TP^2-DETR: Unlocking Deformable DETR for Zero-Shot Temporal Action Proposal Generation with Temporal Feature Pyramids

Ya-Yun Cheng[1] Kan Tippayamontri[2] Chih-Yuan Yang[23✉] Jane Yung-jen Hsu[12✉]
{r12922062,yjhsu}@csie.ntu.edu.tw {m1461023,cyyang}@cgu.edu.tw
[1]Department of Computer Science and Information Engineering, National Taiwan University
[2]Department of Artificial Intelligence, Chang Gung University
[3]Artificial Intelligence Research Center, Chang Gung University

Abstract

Standard Transformer attention often causes over-smoothing in temporal action localization, particularly with slow-moving video content. A compelling remedy is to adopt deformable attention because of its enhanced capability to focus on relevant features. However, existing Deformable DETR-based methods often fail to fully exploit this benefit, especially for detecting short actions in zero-shot settings where features are derived from vision-language models. This limitation stems from the lack of an effective temporal feature pyramid, which restricts the method's ability to precisely localize short actions, similar to how feature pyramids aid small object detection in images.

To this end, we propose TP^2-DETR, a novel end-to-end framework designed to fully unlock the potential of Deformable DETR for Zero-Shot Temporal Action Proposal Generation (ZSTAPG). Our key innovation is the integration of a dedicated temporal feature pyramid network, with several effective variants explored. For system efficiency and stability, we also design a shared, lightweight, and multi-scale-aware salient head for early supervision, which is augmented by auxiliary prediction heads providing deep supervision. Evaluated on the THUMOS14 and ActivityNet1.3 datasets, TP^2-DETR achieves state-of-the-art performance under standard zero-shot split settings, thereby validating its success in harnessing Deformable DETR's potential for precise and effective ZSTAPG. Our code is available at https://github.com/kantippayamontri/TP2_DETR

1. Introduction

Temporal Action Localization (TAL) is a fundamental task in video understanding that aims to precisely identify the start time, end time, and the category of specific actions within long, untrimmed videos. This capability is highly valuable for numerous real-world applications. Based on this definition, TAL can be naturally decomposed into two subtasks: proposal generation, which determines the candidate temporal segments containing actions, and proposal classification, which assigns the definitive action label to each segment.

In recent years, TAL methods have made remarkable progress under the closed-set setting [19, 20, 35]. However, the labor-intensive and expensive nature of obtaining temporal annotations for long, untrimmed videos significantly limits the scalability of fully-supervised approaches. Consequently, the emergence of powerful semantic alignment and transferability in vision-language models (VLMs) has made Zero-Shot Temporal Action Localization (ZSTAL) a compelling area of research. Most ZSTAL methods leverage pre-trained VLMs to address the classification subtask.

Transformers are widely used to model temporal relationships in videos due to their powerful attention mechanism. However, applying standard dense attention across long video sequences often leads to over-smoothing because slow-changing video content causes repeated dense attention to over-average similar frame features. Consequently, the discriminability at each timestamp is weakened, making it harder to accurately localize action boundaries.

To address the over-smoothing issue, deformable attention offers a promising solution. As first introduced in Deformable DETR [36] for object detection, it employs a sparse attention strategy that focuses only on a small set of relevant positions. This inherent sparsity mitigates over-smoothing, which has led several recent studies [20, 25] to adopt Deformable DETR as the core architecture for TAL.

Existing approaches often emphasize deformable attention, yet another key strength of Deformable DETR—its use of multi-scale feature pyramids to improve small object detection—has not been fully leveraged in the temporal domain. This omission is due to the fundamental differ-

ence between modalities: unlike images, which naturally produce multi-scale feature maps from their ResNet [6] feature extractor, videos lack an explicit and natural temporal downsampling path. This challenge is amplified in zero-shot settings, relying on features from pre-trained VLMs. These features, typically extracted from the final encoder layer, provide a flat temporal representation.

To address these limitations and fully unlock the potential of Deformable DETR in the temporal domain, we propose TP^2-DETR (**T**emporal Feature **P**yramids-based **T**emporal Action **P**roposal Generation DETR). This novel architecture is specifically designed for Zero-Shot Temporal Action Proposal Generation (ZSTAPG), the proposal generation subtask of ZSTAL.

Based on the insights above, TP^2-DETR is designed with three key components:

1. A Temporal Feature Pyramid Network (TFPN), which generates the multi-scale feature representation, i.e. a Temporal Feature Pyramid (TFP).
2. Multi-Scale-Aware Salient Heads, adopted from [4] for early supervision but modified by a unified MLP.
3. Auxiliary Supervision for Stability, additional prediction heads to each intermediate decoder layer for deep supervision to improve training stability.

