposal relationships exploration [19], [40], [41] and fine-grained temporal representation design [42], [43]. The method proposed in [8] employed a long short-term memory (LSTM) network and video temporal information to calibrate the temporal boundaries of sliding windows. Boundary matching network (BMN) [10] proposed a mechanism which denoted proposal as a matching pair of start and end boundaries and converted all proposals to a boundary-matching confidence map. Boundary sensitive network (BSN) [11] adopted a 'local-to-global' strategy to generate proposals. In this strategy, the proposals were first obtained by gathering locations with high probabilities, and then retrieved by evaluating the confidence of whether a proposal contains an action within its region. Xu et al. casted TAL as a sub-graph localization problem and proposed a graph convolutional network (GCN) to adaptively incorporate multi-level semantic context captured from videos [19]. Although the two-stage methods have achieved satisfactory performance on TAL task, they always suffer from the high complexity problem in both model size and optimization aspects. That is, the two-stage TAL methods have a large number of parameters and cannot be trained in an end-to-end manner.

One-stage approaches abandon the proposal-then-classification strategy, and perform TAL on the input videos in a single shot manner. Some previous studies [13], [14], [15] implemented the one-stage TAL by building a hierarchical network architecture based on convolutional neural network (CNN). Ref. [15] presented the first one-stage TAL using 1D temporal convolutional networks, which skipped the proposal generation step via directly detecting action instances. Based on convolutional network, gaussian temporal awareness networks (GTAN) utilized Gaussian kernels to dynamically optimize the scale of each anchor [44]. Lin et al. introduced a one-stage anchor-free model through integrating a saliency-based refinement module into convolutional network [13]. Nag et al. also presented a proposal-free one-stage model which learned global segmentation masks of various actions jointly [45].

B. Transformer in TAL

Transformer has achieved remarkable success in natural language processing, particularly in machine translation tasks. ViT [46] and its variants also demonstrated the impressive performance of Transformer in many computer vision tasks. For TAL, since Transformer is well-suited to capture long-range dependencies among the feature representations of video frames, it has been adopted to improve the action localization and classification performance [23], [24], [47], [48], [49].

Some researchers introduced Transformer into two-stage TAL framework [47], [50], [51]. Tan et al. employed Transformer as a decoder to model the inter-proposal dependencies from a global view for better proposal generation [47]. TAPG Transformer [51] proposed a boundary Transformer and a proposal Transformer to capture long-term context in the video and model proposal relationships, respectively. ATAG [50] presented a dual-path module consists of an augmented Transformer and an adaptive GCN for global and local temporal context exploitation. The Transformer effectively enhanced the performance of two-stage TAL. However, it should be noted that the methods in [47], [50], [51] still cannot be trained in end-to-end manner.

Recently, Transformer has also been incorporated into one-stage based TAL methods [23], [24], [25], [26], [48]. TadTR [48] extracted the video context information by a temporal deformable attention based Transformer which selectively attends to a sparse subset in a video. To improve the efficiency of TAL, Cheng et al. [23] proposed an end-to-end trainable TALLFormer. Through a short-term Transformer encoder and a long-term memory mechanism, the GPU memory cost in TALLFormer can be reduced. Moreover, the three methods we have introduced in Section I (i.e., ActionFormer [24], TemporalMaxer [25] and TriDet [26]) are also state-of-the-art approaches which employed Transformer or its variants in one-stage framework to promote the TAL performance. Nevertheless, as we have analyzed, they still have some inherent limitations and neglect the discriminative information in multi-head token mixing features.

C. Token Mixers in Transformer

The vanilla Transformer employs the multi-head self-attention as token mixer, which captures the global similarity between tokens in different subspaces based on self-attention [46]. However, the self-attention suffers from a quadratic complexity with respect to the number of tokens. To simplify token mixing and reduce the number of parameters, some studies substituted the multi-head self-attention in Transformer with multi-layer perceptron (MLP) [52], [53]. MLP-Mixer employed MLPs to mix tokens and channels alternatively [53]. gMLP utilized a spatial gating unit to capture the interactions among tokens [52]. Nevertheless, the MLP based token mixers still lack scalability due to the quadratic complexity of MLP projection. Thus, they are inefficient for some tasks with a large number of input tokens (such as TAL) [32], [54]. Additionally, some other researchers investigated Transformer from the architecture aspect and employed a non-parametric average pooling operator as an alternative to multi-head self-attention for token mixing [55]. More recently, the Fourier transform has also been leveraged to accelerate the token mixing and achieved state-of-the-art performance on some tasks [32], [54], [56]. FNet [56] combined the input tokens and the real part of their Fourier spectrum for token mixing. GFNet [54] learned a global filter to mix the tokens in Fourier domain. AFNO [32] framed parallelly multi-head token mixing as efficient convolutions in Fourier domain based on a principled foundation of operator learning.

III. Method

A. Problem Statement

Given an untrimmed video X, we assume that X can be represented by a set of feature vectors $X = \{x_1, x_2, \ldots, x_T\}$, where T is the number of discrete time steps and its value depends on the length of the video. For instance, $x_t(t = 1, \ldots, T)$ can be the feature vector extracted by a pretrained 3D convolutional

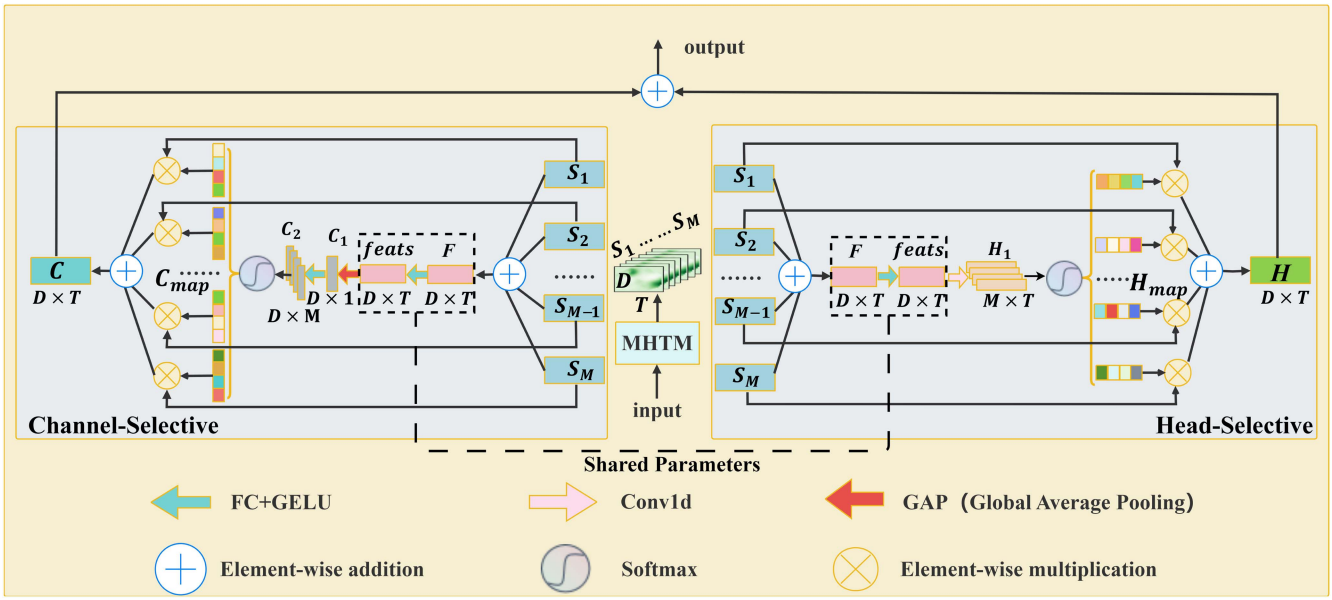

Fig. 1. Illustration of the structure of Dual Selective Multi-Head Token Mixer (DSMHTM).

network (e.g., I3D [16], SlowFast [17]) from a frame at moment t in the input video. The aim of TAL is to predict a set of action instances $Y = \{y_1, y_2, \ldots, y_N\}$ based on the input video sequence X, where N is the number of actions. The n-th $(n = 1, \ldots, N)$ element in Y is defined as $y_n = (s_n, e_n, a_n)$, where $s_n \in [1, T]$, $e_n \in [1, T]$ and $a_n \in \{1, \ldots, N_C\}$ are the start time (onset), end time (offset) and corresponding label of the n-th action, respectively. N_C is the predefined number of action categories.

B. Dual Selective Token Mixer

As we have discussed previously, the dual selective multi-head token mixer (DSMHTM) is the core of our ADSFormer for exploiting the critical information from multiple token mixing heads well. Thus, we first introduce it in this subsection. From the detailed structure in Fig. 1, the proposed DSMHTM consists of three main parts: a standard multi-head token mixer (MHTM) to obtain multiple head features, a channel-selective block to adaptively adjust the weights of different head features in the channel dimension and a head-selective block to adaptively calibrate the importance of different head features in the head dimension. By comprehensively considering both the channel and head aspects, the important and discriminative features can be selectively concentrated and integrated to facilitate the performance of our ADSFormer.