TP^2-DETR is evaluated on two public benchmarks: THUMOS14 [8] and ActivityNet v1.3 [7]. Our model demonstrates superior performance across nearly all zero-shot split settings compared to prior work. The improvements are particularly significant on THUMOS14, a dataset characterized by short actions. This strong result validates our core motivation: effectively adapting the strengths of Deformable DETR's small-object detection capabilities to the challenging task of short-action localization in the ZSTAPG setting.

2. Related Work

ZSTAPG is a challenging sub-task within ZSTAL, whose overall goal is to locate the start and end times of actions in an untrimmed video and classify them, even if the action category was not seen during the model's training phase.

Temporal Action Localization. Typical TAL methods assume that all action classes are available during both training and inference, and they generally fall into two categories: two-stage methods [4, 22, 28] and one-stage methods [9, 16, 20, 31, 35]. The former separates the process into proposal generation and proposal classification, often using dedicated modules for each, while the latter jointly handles both subtasks within a unified framework.

TAL is conceptually analogous to object detection. The key difference is that while object detection localizes objects in 2D spatial space, TAL localizes actions over a 1D temporal sequence. This similarity has enabled the adaptation of many object detection techniques to the TAL domain. In particular, DETR-based [4, 11–13, 20, 27, 37] and DETR-like [29, 30] architectures have been widely adopted in recent TAL research. For example, TadTR [20] builds directly upon Deformable DETR, while GAP [4] adopts Conditional DETR for generalizable zero-shot proposal generation, and several other studies propose improvements upon the vanilla DETR [12, 27, 37]. Other models, such as RTD-Net [29] and PointTAD [30], follow similar DETR-style decoding and bipartite matching, but deviate from standard DETR-based designs by removing the transformer encoder, relaxing one-to-one matching, or incorporating point-based queries and custom interaction modules.

Zero-Shot Temporal Action Localization. Most ZSTAL methods leverage VLMs, which offer remarkable zero-shot generalization capabilities. Since the emergence of VLMs such as CLIP [26] in 2021, ZSTAL has attracted increasing attention, with relevant studies starting in 2022. Similar to general TAL models, training-based ZSTAL methods can be categorized into two-stage and one-stage frameworks, depending on whether they implement proposal generation and proposal classification as sequential components or as a unified architecture.

Early two-stage training-based ZSTAL methods, such as EffPrompt [9], adopt existing detectors AFSD [15] and A2Net [33] for proposal generation, then classify them by adopting CLIP via prompting. MMPrompt [10] enhances this by using optical flow for motion cues and visual-conditioned prompts for classification. More recently, GAP [4] focuses solely on improving proposal generation. It utilizes a Conditional DETR architecture trained with proposal-level objectives, rather than frame-level ones, and incorporates static representations for refinement. GAP is the current state-of-the-art and serves as the primary DETR-based reference for our work.

One-stage training-based ZSTAL was pioneered by STALE [22], which jointly performs localization and classification using parallel branches and a consistency loss to align them. UnLoc [32] advanced this by using a feature pyramid and an early video-text fusion module to inject language priors during decoding. More recently, ZEETAD [25] replaced text prompt tuning with an adapter-based module and incorporated Deformable DETR to improve the handling of ambiguous temporal boundaries during proposal generation.

Training-Free Approaches. With the increasing expressiveness of VLMs and the goal of avoiding bias introduced by the training process, training-free approaches have started to gain attention in recent ZSTAL research. T3AL [14] aligns visual and textual embeddings using CoCa [34] to obtain video-level pseudo labels, applies test-time adaptation via self-supervised projectors, and uses CoCa again to perform text-guided region suppression as post-processing. ZEAL [1] prompts a Large Language

Model (LLM), such as GPT-4 [24], to expand action class names into detailed descriptions, which serve as queries for a Large Vision-Language Model (LVLM). The LVLM generates frame-level confidence scores, while CLIP is used to narrow the search space. FreeZAD [5] replaces general similarity scores from CoCa with the LogOIC score to enable more stable boundary evaluations, and integrates frequency-based signals to calibrate actionness for more reliable ranking in the final localization outputs. Overall, although training-free approaches currently underperform compared to training-based methods, they demonstrate the promising potential of VLMs to enable ZSTAL without requiring any additional training.

Transformer-based TAL methods against Over-smoothing. Transformer architectures are popular in TAL, including DETR-based models, for their ability to model long-range temporal dependencies. However, applying standard transformer attention to video data causes the problem of over-smoothing. This occurs because standard attention employs dense aggregation, which, when combined with the high temporal redundancy of video features, results in overly homogenized representations. This lack of discriminability blurs temporal boundaries and hinders precise action localization, particularly for short actions.

To address over-smoothing, recent DETR-based TAL methods introduce architectural modifications to revise or replace standard attention mechanisms. TranZAD [23] and RTD-Net [29] replace the transformer encoder with a multi-layer perceptron (MLP) to reduce feature mixing and preserve temporal distinctiveness. TadTR [20] and ZEETAD [25] utilize Deformable DETR for its deformable attention. TadTR is applied to fully-supervised TAL, while ZEETAD targets the zero-shot setting. ReAct [27] extends a similar concept by introducing a relational attention mechanism that allows the model to selectively attend to relevant action queries and preserve their discriminability.

Feature Pyramid. For object detection, multi-scale feature pyramids have been widely used in literature such as FPN [17] and RetinaNet [18] where features from the last few layers provide a spatial pyramid with increasing semantic abstraction and receptive field in the backbone networks such as ResNet [6]. However, videos do not have such an intuitive temporal downsampling path. Especially in ZSTAL or ZSTAPG, it is common to use VLMs such as CLIP to extract frame-wise or snippet-wise features, which are typically pre-extracted and come from the final encoder layer. These features lack intermediate representations and therefore do not form a natural temporal hierarchy in the same way that ResNet does for images. Even if intermediate features are extracted from VLMs, they are not structurally aligned or semantically organized in a way that supports a temporal resolution hierarchy, making them unsuitable for direct use as a temporal pyramid. As a result, these limitations highlight the need for a dedicated TFPN design.

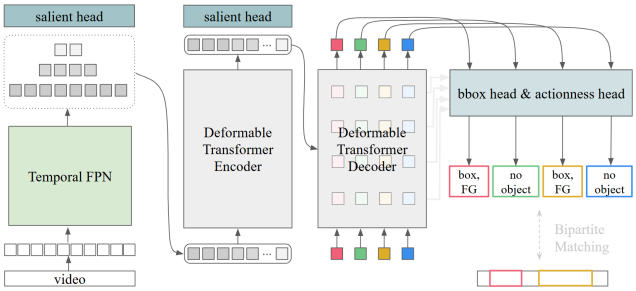


Figure 1. Model Overview. Given an input video, snippet-wise features are extracted using the CLIP visual encoder. These features are processed by a Temporal FPN (TFPN) to produce multi-scale temporal representations, which are then encoded and decoded by the Deformable DETR architecture. The model generates action proposals through prediction heads, while auxiliary salient and prediction heads are added to support stable end-to-end learning.

3. Proposed Method

Our approach involves creating a TFP on a Deformable DETR-based TAPG method and adding auxiliary heads to stabilize the training process, as illustrated in Figure 1.

Given an input video, we first extract snippet-wise features using the visual encoder of CLIP [26] for computational efficiency. These features are then processed by a TFPN, which generates a TFP. The TFP is subsequently encoded by a Deformable Transformer encoder and decoded by a Deformable Transformer decoder initialized with a set of learnable object queries. The resulting output embeddings from the decoder are passed through prediction heads (i.e., bounding box and actionness heads) to output the proposal segments and their corresponding actionness scores. To stabilize training and encourage multi-scale temporal reasoning throughout the pipeline, we incorporate auxiliary salient heads and prediction heads.

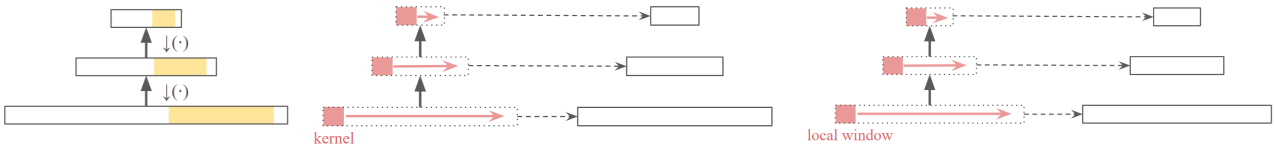
Our contributions lie in three components: (a) TFPN (b) Multi-Scale Aware Salient Head (c) Auxiliary Supervision for Stable Training, and we explain them as follows.

3.1. TFPN variants

We design four TFPN variants to investigate their performance within Deformable DETR architecture.