Given an input feature Z, we first leverage a standard multi-head token mixer (such as multi-head self-attention [57] or adaptive Fourier neural operators [32] used in our experiment) to obtain multiple head features. Suppose that the number of heads in the token mixer is M, the obtained features can be denoted as $O = [S_1^T, S_2^T, \ldots, S_M^T] \in \mathbb{R}^{M \times D \times T}$, where $S_m (m = 1, \cdots M)$ is the output feature of the m-th token mixing head, D and T are the dimensions of S_m. Unlike the vanilla Transformer, which simply concatenates the multiple features in O and treats them equally, our DSMHTM leverages two selective blocks to aggregate features of different heads by adaptively adjusting the feature weights in both the channel and head dimensions. Next, the details of each selective block are described.

Channel-Selective Block: Given $O = [S_1^T, S_2^T, \ldots, S_M^T]$ which contains features with T time steps and D channels obtained by M token mixing heads, the aim of a channel-selective block is to assign different weights to each S_m $(m = 1, \cdots M)$ in the channel dimension, so that the discriminative features can be emphasized. As shown in the left part of Fig. 1, the M mixing features are first fused by an elementwise addition operation as follows:

$$F = \sum_{m=1}^{M} S_m^T \tag{1}$$

Then, the integrated feature $F \in \mathbb{R}^{D \times T}$ is input to a fully connected layer followed by a GELU to obtain the refined feature $feats \in \mathbb{R}^{D \times T}$, and the channelwise statistics $C_1 \in \mathbb{R}^{D \times 1}$ of *feats* is obtained by a global average pooling (GAP) along the temporal dimension, which can be fomulated as follows:

$$feats = GELU\left(FC_1(F)\right) \tag{2}$$

$$C_1 = GAP(feats) = \frac{1}{T}\sum_{t=1}^{T} feats\{:, t\} \tag{3}$$

where FC_1 is the parameter matrix of the fully connected layer, GELU denotes the activation function and $feats\{:, t\}$ is the t-th column of *feats*.

Next, C_1 passes through another fully connected layer and a GELU activation function to produce $C_2 \in \mathbb{R}^{D \times M}$, which contains M new features. That is:

$$C_2 = GELU\left(FC_2\left(C_1\right)\right) \tag{4}$$

where FC_2 is also a parameter matrix of the fully connected layer.

Finally, a Softmax function is applied along the channel dimension (i.e., column) of C_2 to obtain the selective maps (i.e., weights) for each channel in different heads as follows:

$$C_{map} = Softmax\left(C_2\right) \tag{5}$$

Since the element values in the *m*-th column $(m = 1, \cdots M)$ of C_{map} indicate the importance of feature channels in the *m*-th head, the final output of the channel-selective block can be obtained by:

$$C = \sum_{m=1}^{M} C_{map}\{:, m\} \otimes S_m^T \tag{6}$$

where $C_{map}\{:, m\}$ is the *m*-th column in C_{map} and $\otimes$ denotes the multiplication between the elements in $C_{map}\{:, m\}$ and their corresponding channels in S_m^T.

Head-Selective Block: The channel selective block can indicate the importance of channels in each head, but the different characteristics of multiple heads are ignored. Thus, it is beneficial to propose a selective mechanism to adjust the importance of different head features. The structure of the head-selective block in our DSMHTM is demonstrated in the right part of Fig. 1. Like the channel-selective block, we first obtain the refined $feats \in \mathbb{R}^{D\times T}$ by adding the *M* head features together, followed by a fully connected layer with GELU activation function. Here, the parameter values in the fully connected layer are shared with the channel-selective block to reduce the number of parameters in our method. Then, *M* 1D convolution operations with the ReLU activation function are applied along the channel dimension of *feats* to obtain the new feature $H_1 \in \mathbb{R}^{M\times T}$ as follows:

$$H_1 = ReLU\left(W_{M\times D}^{1D} * feats\right) \tag{7}$$

where $W_{M\times D}^{1D}$ is a matrix containing *M* 1D convolutional kernels, * denotes the convolution and ReLU is activation function. In H_1, each row corresponds to a token mixing head with features of *T* time steps. Finally, we also apply the Softmax function to each column of H_1 to obtain the head selective map H_{map} in (8) and adopt H_{map} to obtain the output of the head-selective block via (9).

$$H_{map} = Softmax\left(H_1\right) \tag{8}$$

$$H = \sum_{m=1}^{M} H_{map}\{m, :\} \otimes S_m^T \tag{9}$$

where $H_{map}\{m, :\}$ is the *m*-th row in H_{map}. $\otimes$ denotes the multiplication between the elements in $H_{map}\{m, :\}$ and their corresponding time step features in S_m^T.

Once *C* and *H* have been obtained by the channel-selective and head-selective blocks, the final features of the proposed DSMHTM can be achieved by:

$$output = C + H \tag{10}$$

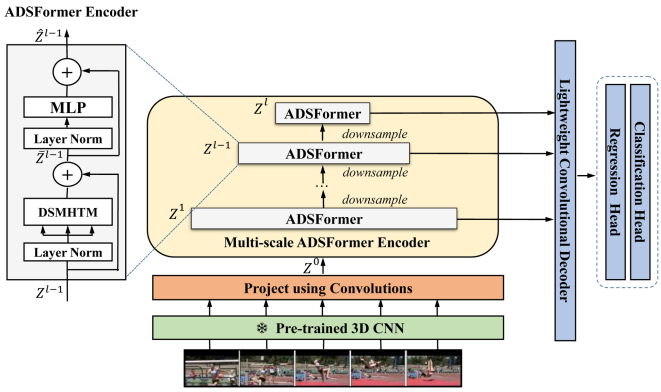

Fig. 2. Overview of ADSFormer. It first extracts features from video using a pretrained CNN. Then, the backbone encodes the features using ADSFormer with DSMHTM to form a multi-scale feature pyramid. Finally, lightweight classification and regression heads decode the feature pyramid into action candidates for each input moment.

C. Method Overview

Based on the dual selective multi-head token mixer (DSMHTM) proposed in the previous subsection, we can build our adaptive dual selective Transformer (ADSFormer) and construct a one-stage network architecture for the TAL problem. In Fig. 2, a high-level overview of our model is provided. Here, we follow other works [24], [25], [26] to adopt an encoder-decoder network structure in our study. The overall network structure of our method consists of three main parts: a video feature extraction layer, a multi-scale ADSFormer based encoder and a shared lightweight convolutional decoder. First, the video features are extracted using some pretrained CNN models (e.g. I3D [16], SlowFast [17], TSP [58]). Then, the ADSFormer is followed with a convolutional layer to form a multi-scale feature pyramid with different resolutions, so that the actions with various temporal lengths can be encoded. Finally, actions are decoded into sequence labels by lightweight classification and regression networks. The main components of our ADSFormer will be described in the following subsections.

D. Encoder With ADSFormer

In our method, the feature *X* obtained from the input video is first encoded into a multi-scale temporal feature pyramid $Z = \{Z^1, Z^2, \ldots, Z^L\}$, where *L* is the number of ADSFormer layers. The encoder has two main components: (1) a convolutional projection layer using a simple convolutional network to capture the local information of the input features and (2) a multi-scale Transformer network with the DSMHTM module called ADSFormer to obtain features of different levels in pyramid *Z*.

1) Projection Using Convolutions: According to some studies [24], convolutional networks can facilitate the learning of local contexts for time series data and help to stabilize the Transformer training. Thus, we utilize a 1D CNN as the projection layer *E* to embed the video feature *X* into a *D*-dimensional space. The projection layer consists of two 1D CNN layers followed by ReLU as the activation function. Formally, the feature projection

layer is described as follows:

$$Z^0 = [E(x_1), E(x_2), \cdots E(x_T)] \tag{11}$$

where $E(x_i) \in \mathbb{R}^D$ is the projected feature of x_i. $Z^0 \in \mathbb{R}^{D\times T}$ is the embedding of *X* in *D*-dimensional feature space.

2) Multi-Scale ADSFormer Encoder: The left side of Fig. 2 shows the structure of the ADSFormer. Compared with the standard Transformer in ActionFormer [24], we can see that the key difference between them is the dual selective multi-head token mixer (DSMHTM), which was introduced in Section III-B. The encoder architecture of our method consists of *L* ADSFormer layers. Each ADSFormer layer has two sublayers with an alternating array of DSMHTM and MLP blocks. Similar to standard Transformers, LayerNorm(LN) [59] is also utilized before each DSMHTM or MLP block for normalization, and a residual connection is added to combine the input and output of each block. In our ADSFormer, the GELU is employed as the activation function for MLP. Thus, suppose that the embedded video feature Z^0 in (11) is input to an ADSFormer, the output can be obtained by:

$$\bar{Z}^0 = \lambda\, DSMHTM\left(LN\left(Z^0\right)\right) + Z^0 \tag{12}$$

$$\hat{Z}^0 = \bar{\lambda} MLP\left(LN\left(\bar{Z}^0\right)\right) + \bar{Z}^0 \tag{13}$$

where $\bar{Z}^0$ and $\hat{Z}^0$ denote the features learned by the DSMHTM and MLP blocks, LN is the LayerNorm operation, λ and $\bar{\lambda}$ are two learnable scaling parameters.

As shown in Fig. 2, the multiple ADSFormers in our encoder are stacked as a pyramidal structure to capture action features at different temporal scales. Thus, a downsampling step is added after each ADSFormer for multi-scale feature extraction. Based on (12) and (13), the features in the *l*-th $(l = 1\cdots L)$ level of the pyramid are obtained by:

$$\bar{Z}^{l-1} = \lambda^{l-1} DSMHTM\left(LN\left(Z^{l-1}\right)\right) + Z^{l-1} \tag{14}$$

$$\hat{Z}^{l-1} = \bar{\lambda}^{l-1} MLP\left(LN\left(\bar{Z}^{l-1}\right)\right) + \bar{Z}^{l-1} \tag{15}$$

$$Z^l = downsamping\left(\hat{Z}^{l-1}\right) \tag{16}$$

where $downsampling$ 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 ADSFormers layers as $Z = \{Z^1, Z^2, \cdots Z^L\}$.

E. Decoder

The decoder, which consists of both regression and classification, learns to predict sequence label for every moment using multi-scale feature pyramid $Z = \{Z^1, Z^2, \cdots Z^L\}$.

Since TriDet [26] has proven that the relative probability distribution of boundaries is important for action localization [26], our study follows the Trident-head decoder structure in it for action boundary regression. Specifically, Trident-head consists of three components: a start head, an end head, and a center-offset head, which are designed to locate the start boundary, end boundary, and temporal center of an action, respectively. Given the feature pyramid *Z*, Trident-head decoder uses 3 layers of 1D convolutions to obtain the feature sequences $F_S \in \mathbb{R}^T$, $F_E \in \mathbb{R}^T$, and $F_C \in \mathbb{R}^{T\times 2\times(B+1)}$, where *B* is the number of neighbors for boundary prediction, F_S and F_E are the probabilities of each time step as the start and end boundaries of an action, respectively, and F_C is used to predict the probabilities of its *B* neighboring time steps as the start and end boundaries if the time step *t* is the center of an action.

According to TriDet [26], the boundary distance can be modeled by combining the outputs of the boundary and center-offset heads. For example, the distance between time step *t* and the start boundary of the action can be obtained by the expectation of its *B* neighboring time steps as follows:

$$d_{st} = E_{b\sim\widetilde{P}_{st}}[b] \approx \sum_{b=0}^{B}\left(b\widetilde{P}_{stb}\right) \tag{17}$$

where $\widetilde{P}_{st}$ is a relative probability that can be calculated by:

$$\widetilde{P}_{st} = Softmax\left(F_S^{[(t-B):t]} + F_C^{t,0}\right) \tag{18}$$

where $F_S^{[(t-B):t]} \in \mathbb{R}^{B+1}$ is the probabilities of the left *B* neighbors of time step *t* being the start boundary estimated by F_S and $F_C^{t,0} \in \mathbb{R}^{B+1}$ is the center offset solely predicted at time step *t*.

The distance between a time step and the end boundary of the action can be estimated in a similar manner as in (19) and (20).

$$d_{et} = E_{b\sim\widetilde{P}_{et}}[b] \approx \sum_{b=0}^{B}\left(b\widetilde{P}_{etb}\right) \tag{19}$$

$$\widetilde{P}_{et} = Softmax\left(F_E^{[t:(t+B)]} + F_C^{t,1}\right) \tag{20}$$

For more details about Trident-head decoder, the readers can refer to [26].

In addition to the boundary regression by Trident-head, we also need to predict the action label of each time step across all levels in the feature pyramid. This can also be realized by a lightweight convolutional network. Here, a network with 3 layers of 1D convolutions, layer normalization and a ReLU activation function is adopted. The sigmoid function is further employed to obtain the classification probabilities of the time steps.

Finally, it is important to note that the decoder weights are shared among different feature levels within the multi-scale feature pyramid *Z*, which can reduce the number of parameters.

F. Loss Function

After decoding the feature pyramid *Z*, the model prediction $\hat{o}_t^l = (\hat{s}_t^l, \hat{e}_t^l, \hat{a}_t^l)$ of each time step in the feature pyramid layer $l(l = 1, \cdots L)$ can be obtained, where $\hat{a}_t^l$ is the predicted action label at time step *t*, $\hat{s}_t^l$ and $\hat{e}_t^l$ indicate the start and end boundaries of the action in time step *t*, which can be calculated by:

$$\hat{s}_t = \left(t - \hat{d}_{st}^l\right) \times 2^{l-1} \tag{21}$$

$$\hat{e}_t = \left(t + \hat{d}_{et}^l\right) \times 2^{l-1} \tag{22}$$

Following other studies [24], [25], [26], the Focal loss [60] and IoU loss [61] are employed to supervise classification and regression outputs respectively. The overall loss function is then defined as follows:

$$L = \frac{1}{N_{pos}} \sum_{l,t} 1_{\{c_t^l > 0\}} \left(\sigma_{IoU} \mathcal{L}_{cls} \left(a_t^l, \hat{a}_t^l \right) + \mathcal{L}_{reg} \left(\left[s_t^l, e_t^l \right], \left[\hat{s}_t^l, \hat{e}_t^l \right] \right) \right) + \frac{1}{N_{neg}} \sum_{l,t} 1_{\{c_t^l = 0\}} \mathcal{L}_{cls} \left(a_t^l, \hat{a}_t^l \right) \quad (23)$$

where σ_{IoU} is the temporal IoU between the predicted action and ground truth. $\mathcal{L}_{cls}$ is the Focal loss for handling imbalanced samples in different action categories. $\mathcal{L}_{reg}$ denotes IoU-based regression loss and is applied only when the function $1_{\{c_t^l > 0\}}$ indicates that the current time step t is a positive sample within an action. N_{pos} and N_{neg} are the numbers of positive and negative samples, respectively. In our study, center sampling [62] is adopted to determine the positive samples. That is, the time steps around the center of an action are labeled as positive, and all the others are considered as negative. The loss function $\mathcal{L}$ is applied to the decoder outputs of all levels in the multi-scale feature pyramid Z during training.

IV. Experiments

A. Model Variants

In this study, two representative techniques are adopted as token mixers in our ADSFormer. One is multi-head self-attention (MHSA) and the other is adaptive Fourier neural operators (AFNO) [32]. Thus, the proposed methods with the two mixers are denoted as ADSFormer_SA and ADSFormer_AFNO, respectively. We opt for the MHSA because it is a classical multi-head token mixer and has been widely employed in TAL and other tasks [23], [24]. For AFNO, it is a novel Fourier transform based multi-head token mixer that exhibits high efficiency and effectiveness.

B. Datasets

We conduct experiments on four datasets, including THUMOS14 [33], MultiTHUMOS [35], EPIC-KITCHENS-100 [34] and ActivityNet-1.3 [36]. 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.

MultiTHUMOS contains dense, multilabel, frame-level action annotations for 30 hours across 400 videos in THUMOS14 dataset. It consists of 38,690 annotations of 65 action classes, with an average of 1.5 labels per frame and 10.5 action classes per video. MultiTHUMOS poses a greater challenge for TAL than THUMOS14 since the average number of distinctive action categories per video is significantly larger in it.

EPIC-KITCHENS-100 is a large dataset with egocentric action videos, which has two sub-tasks: *noun* localization (e.g. knife) and *verb* localization (e.g. take knife). The dataset contains 100 hours of video from 700 sessions capturing cooking activities in different kitchens. There are 495 and 138 videos with 67,217 and 9,668 action instances for training and testing, and the number of actions in it is almost 10 times larger than THUMOS14. Furthermore, the first person perspective videos in EPIC-KITCHENS-100 also include significant camera motion, rendering temporal action localization a more difficult challenge.

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.

C. Evaluation Metric

To compare our ADSFormer with the existing methods, 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, MultiTHUMOS and EPIC-KITCHENS-100 datasets, we report the results at IoU thresholds [0.3:0.7:0.1], [0.1:0.9:0.1] and [0.1:0.5:0.1]. For ActivityNet-1.3 dataset, we report the results at IoU thresholds [0.5,0.75,0.95] and the average mAP is computed at [0.5:0.95:0.05].

D. Implementation Details

In our experiment, position embedding is disabled by default except for ActivityNet-1.3 dataset. Following ActionFormer [24], we employ the local mechanism in our ADSFormer_SA by limiting the MHSA within a local window, and the window size is shared in different levels of feature pyramid. For ADSFormer_AFNO, the global token mixing strategy is employed due to AFNO's efficiency. The AdamW [63] is adopted for model optimization. The learning rate is updated with cosine annealing schedule [64]. We conduct our experiments on Python3.8, Pytorch2.0 and CUDA11.8 on a single NVIDIA RTX A6000 GPU. The output feature dimension of projection layer is 256 on ActivityNet-1.3 dataset and 512 on the other datasets. The number of feature levels is set as L=6 for all datasets while the values of other parameters in our model vary across different datasets, therefore we will elaborate on the specifics of each dataset in next subsection.