TFPN by Direct Downsampling. The simplest way is to downsample the video feature sequence directly, level by level, to form a trivial feature pyramid as illustrated in Figure 2(A). Let X^1 be the snippet features generated from an input video V through a CLIP-based snippet feature extractor F_v as

$$X^1 = F_v(V) \in \mathbb{R}^{T^1 \times C}, \tag{1}$$



(A) Direct downsampling (B) CNN-based TFPN (C)Transformer-based TFPN

Figure 2. Three types of TFPN. The yellow regions in (A) indicate an action instance located within a temporal span.

where T^1 is the number of timestamps of V and C is the feature dimension. We generate the l-th layer by

$$X^l = \downarrow_n (X^{l-1}) \in \mathbb{R}^{T^l \times C}, \tag{2}$$

where $\downarrow_n (\cdot)$ denotes nearest-neighbor downsampling with a fixed rate 2, and $T^l = \frac{1}{2}T^{l-1}$.

CNN-based TFPN. Similar to the spatial FPN [17], which expands the receptive field and preserves complementary semantics across scales, we build a TFP by applying one-dimensional convolutions with a stride of 2, which is formulated as

$$X^l = \text{LN}(\text{Conv}(X^{l-1})) \in \mathbb{R}^{T^l \times C}, \tag{3}$$

where Conv means convolution and LN$(\cdot)$ denotes Layer Normalization.

Enhanced CNN-based TFPN. To enhance feature discriminability, we use the Scalable-Granularity Perception (SGP) layers from TriDet [28] as our TFP. The SGP module is designed to be a substitute for the self-attention module in a transformer layer, but still captures both global context and local receptive fields. Given that Deformable DETR's core strength lies in preserving discriminability via deformable attention, we explore the synergy of integrating SGP layers and Deformable DETR. We generate the pyramid layer X^l by first creating an intermediate layer

$$\bar{X}^l = \text{SGP}\big(\text{LN}(X^{l-1})\big) + X^{l-1}, \tag{4}$$

where $+$ means element-wise addition and

$$\begin{aligned}\text{SGP}(X) &= \phi(X) \odot \text{FC}(X) \\ &+ \text{Conv}_w(X) \odot \big(\text{Conv}_w(X) + \text{Conv}_{kw}(X)\big) + X,\end{aligned} \tag{5}$$

$$\phi(X) = \text{ReLU}(\text{FC}(\text{AvgPool}(X))), \tag{6}$$

where $\odot$ means element-wise multiplication, and FC and Conv_w denote the fully-connected layer and the 1-D depth-wise convolution layer [3] over the temporal dimension with window size w. Then, we compute

$$\widetilde{X}^l = \downarrow_p (\text{FFN}\big(\text{GN}(\bar{X}^l)\big) + \bar{X}^l, \tag{7}$$

where FFN$(\cdot)$ is an MLP feed-forward network, and GN$(\cdot)$ indicates Group Normalization. Finally, we generate the next layer

$$X^l = \downarrow_p (\widetilde{X}^l) \in \mathbb{R}^{T^l \times C}, \tag{8}$$

where $\downarrow_p (\cdot)$ denotes max-pooling downsampling.

Transformer-based TFPN. We modify the multi-scale transformer encoder proposed by ActionFormer [35] as our TFPN in this type. ActionFormer's multi-scale transformer encoder uses local attention rather than the typical global attention, which only pays attention to nearby tokens in a local window to speed up computation, like the kernel size in CNN, as shown in Figure 2(C). We modify the original multi-scale transformer encoder to fit our ZSTAPG setting. Since our input features X are already high-level representations encoded by CLIP, we eliminate the depthwise convolutions and normalization layers that were originally applied at the beginning of each pyramid level. In addition, instead of using additional strided convolutions, we simply apply nearest-neighbor interpolation for downsampling. We make these adjustments to retain the original temporal structure and avoid introducing unnecessary transformations that might distort the feature semantics. We calculate X^l through

$$\bar{X}^l = \alpha^l \text{LocalMHSA}\big(\text{LN}(X^{l-1})\big) + X^{l-1}, \tag{9}$$

$$\widetilde{X}^l = \bar{\alpha}^l \text{FFN}\big(\text{LN}(\bar{X}^l)\big) + \bar{X}^l, \tag{10}$$

$$X^l = \downarrow_n (\widetilde{X}^l) \in \mathbb{R}^{T^l \times C}, \tag{11}$$

where LocalMHSA means local multi-head self-attention, and α^l and $\bar{\alpha}^l$ are hyperparameters.

3.2. Salient Head

The salient head for ZSTAPG is proposed by GAP [4] to stabilize the training process. It is a small, independent network to predict the foreground probability (i.e. actionness) for each timestamp in a video, as illustrated in Figure 3, as early supervision for the final proposal generation. It is used only during training, and disregarded during inference. We adopt the idea of using a salient head, but adjust its network architecture and add one more salient head. GAP's original salient head is implemented as a 1D CNN on top of the transformer encoder's output sequence.