E. Results and Analysis

Results on THUMOS14 On THUMOS14 dataset, the number of heads and local window size in ADSFormer_SA are

TABLE I
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON THUMOS14 DATASET

Type	Method	Feature	tIoU					
			0.3	0.4	0.5	0.6	0.7	Avg.
Two-Stage	G-TAD [19] (CVPR 2020)	TSN	54.5	47.6	40.3	30.8	23.4	39.3
	BC-GNN [40] (ECCV 2020)	TSN	57.1	49.1	40.4	31.2	23.1	40.2
	TAL-MR [39] (ECCV 2020)	I3D	53.9	50.7	45.4	38.0	28.5	43.3
	DBG [9] (AAAI 2020)	TSN	57.8	49.4	39.8	30.2	21.7	39.8
	TSA-Net [38] (ICME 2020)	P3D	61.2	55.9	46.9	36.1	25.2	45.1
	A2Net [14] (TIP 2020)	I3D	58.6	54.1	45.5	32.5	17.2	41.6
	PBRNet [66] (AAAI 2020)	I3D	58.5	54.6	51.3	41.8	29.5	47.1
	P-GCN [41] + TSP [58] (ICCV 2021)	R(2+1)1 D	69.1	63.3	53.5	40.4	26.0	50.5
	MUSES [12] (CVPR 2021)	I3D	68.9	64.0	56.9	46.3	31.0	53.4
	TCANet [42] (CVPR 2021)	TSN	60.6	53.2	44.6	36.8	26.7	44.3
	BMN-CSA [43] (ICCV 2021)	TSN	64.4	58.0	49.2	38.2	27.8	47.7
	ContextLoc [67] (ICCV 2021)	I3D	68.3	63.8	54.3	41.8	26.2	50.9
	VSGN [20] (ICCV 2021)	TSN	66.7	60.4	52.4	41.0	30.4	50.2
	RTD-Net [47] (ICCV 2021)	I3D	68.3	62.3	51.9	38.8	23.7	49.0
	Disentangle [68] (AAAI 2022)	I3D	72.1	65.9	57.0	44.2	28.5	53.5
	SAC [69] (TIP 2022)	I3D	69.3	64.8	57.6	47.0	31.5	54.0
	K.Xia et al. [1] (TMM 2023)	I3D	81.6	78.4	72.2	59.0	44.5	67.1
One-Stage	AFSD [13] (CVPR 2021)	I3D	67.3	62.4	55.5	43.7	31.1	52.0
	TAGS [45] (ECCV 2022)	I3D	68.6	63.8	57.0	46.3	31.8	52.8
	HTNet [70] (SMC 2022)	I3D	71.2	67.2	61.5	51.0	39.3	58.0
	TadTR [48] (TIP 2022)	I3D	74.8	69.1	60.1	46.6	32.8	56.7
	ReAct [49] (ECCV 2022)	TSN	69.2	65.0	57.1	47.8	35.6	55.0
	TALLFormer [23] (ECCV 2022)	Swin Transformer	76.0	—	63.2	—	34.5	59.2
	ActionFormer [24] (ECCV 2022)	I3D	82.1	77.8	71.0	59.4	43.9	66.8
	TemporalMaxer [25] (arxiv 2023)	I3D	82.8	78.9	71.8	60.5	44.7	67.7
	TriDet [26] (CVPR 2023)	I3D	83.6	**80.1**	72.9	62.4	47.4	69.3
	ADSFormer_SA	I3D	82.9	79.9	**73.4**	62.8	**47.8**	69.4
	ADSFormer_AFNO	I3D	**84.4**	80.0	73.1	**62.9**	46.9	**69.5**
	ActionFormer [24] (ECCV 2022)	VideoMAE V2	84.0	79.6	73.0	63.5	47.7	69.6
	TriDet [26] (CVPR 2023)	VideoMAE V2	84.8	80.0	73.3	63.8	48.8	70.1
	ADSFormer_SA	VideoMAE V2	85.0	80.7	**73.9**	**64.1**	49.7	70.7
	ADSFormer_AFNO	VideoMAE V2	**85.3**	**80.8**	**73.9**	64.0	**49.8**	**70.8**

The best results of different tIoU thresholds are highlighted in bold.

set to 8 and 37, respectively. The initial learning rate is set to 0.000125, with 18 epochs for warmup and 40 epochs for training. For ADSFormer_AFNO, the model is configured with 16 heads, and the initial learning rate is set to 0.0001 with a warmup period of 20 epochs followed by 40 epochs for training. For both ADSFormer_SA and ADSFormer_AFNO, the number of neighbors B in Trident-head [26] is set to 12, the mini-batch size is set to 2, and a weight decay of 0.025 is used. Like most of other approaches, we employ the I3D [16] and VideoMAE V2 [65] as our video feature extractor. Table I illustrates that our ADSFormer_SA and ADSFormer_AFNO methods achieve the average mAPs of 69.4% and 69.5% when utilizing I3D features, which outperform other methods. Furthermore, with VideoMAE V2 features, ADSFormer_SA and ADSFormer_AFNO exhibit even stronger performance, achieving 70.7% and 70.9% mAP, respectively. These results surpass the performance of other methods with the same VideoMAE V2 features. Moreover, we can also observe that the performance of ADSFormer_AFNO is marginally better than that of ADSFormer_SA. This is attributed to the ability of the AFNO to effectively capture global dependencies among the frame features. For two-stage approaches, their performances are generally inferior to those of recent one-stage methods. This may be due to the inaccurate proposals generated by them. Among the one-stage methods, it is clear that the models based on the Transformer architecture or its variants (such as TadTR [48], ReAct [49], TALLFormer [23], ActionFormer [24], TemporalMaxer [25] and TriDet [26]) outperform others. This phenomenon demonstrates the advantage of Transformer with a token mixer in the TAL task. Nevertheless, through introduction of dual selective mechanisms to exploit discriminative information from multi-head features, the two variants of the

TABLE II
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON MULTITHUMOS DATASET

Method	Feature	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	Avg.
PDAN [71] (WACV 2021)	I3D(only RGB)	-	-	-	-	-	-	-	-	-	17.3
MLAD [72] (CVPR 2021)	I3D(only RGB)	-	-	-	-	-	-	-	-	-	14.2
MS-TCT [73] (CVPR 2022)	I3D(only RGB)	-	-	-	-	-	-	-	-	-	16.2
PointTAD [74] (NeurIPS 2022)	I3D(only RGB)	42.3	39.7	35.8	30.9	24.9	18.5	12.0	5.6	1.4	23.5
ActionFormer [24] (ECCV 2022)	I3D(only RGB)	-	46.4	-	-	32.4	-	15.0	-	-	28.6
TemporalMaxer [25] (arxiv 2023)	I3D(only RGB)	49.1	47.5	44.3	39.4	33.4	26.5	17.4	9.1	2.24	29.9
TriDet [26] (CVPR2023)	I3D(only RGB)	-	49.1	-	-	34.3	-	17.8	-	-	30.7
ADSFormer_SA	I3D(only RGB)	**56.9**	55.1	**52.2**	47.4	40.6	**31.9**	**21.9**	**11.5**	**3.1**	**35.6**
ADSFormer_AFNO	I3D(only RGB)	56.8	**55.3**	51.6	**47.5**	**40.9**	31.8	**21.9**	11.3	**3.1**	**35.6**
TriDet [26] (CVPR2023)	I3D	-	55.7	-	-	41.0	-	23.5	-	-	36.2
ADSFormer_SA	I3D	63.5	61.6	58.7	54.2	**48.0**	**39.9**	28.4	15.8	4.1	41.6
ADSFormer_AFNO	I3D	**63.9**	**62.3**	**59.3**	**54.8**	**48.0**	39.3	**28.5**	**16.1**	**4.2**	**41.8**
TriDet [26] (CVPR2023)	VideoMAE V2	-	57.7	-	-	42.7	-	24.3	-	-	37.5
ADSFormer_SA	VideoMAE V2	65.8	64.2	61.2	56.4	50.0	41.5	30.9	17.5	4.6	43.5
ADSFormer_AFNO	VideoMAE V2	**66.1**	**64.4**	**61.3**	**56.9**	**51.0**	**42.9**	**31.7**	**18.2**	**4.7**	**44.1**

The best results of different tIoU thresholds are highlighted in bold.

proposed ADSFormer surpasses other Transformer-based methods.

Results on MultiTHUMOS: On MultiTHUMOS dataset, we also use I3D and VideoMAE V2 to extract video features. The specific settings of the experimental parameters for this dataset are consistent with those used for THUMOS14. Table II provides a performance comparison between our ADSFormers (ADSFormer_SA and ADSFormer_AFNO) and several recent state-of-the-art methods. We find that our methods outperform other Transformer-based approaches by a large margin. For instance, the average mAPs achieved by ADSFormer_SA with I3D (only RGB) features are 7%, 5.5% and 4.9% greater than those achieved by ActionFormer [24], TemporalMaxer [25] and TriDet [26], respectively. Moreover, the performance of our methods can be further improved by the use of the more sophisticated feature extractor VideoMAE V2.

Results on EPIC-KITCHENS-100: On this dataset, we follow previous studies [24], [25] to leverage the pretrained SlowFast network [17] for video feature extraction. In our ADSFormer_SA model, we set the number of heads, local window size, and B in Trident-head to 4, 19, and 16, respectively. The learning rate during the initial phase is set to 0.0001, followed by a warm-up period of 5 epochs and a main training period of 20 epochs. For ADSFormer_AFNO, we set both the number of heads and B in Trident-head to 16. In the initial phase of model training, the learning rate is initialized to 0.0001, followed by a warm-up period of 22 epochs and a main training period of 25 epochs. In both ADSFormer_SA and ADSFormer_AFNO, the mini-batch size is consistently set to 2, and a weight decay of 0.025 is applied throughout the entire training process.

From the results in Table III, our ADSFormer_SA achieves average mAPs ([0.1:0.1:0.5]) of 25.7% and 24.1% for *verb* and *noun* subtasks, respectively, which are lower than the results on the THUMOS14 and MultiTHUMOS datasets. Similarly, the average mAPs of ADSFormer_AFNO are also inferior to the experimental results on previous datasets. As we have mentioned, the performance reduction in this dataset is attributed to the severe camera motion of egocentric perspective videos. However, the proposed ADSFormers (ADSFormer_SA and ADSFormer_AFNO) still outperform the other comparison methods and obtain the state-of-the-art results for both *verb* and *noun* TAL subtasks.

TABLE III
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON EPIC-KITCHENS-100 DATASET

Method	Task	0.1	0.2	0.3	0.4	0.5	Avg.
BMN [10] (ICCV 2019)	*verb*	10.8	8.8	8.4	7.1	5.6	8.4
G-TAD [19] (CVPR 2020)		12.1	11.0	9.4	8.1	6.5	9.4
ActionFormer [24] (ECCV 2022)		26.6	25.4	24.2	22.3	19.1	23.5
TemporalMaxer [25]		27.8	26.6	25.3	23.1	19.9	24.5
TriDet [26] (CVPR 2023)		28.6	27.4	26.1	24.2	20.8	25.4
ADSFormer_SA		**28.8**	**27.9**	**26.4**	**24.3**	20.9	**25.7**
ADSFormer_AFNO		**28.8**	**27.9**	**26.4**	**24.3**	**21.0**	**25.7**
BMN [10] (ICCV 2019)	*noun*	10.3	8.3	6.2	4.5	3.4	6.5
G-TAD [19] (CVPR 2020)		11.0	10.0	8.6	7.0	5.4	8.4
ActionFormer [24] (ECCV 2022)		25.2	24.1	22.7	20.5	17.0	21.9
TemporalMaxer [25]		26.3	25.2	23.5	21.3	17.6	22.8
TriDet [26] (CVPR 2023)		27.4	26.3	24.6	22.2	18.3	23.8
ADSFormer_SA		27.7	26.7	**24.7**	22.5	18.9	24.1
ADSFormer_AFNO		**27.8**	**26.8**	24.6	**22.6**	**19.0**	**24.2**

The best results of different tIoU thresholds are highlighted in bold.

Results on ActivityNet-1.3: On ActivityNet-1.3 dataset, TSP R(2+1)D [58] is utilized as the pretrained model to extract video features, similarly to some recent studies [24], [26], [48]. We trained our ADSFormer_SA model with 8 self-attention heads and a local window size of 7 for 10 warm-up epochs, followed by 20 training epochs. The weight decay is set to 0.05. For ADSFormer_AFNO, we employ 16 heads to train the model with 10 warm-up epochs and 15 main training epochs, and set the weight decay to 0.04. In both ADSFormer_SA and ADSFormer_AFNO, the initial learning rate is set to 0.001, and the parameter B in Trident-head [26] is set to 12. During model training, we use a mini-batch size of 16. Following ActionFormer [24], the absolute position encoding is adopted for this

TABLE IV
COMPARISON WITH THE STATE-OF-THE-ART METHODS ON ACTIVITYNET1.3 DATASET

Type	Method	Feature	tIoU			Avg.
			0.5	0.75	0.95	0.5:0.95:0.05
Two-Stage	PGCN [41] (ICCV 2019)	I3D	48.3	33.2	3.3	31.1
	BMN [10] (ICCV 2019)	TSN	50.1	34.8	8.3	33.9
	G-TAD [19] (CVPR 2020)	TSN	50.4	34.6	9.0	34.1
	VSGN [20] (ICCV 2021)	I3D	52.3	35.2	8.3	34.7
	TCANet [42] + BMN [10] (CVPR 2021)	TSN	52.3	36.7	6.9	35.5
	TAPP [3] (TMM 2022)	I3D	53.1	34.3	9.1	34.4
	K.Xia et al. [1] (TMM 2023)	I3D	54.2	35.1	7.3	34.2
One-Stage	PBRNet [66] (AAAI 2020)	I3D	54.0	35.0	9.0	35.0
	AFSD [13] (CVPR 2021)	I3D	52.4	35.2	6.5	34.3
	ReAct [49] (ECCV 2022)	TSN	49.6	33.0	8.6	32.6
	TadTR [48] (TIP 2022)	TSN	51.3	35.0	9.5	34.6
	TadTR [48] (TIP 2022)	R(2+1)D	53.6	37.5	**10.5**	36.8
	TALLFormer [23] (ECCV 2022)	Swin	54.1	36.2	7.9	35.6
	ActionFormer [24] (ECCV 2022)	R(2+1)D	54.7	37.8	8.4	36.6
	TriDet [26] (CVPR 2023)	R(2+1)D	54.7	38.0	8.4	36.8
	ADSFormer_SA	R(2+1)D	**55.3**	**38.4**	8.3	37.0
	ADSFormer_AFNO	R(2+1)D	**55.3**	**38.4**	8.4	**37.1**

The best results of different tIoU thresholds are highlighted in bold.

TABLE V
COMPARISON OF COMPUTATIONAL COSTS ON THUMOS14

Method	mAP			GMACs			Latency(ms)
	0.3	0.7	Avg.	encoder	decoder	All	
ActionFormer	82.1	43.9	66.8	30.68	14.42	45.10	103.5
TemporalMaxer	82.8	44.7	67.7	9.07	14.42	23.49	60.6
TriDet	83.6	47.4	69.3	14.54	29.19	43.73	82.7
ADSFormer_SA	82.9	47.8	69.4	30.71	29.19	59.90	118.1
ADSFormer_AFNO	84.4	46.9	69.5	14.49	29.19	43.68	81.8

dataset. From the results in Table IV, we can see that the performances obtained by the two ADSFormer variants are superior to those of two-stage methods and comparable with those of state-of-the-art one-stage approaches.

Statistical Significance Test: From the experimental results in Tables I–IV, we can see the improvement of our ADSFormers over state-of-the-art TriDet [26] is limited in some cases. Thus, a one-tailed Wilcoxon rank sum test is employed to provide more compelling evidence of the effectiveness of our proposed method. In this test, the null hypothesis posits that ADSFormer exhibits no significantly different performance compared with TriDet, while the alternative hypothesis suggests that the performance of ADSFormer is superior to TriDet. For example, if we want to compare the performance of ADSFormer_SA with that of TriDet (ADSFormer_SA vs. TriDet), the null and alternative hypotheses can be defined as $H_0 : M_{ADSFormer\_SA} = M_{TriDet}$ and $H_1 : M_{ADSFormer\_SA} > M_{TriDet}$, where $M_{ADSFormer\_SA}$ and M_{TriDet} are the medians of average mAPs obtained by ADSFormer_SA and TriDet for test samples in all datasets. In our experiments, we set the significance level as 1%. The *p-values* obtained from the comparisons ADSFormer_SA vs. TriDet, and ADSFormer_AFNO vs. TriDet are 4.926e-3 and 8.94e-4, respectively. This indicates our ADSFormer_SA and ADSFormer_AFNO both significantly outperform TriDet.

Computational Complexity: We conduct a comparative analysis of the computational complexity of different Transformer architecture models in Table V. Specifically, we present the GMACs required for both encoders and decoders in various methods when processing a video with 2304 time steps from the THUMOS14 dataset. Moreover, the inference latency of each method is also provided. Table V shows that although our ADSFormer_SA outperforms ActionFormer, TemporalMaxer and TriDet, its computational overhead is higher than that of the other methods. This is attributed to the high computational complexity of the MHSA-based token mixer and the additional computations introduced by dual selective blocks. Nevertheless, once the AFNO is adopted in our ADSFormer as a token mixer, the computational burden and latency of ADSFormer_AFNO can be effectively reduced to be slightly lower than those of ActionFormer and TriDet. This is because the AFNO implements multi-head token mixing in the Fourier domain, which is more efficient than self-attention in the time domain. Here, it is noteworthy that TemporalMaxer exhibits the lowest computational complexity and latency among all the comparison methods because it merely utilizes a single max-pooling in its encoder. However, its performance is inferior to that of TriDet and our ADSFormers.

F. Ablation Study

We perform various ablation studies on the THUMOS14 dataset to verify the effectiveness of ADSFormer. To better understand the role of DSMHTM in our ADSFormer, we first analyze the effectiveness of dual selective mechanisms in DSMHTM. Then, the features learned by the ADSFormer encoder are compared with those of other approaches. Next, the influences of several parameters (such as the local window size and number of self-attention heads) on our ADSFormer_SA and ADSFormer_AFNO are evaluated. Finally, we conduct an error diagnosis [75] to further examine the results obtained by our method.

Effectiveness of Selective Blocks: To fully verify the effectiveness of dual selective mechanisms (i.e., channel-selective and head-selective) in DSMHTM, we perform an ablation experiment on our ADSFormer_SA and ADSFormer_AFNO using THUMOS14 and other datasets. According to the results shown in Table VI, two observations can be made. First, compared with the baseline without any selective mechanism, the utilization of channel-selective or head-selective block can improvement the mAP. This indicates that selecting features in either the channel or head dimension is beneficial for exploiting discriminative information. Second, employing both selective blocks leads to a remarkable boost in the mAP. For instance, in the case of ADSFormer_SA with I3D features on THUMOS14, the average mAP increases notably from 68.4% to 69.4% when both the channel and head-selective blocks are adopted. This means that the combination of channel and head-selective mechanisms is more effective at promoting the TAL performance than using either one alone. Thus, we can conclude that both channel and head-selective blocks play essential roles in the proposed DSMHTM.

Moreover, we incorporate the channel and head-selective blocks into RTD-Net [47] and TALLFormer [23], which also leverage the self-attention-based Transformer for TAL, to further test the effectiveness of dual selective mechanism proposed in this paper. In our experiments, we leverage the code provided

TABLE VI
ABLATION EXPERIMENTS OF DSMHTM ON DIFFERENT DATASETS

	Method	THUMOS14		MultiTHUMOS			EPIC-KITCHENS-100		ActivityNet-1.3
		I3D	VideoMAE V2	onlyRGB	I3D	VideoMAE V2	verb	noun	R(2+1)D
Without DSMHTM	ADSFormer_SA	68.4	69.9	35.2	41.2	43.2	25.1	23.1	36.7
	ADSFormer_AFNO	68.5	70.1	35.2	41.5	43.7	25.2	23.4	36.7
With Channel-Selective	ADSFormer_SA	68.9	70.5	35.3	41.4	43.4	25.6	23.8	36.9
	ADSFormer_AFNO	68.8	70.4	35.4	41.7	44.0	25.4	23.9	36.8
With Head-Selective	ADSFormer_SA	68.8	70.3	35.2	41.3	43.4	25.3	23.7	36.8
	ADSFormer_AFNO	68.9	70.3	35.3	41.6	43.9	25.5	23.7	36.8
With DSMHTM	ADSFormer_SA	69.4	70.7	35.6	41.6	43.5	25.7	24.1	37.0
	ADSFormer_AFNO	69.5	70.8	35.6	41.8	44.1	25.7	24.2	37.1

TABLE VII
ABLATION EXPERIMENTS OF DSMHTM ON RTD-NET AND TALLFORMER

Method	mAP			
	without DSMHTM	With Channel-Selective	With Head-Selective	with DSMHTM
RTD-Net	49.0	49.1	49.1	49.2
TALLFormer	59.2	59.4	59.3	59.5

TABLE VIII
COMPARISON OF FEATURES OBTAINED BY DIFFERENT ENCODERS ON THUMOS14 DATASET

Decoder	Average mAP			
	ActionFormer	Tridet(SGP)	ADSFormer_SA	ADSFormer_AFNO
ActionFormer-head	66.8	67.7	67.8	67.9
Trident-head	68.4	69.3	69.4	69.5

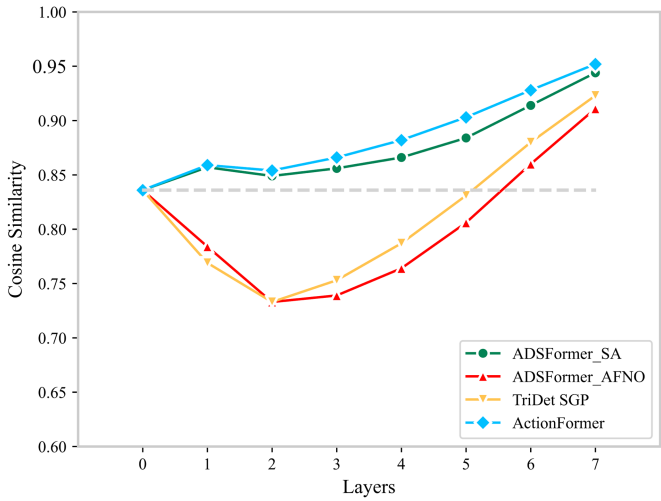

Fig. 3. Average cosine similarity between features obtained by different encoders.

TABLE IX
ABLATION EXPERIMENTS OF PARAMETER SHARING IN DSMHTM ON THUSMOS14

Shared Parameters	Encoder Parameters	Encoder GMACs	Avg.
✓	26.26M	30.71	69.4
x	26.29M	30.74	69.4

by the authors. Rather than modifying the network structure, we seamlessly integrate our proposed channel and head-selective blocks into a multi-head self-attention-based token mixer in the Transformer component of their methods. For efficient training, we freeze the pretrained parameters in these two models but only update those parameters associated with channel and head selective blocks during the training process. From the experimental results in Table VII, we can see that the dual selective mechanism can improve the performance of these two methods.

Effectiveness of the Encoder: To confirm that our proposed DSMHTM method can extract more discriminative information, we compare the features obtained by the ADSFormer_SA and ADSFormer_AFNO encoders with those of ActionFormer and TriDet. To fairly evaluate the features of different encoders, we employ the decoders in ActionFormer and TriDet. Table VIII shows that the average mAPs achieved by ADSFormer_SA and ADSFormer_AFNO are superior to those achieved by the other two methods, which indicates that the features encoded by our method are more discriminative. Here, it is noteworthy that the performance of ActionFormer is worse than that of TriDet and our ADSFormers even when using its own decoder. This may due to it adopts vanilla Transformer as encoder, which combines the multi-head features equally and neglects the distinct characteristics of different self-attention heads. Moreover, we also compare the average cosine similarity between features obtained by different encoders at each level of the pyramid. As shown in Fig. 3, the similarities of the ADSFormer_SA features are lower than those of the ActionFormer features since discriminative information is selected in the channel and head dimensions from the MHSA. However, the feature similarities of our method are still higher than TriDet which utilizes a CNN-based SGP as an encoder. For ADSFormer_AFNO, it leverages operator learning to mix tokens in the Fourier domain, which is mathematically equivalent to a global convolution in time domain. Thus, the feature similarities conducted by it are analogous to CNN based SGP in TriDet. From the experimental results in Table I and Fig. 3, we can see that the performance of a TAL model is not solely determined by the feature similarity and high feature diversity does not always result in high discriminability.

The Influence of Parameter Sharing: We conduct an ablation study to evaluate the influence of parameter sharing in the fully connected layers of channel and head selective blocks. Here, the ADSFormer_SA is utilized as an example. From Table IX, we can see that sharing fully connected layer parameters slightly reduces the number of parameters and GMACs in the encoder of ADSFormer_SA without compromising the performance.

The Influence of Position Embedding: We investigate the impact of position embedding on our method across various datasets in Table X. The experimental results reveal that the position embedding does not bring performance gain on THUMOS14, MultiTHUMOS and EPIC-KITCHENS-100 datasets. This observation is similar with that in ActionFormer [24].

TABLE X
INFLUENCE OF POSITION EMBEDDING ON DIFFERENT DATASETS

	Method	THUMOS14		MultiTHUMOS			EPIC-KITCHENS-100		ActivityNet-1.3
		I3D	VideoMAE V2	onlyRGB	I3D	VideoMAE V2	verb	noun	R(2+1)D
With Position Embedding	ADSFormer_SA	69.2	70.5	35.5	41.4	43.4	25.6	24.0	37.0
	ADSFormer_AFNO	69.4	70.5	35.6	41.6	43.8	25.5	24.0	37.1
Without Position Embedding	ADSFormer_SA	69.4	70.7	35.6	41.6	43.5	25.7	24.1	36.4
	ADSFormer_AFNO	69.5	70.8	35.6	41.8	44.1	25.7	24.2	36.4

TABLE XI
INFLUENCE OF LOCAL WINDOW SIZE AND NUMBER OF HEADS IN ADSFORMER_SA ON THUMOS14 DATASET

	ADSFormer_SA				ActionFormer			
	4	6	8	10	4	6	8	10
9	68.3	67.2	68.7	68.0	66.5	66.5	66.8	66.3
19	67.7	67.7	67.8	68.4	66.8	66.2	66.7	66.3
25	67.8	67.4	67.9	68.3	66.4	66.6	66.3	66.3
37	67.8	67.9	69.4	67.7	66.7	66.7	66.5	66.3
Full	67.9	67.7	67.9	68.4	66.8	66.6	66.5	66.2

TABLE XII
INFLUENCE OF DIFFERENT NUMBER OF HEADS IN ADSFORMER_AFNO ON THUMOS14 DATASET

heads	4	8	16	32
mAP	68.7	67.8	69.5	68.7

TABLE XIII
EVALUATION RESULTS OF DIFFERENT NUMBERS OF PYRAMID LEVELS ON THUMOS14 DATASET

Levels	mAP					
	ADSFormer_SA			ADSFormer_AFNO		
	0.5	0.7	Avg	0.5	0.7	Avg
1	35.8	10.9	36.6	37.5	11.6	38.0
2	58.4	25.3	54.9	58.3	25.3	54.5
3	65.7	39.6	62.6	65.6	37.6	62.4
4	70.0	43.6	65.8	70.4	43.5	66.5
5	70.9	45.1	67.2	71.6	45.8	68.2
6	73.4	47.8	69.4	73.1	46.9	69.5
7	72.0	45.2	68.1	72.8	46.3	68.9

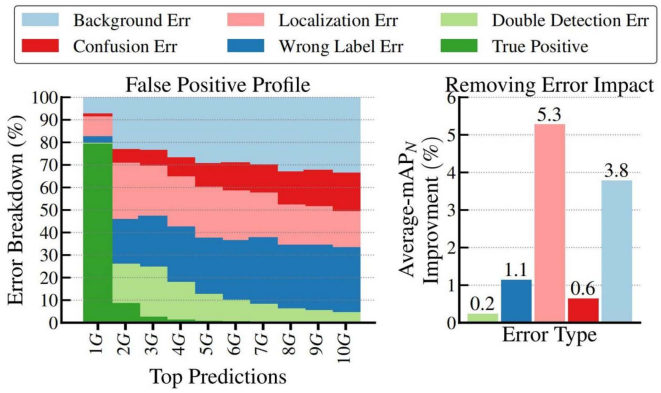

Fig. 4. False positive profile obtained by ADSFormer_SA and the impact of error types.

Conversely, the position embedding exhibits a notable effectiveness on ActivityNet-1.3 dataset. This may be due to the ActivityNet-1.3 dataset is larger and more complex than other datasets. Furthermore, the durations of videos in ActivityNet-1.3 are also generally longer than those in other datasets. Thus, the position embedding can help our method to capture the location information on this dataset, facilitating the precise localization of actions.

Parameter Sensitivity: First, we test the influences of the number of token mixing heads and local window size on the performance of our ADSFormer_SA. In this experiment, the number of self-attention heads is set to 4, 6, 8, and 10 while the local window size is set to 9, 19, 25, 37 and Full, where Full denotes all features in each level of pyramid are employed to compute the self-attention. From the comparison results in Table XI, it can be found that the proposed ADSFormer_SA achieves its optimal performance when the local window size and number of heads are 37 and 8. Moreover, we can see that due to the channel and head selective mechanisms, our ADSFormer_SA outperforms ActionFormer under all parameter values. For ADSFormer_AFNO, since it mixes the tokens by a global manner in each head, there is no window size parameter for it. However, the mAP results of ADSFormer_AFNO in Table XII with different numbers of token mixing heads can justify our parameter settings in the previous experiments.

To study the impact of feature pyramid, we test the performances of ADSFormer_SA and ADSFormer_AFNO under different numbers of pyramid levels. From Table XIII, we can see that the TAL results of our methods improve with the increase of pyramid levels at beginning. This proves the feature pyramid structure is important in our ADSFormer. Nevertheless, after achieving their optimal results with 6 pyramid levels, the mAPs obtained by our ADSFormer_SA and ADSFormer_AFNO drop from 69.4% and 69.5% to 68.1% and 68.9%, respectively. The reason to this phenomenon may lies in that the over compressed features in high pyramid level (i.e., the 7-th level) will lose the information of some short actions with very small time steps.

G. Error Diagnosis

To further evaluate our method, we conduct several analyses using DETAD tools [75] on THUMOS14 dataset. The ADSFormer_SA is taken as an example in this error diagnosis. For more detailed explanations and metrics used in error diagnosis of DETAD, please refer to [75].

False Positive Analysis: Fig. 4 shows the false positive profiles and impact of error types on the performance of our ADSFormer_SA. It demonstrates the percentage of different action types in various k-G numbers, where G is the number of ground-truth for each action category and the top $k \times G$ predicted actions are kept for visualization.

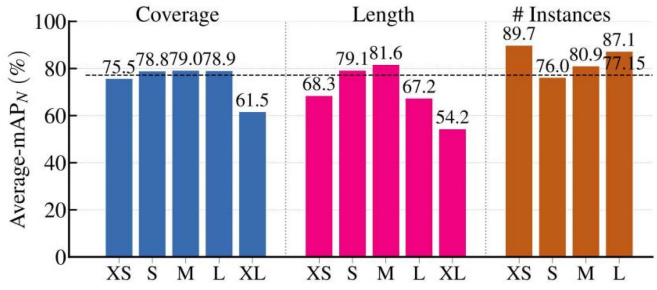

Fig. 5. Average-mAP of our ADSFormer_SA for different action metrics, where mAP_N is the normalized mAP at tIoU = 0.5 and the dashed line is the overall performance.

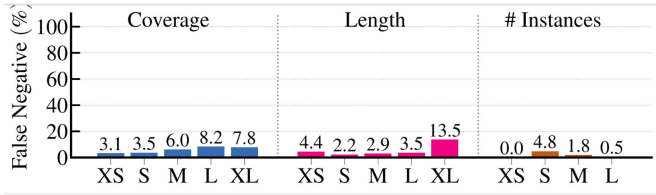

Fig. 6. Average false negative rate of ADSFormer_SA for various characteristics of actions.

From Fig. 4, we observe that the True Positive rate is high in 1 G prediction, i.e., about 80% at IoU = 0.5, which indicates that our method can accurately classify most of actions to their corresponding classes. Furthermore, we can find from the right of Fig. 4 that localization error and background confusion are the top two error types impact the performance of our methods. This observation is similar to those in some other studies [24], [26], [75]. That is, eliminating these two errors is an important way to improve the performance of TAL.

Sensitivity Analysis: Fig. 5 shows the sensitivity of our ADSFormer_SA to the characteristics of different actions. Specifically, Coverage is the relative length of action (compared with the whole video contains it), Length denotes the absolute length (in seconds) of action and # Instances refers to the total count of actions (from the same class) in a video. The TAL results of each metrics are further divided into various categories, that is, XS (extremely short), S (short), M (medium), L (long) and XL (extremely long). We can see that the performance of our method achieves satisfied results over most of the action coverage, except for XL. Additionally, our method performs better when the length of action is neither too short nor too long. This may be attributed to that the extremely short actions are hard to be captured and the extremely long actions contain more complicated information, which makes them naturally more challenging.

False Negative Analysis: From Fig. 6 we can find our model suffers from L and XL in Coverage. For Length metric, the False Negative results in XS and XL categories are relatively high. The above observations are consistent with those in Fig. 5. In # Instances metric, it can be seen that for the videos contain only one action, our model can accurately detect them without any miss (0.0 in XS of # Instances). Moreover, the False Negative result in L of # Instances is only 0.5, which demonstrates our method also has strong capability to handle the videos with large number of actions.

V. Conclusion

In this article, a Transformer architecture with selective mechanisms is proposed to deal with the TAL problem. By introducing the channel and head selective blocks into the multi-head token mixer in the Transformer, important and discriminative features can be obtained from different token mixing heads in our ADSFormer. Furthermore, we incorporate the proposed ADSFormer in a pyramid structure to encode video features and the multi-scale representations of different pyramid levels are combined for action classification and localization. Experimental results on four benchmark datasets demonstrate the effectiveness of the two proposed ADSFormer variants, i.e., ADSFormer_SA and ADSFormer_AFNO.

As shown in error diagnosis, although our ADSFormer achieves good performance in most cases, its TAL results for some extremely short and long actions are not ideal. Thus, we will try to find a solution of this limitation in our future work. Furthermore, designing a more sophisticated way to accurately estimate the action boundaries and suppress the localization errors is another important future research direction.

References

[1] K. Xia et al., "Exploring action centers for temporal action localization," *IEEE Trans. Multimedia*, vol. 25, pp. 9425–9436, 2023.
[2] P. Chen et al., "Relation attention for temporal action localization," *IEEE Trans. Multimedia*, vol. 22, no. 10, pp. 2723–2733, Oct. 2020.
[3] M.-G. Gan and Y. Zhang, "Temporal attention-pyramid pooling for temporal action detection," *IEEE Trans. Multimedia*, vol. 25, pp. 3799–3810, 2023.
[4] X. Feng and P. Perona, "Human action recognition by sequence of movelet codewords," in *Proc. IEEE 1st Int. Symp. 3D Data Process. Visual. Transmiss.*, 2002, pp. 717–723.
[5] N. Dalal, B. Triggs, and C. Schmid, "Human detection using oriented histograms of flow and appearance," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 428–441.
[6] J. C. Niebles, H. Wang, and L. Fei-Fei, "Unsupervised learning of human action categories using spatial-temporal words," *Int. J. Comput. Vis.*, vol. 79, no. 3, pp. 299–318, 2008.
[7] H. Wang, A. Kläser, C. Schmid, and C. Liu, "Dense trajectories and motion boundary descriptors for action recognition," *Int. J. Comput. Vis.*, vol. 103, no. 1, pp. 60–79, 2013.
[8] V. Escorcia, F. C. Heilbron, J. C. Niebles, and B. Ghanem, "DaPs: Deep action proposals for action understanding," in *Proc. 14th Eur. Conf. Comput. Vis.*, 2016, pp. 768–784.
[9] C. Lin et al., "Fast learning of temporal action proposal via dense boundary generator," in *Proc. 34th AAAI Conf. Artif. Intell., 32nd Innov. Appl. Artif. Intell. Conf., 10th AAAI Symp. Educ. Adv. Artif. Intell.*, 2020, pp. 11499–11506.
[10] T. Lin, X. Liu, X. Li, E. Ding, and S. Wen, "BMN: Boundary-matching network for temporal action proposal generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 3888–3897.
[11] T. Lin, X. Zhao, H. Su, C. Wang, and M. Yang, "BSN: Boundary sensitive network for temporal action proposal generation," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 3–21.
[12] X. Liu et al., "Multi-shot temporal event localization: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 12596–12606.
[13] C. Lin et al., "Learning salient boundary feature for anchor-free temporal action localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3320–3329.

[14] L. Yang, H. Peng, D. Zhang, J. Fu, and J. Han, "Revisiting anchor mechanisms for temporal action localization," *IEEE Trans. Image Process.*, vol. 29, pp. 8535–8548, 2020.

[15] T. Li, X. Zhao, and Z. Shou, "Single shot temporal action detection," in *Proc. ACM Multimedia Conf.*, 2017, pp. 988–996.

[16] J. Carreira and A. Zisserman, "Quo vadis, action recognition? A new model and the kinetics dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4724–4733.

[17] D. Wei et al., "Efficient dual attention slowfast networks for video action recognition," *Comput. Vis. Image Understanding*, vol. 222, 2022, Art. no. 103484.

[18] L. Wang et al., "Temporal segment networks: Towards good practices for deep action recognition," in *Proc. 14th Eur. Conf. Comput. Vis.*, 2016, pp. 20–36.

[19] M. Xu, C. Zhao, D. S. Rojas, A. K. Thabet, and B. Ghanem, "G-TAD: Sub-graph localization for temporal action detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 10153–10162.

[20] C. Zhao, A. K. Thabet, and B. Ghanem, "Video self-stitching graph network for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13638–13647.

[21] S. Buch, V. Escorcia, B. Ghanem, L. Fei-Fei, and J. C. Niebles, "End-to-end, single-stream temporal action detection in untrimmed videos," in *Proc. Brit. Mach. Vis. Conf.* 2017, pp. 93.1–93.12.

[22] A. Vaswani et al., "Attention is all you need," in *Proc. 30th Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[23] F. Cheng and G. Bertasius, "Tallformer: Temporal action localization with a long-memory transformer," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 503–521.

[24] C. Zhang, J. Wu, and Y. Li, "ActionFormer: Localizing moments of actions with transformers," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 492–510.

[25] T. N. Tang, K. Kim, and K. Sohn, "TemporalMaxer: Maximize temporal context with only max pooling for temporal action localization," 2023, *arXiv:2303.09055*.

[26] D. Shi et al., "Tridet: Temporal action detection with relative boundary modeling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 18857–18866.

[27] T. Chen, Z. Zhang, Y. Cheng, A. H. Awadallah, and Z. Wang, "The principle of diversity: Training stronger vision transformers calls for reducing all levels of redundancy," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12010–12020.

[28] C. Gong, D. Wang, M. Li, V. Chandra, and Q. Liu, "Vision transformers with patch diversification," 2021, *arXiv:2104.12753*.

[29] A. R. Webb, R.O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*. Hoboken, NJ, USA: Wiley, 2001.

[30] T. Zheng, B. Li, H. Bao, T. Xiao, and J. Zhu, "EIT: Enhanced interactive transformer," 2022, *arXiv:2212.10197*.

[31] N. Shazeer, Z. Lan, Y. Cheng, N. Ding, and L. Hou, "Talking-heads attention," 2020, *arXiv:2003.02436*.

[32] J. Guibas et al., "Efficient token mixing for transformers via adaptive fourier neural operators," in *Proc. Int. Conf. Learn. Representations*, 2021, pp. 1–15.

[33] H. Idrees et al., "The THUMOS challenge on action recognition for videos 'in the wild'," *Comput. Vis. Image Underst.*, vol. 155, pp. 1–23, 2017.

[34] D. Damen et al., "Rescaling egocentric vision: Collection, pipeline and challenges for EPIC-KITCHENS-100," *Int. J. Comput. Vis.*, vol. 130, no. 1, pp. 33–55, 2022.

[35] S. Yeung et al., "Every moment counts: Dense detailed labeling of actions in complex videos," *Int. J. Comput. Vis.*, vol. 126, no. 2–4, pp. 375–389, 2018.

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

[37] F. C. Heilbron, J. C. Niebles, and B. Ghanem, "Fast temporal activity proposals for efficient detection of human actions in untrimmed videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1914–1923.

[38] G. Gong, L. Zheng, and Y. Mu, "Scale matters: Temporal scale aggregation network for precise action localization in untrimmed videos," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2020, pp. 1–6.

[39] P. Zhao et al., "Bottom-up temporal action localization with mutual regularization," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 539–555.

[40] Y. Bai et al., "Boundary content graph neural network for temporal action proposal generation," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 121–137.

[41] R. Zeng et al., "Graph convolutional networks for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 7093–7102.

[42] Z. Qing et al., "Temporal context aggregation network for temporal action proposal refinement," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 485–494.

[43] D. Sridhar et al., "Class semantics-based attention for action detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13719–13728.

[44] F. Long et al., "Gaussian temporal awareness networks for action localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 344–353.

[45] S. Nag, X. Zhu, Y. Song, and T. Xiang, "Proposal-free temporal action detection via global segmentation mask learning," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 645–662.

[46] A. Dosovitskiy et al., "An image is worth 16x16 words: Transformers for image recognition at scale," in *Proc. 9th Int. Conf. Learn. Representations*, 2021, pp. 1–22.

[47] J. Tan, J. Tang, L. Wang, and G. Wu, "Relaxed transformer decoders for direct action proposal generation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13506–13515.

[48] X. Liu et al., "End-to-end temporal action detection with transformer," *IEEE Trans. Image Process.*, vol. 31, pp. 5427–5441, 2022.

[49] D. Shi et al., "React: Temporal action detection with relational queries," in *Proc. 17th Eur. Conf. Comput. Vis.*, 2022, pp. 105–121.

[50] S. Chang, P. Wang, F. Wang, H. Li, and Z. Shou, "Augmented transformer with adaptive graph for temporal action proposal generation," in *Proc. 3rd Int. Workshop Hum.-Centric Multimedia Anal.*, 2022, pp. 41–50.

[51] L. Wang, H. Yang, W. Wu, H. Yao, and H. Huang, "Temporal action proposal generation with transformers," 2021, *arXiv:2105.12043*.

[52] H. Liu, Z. Dai, D. R. So, and Q. V. Le, "Pay attention to MLPs," in *Proc. Int. Conf. 34th Adv. Neural Inf. Process. Syst.*, 2021, pp. 9204–9215.

[53] I. O. Tolstikhin et al., "MLP-mixer: An all-MLP architecture for vision," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst., 34th Annu. Conf. Neural Inf. Process. Syst.*, 2021, pp. 24261–24272.

[54] Y. Rao, W. Zhao, Z. Zhu, J. Zhou, and J. Lu, "GFNet: Global filter networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 9, pp. 10960–10973, Sep. 2023.

[55] W. Yu et al., "Metaformer is actually what you need for vision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 10809–10819.

[56] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontañón, "FNet: Mixing tokens with fourier transforms," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics: Hum. Lang. Technol.*, 2022, pp. 4296–4313.

[57] K. M. Choromanski et al., "Rethinking attention with performers," in *Proc. 9th Int. Conf. Learn. Representations*, 2021, pp. 1–38.

[58] H. Alwassel, S. Giancola, and B. Ghanem, "TSP: Temporally-sensitive pretraining of video encoders for localization tasks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshops*, 2021, pp. 3166–3176.

[59] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.

[60] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 2999–3007.

[61] H. Rezatofighi et al., "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 658–666.

[62] S. Zhang, C. Chi, Y. Yao, Z. Lei, and S. Z. Li, "Bridging the gap between anchor-based and anchor-free detection via adaptive training sample selection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9756–9765.

[63] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. 7th Int. Conf. Learn. Representations*, 2019, pp. 1–8.

[64] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. 5th Int. Conf. Learn. Representations*, 2017, pp. 1–16.

[65] L. Wang et al., "Videomae V2: Scaling video masked autoencoders with dual masking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 14549–14560.

[66] Q. Liu and Z. Wang, "Progressive boundary refinement network for temporal action detection," in *Proc. 34th AAAI Conf. Artif. Intell., 32nd Innov. Appl. Artif. Intell. Conf., 10th AAAI Symp. Educ. Adv. Artif. Intell.*, 2020, pp. 11612–11619.

[67] Z. Zhu, W. Tang, L. Wang, N. Zheng, and G. Hua, "Enriching local and global contexts for temporal action localization," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13496–13505.

[68] Z. Zhu et al., "Learning disentangled classification and localization representations for temporal action localization," in *Proc. 36th AAAI Conf. Artif. Intell., AAAI 34th Conf. Innov. Appl. Artif. Intell., IAAI 12th Symp. Educ. Adv. Artif. Intell., Virtual Event*, 2022, pp. 3644–3652.

[69] L. Yang, J. Han, T. Zhao, N. Liu, and D. Zhang, "Structured attention composition for temporal action localization," *IEEE Trans. Image Process.*, Jun. 13, 2022, doi: 10.1109/TIP.2022.3180925.

[70] T. Kang, G. Lee, and S. Lee, "HTNet: Anchor-free temporal action localization with hierarchical transformers," in *Proc. IEEE Int. Conf. Syst., Man, Cybern.*, 2022, pp. 365–370.

[71] R. Dai et al., "PDAN: Pyramid dilated attention network for action detection," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2021, pp. 2969–2978.

[72] P. Tirupattur, K. Duarte, Y. S. Rawat, and M. Shah, "Modeling multi-label action dependencies for temporal action localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 1460–1470.

[73] R. Dai, S. Das, K. Kahatapitiya, M. S. Ryoo, and F. Brémond, "MS-TCT: Multi-scale temporal convtransformer for action detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 20009–20019.

[74] J. Tan, X. Zhao, X. Shi, B. Kang, and L. Wang, "Pointtad: Multi-label temporal action detection with learnable query points," in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2022, pp. 15268–15280.

[75] H. Alwassel, F. C. Heilbron, V. Escorcia, and B. Ghanem, "Diagnosing error in temporal action detectors," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 264–280.