MLP-based Salient Head. We replace the CNN layers with MLPs and refer to it as the MLP-based Salient Head. Although MLPs are less capable of modeling local temporal context, they better preserve sharp boundary predictions

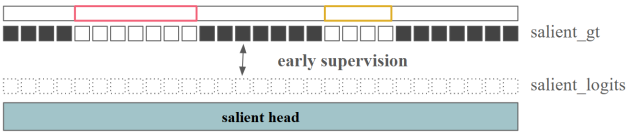


Figure 3. Illustration of the salient head. Ground-truth action segments (reg and brown rectangles) produce a binary mask salient_gt where foreground frames are shown as white squares while background frames as black squares. The salient head outputs per-timestamp scores (salient_logits), which are compared to salient_gt to compute the salient loss for early supervision.

Algorithm 1: Unified MLP-based Salient Head

Input: TFP layers $\{X^l\}_{l=1}^{L} \in \mathbb{R}^{T^l \times C}$
Output: Salient logits $S \in \mathbb{R}^{T^1 \times 1}$
Parameters: W_1, b_1, W_2, b_2 are weights and biases of two linear layers

$\{\tilde{X}^l\}_{l=2}^{L} \leftarrow$ Upsample $\{X^l\}_{l=2}^{L}$, temporally linearly to T^1
$X \leftarrow \text{Concat}(\{\tilde{X}^l\}_{l=1}^{L})$
$X_{\text{flat}} \leftarrow \text{Flatten}(X)$
$H \leftarrow \text{ReLU}(X_{\text{flat}} \cdot W_1 + b_1)$
$S \leftarrow H \cdot W_2 + b_2$
return S

because each neuron operates independently across timestamps. This property is particularly beneficial for short action proposals that require high boundary precision.

Second Salient Head. We observe that not only the output sequence of the transformer encoder, but also the outputs of the TFPN can serve as the sources of another salient head. Thus, we propose a simple and effective fusion method, as detailed in Algorithm 1, to merge multi-scale sequences to predict salient scores. We concatenate the multi-scale features along the channel dimension and feed them into the MLP, allowing it to learn how to integrate different scale representations per timestamp. This design eliminates the need for manual fusion if each scale generates an individual salient score.

3.3. Deep supervision

Our remaining components, including bipartite matching and object-detection-based bounding box loss and classification loss, are similar to existing TAL and TAPG methods [4, 28, 35]. Because TP^2-DETR is a TAPG method, we convert all class labels into a binary actionness score $a \in \{0, 1\}$.

Bipartite Matching Loss. Let N_q be the number of our learnable queries and greater than the number of ground-truth proposals N_{gt} in a video. Let $\hat{p}_i = (\hat{b}_i, \hat{a}_i)$ denote a predicted proposal where $\hat{b}_i = (\hat{t}_i^s, \hat{t}_i^e)$ is a temporal window and $\hat{a}_i$ be an actionness score, and $p_j = (b_j, a_j), j = 1, \cdots, N_{gt}$ be the ground-truth action temporal windows and all a_j are assigned to the value 1. Padded $b_j = \emptyset$ for $N_{gt} < j \leq N_q$ and let π be a permutation of N_q elements, the optimal assignment $\hat{\pi}$ is determined by

$$\hat{\pi} = \arg\min_{\pi} \sum_{i=1}^{N_q} L_{\text{match}}(p_i, \hat{p}_{\pi(i)}), \tag{12}$$

and

$$L_{\text{match}}(p_i, \hat{p}_{\pi(i)}) = \mathbb{1}_{\{b_i \neq \emptyset\}}[\alpha_{L1} \cdot L_{L1}(b_i, \hat{b}_{\pi(i)}) + \alpha_{\text{tIoU}} \cdot L_{\text{tIoU}}(b_i, \hat{b}_{\pi(i)}) + \beta \cdot L_{\text{focal}}(a_i, \hat{a}_{\pi(i)})], \tag{13}$$

where α_{L1}, α_{tIoU}, and β are hyperparameters, and L_{L1} is the L1 loss, L_{tIoU} is the temporal IoU loss [20], and L_{focal} is a binary classification loss implemented via focal loss [18].

Training Objectives. Given the one-to-one assignments from bipartite matching, we define the overall training objective as

$$L = \lambda_{\text{bbox\_L1}} \cdot L_{\text{bbox\_L1}} + \lambda_{\text{bbox\_tIoU}} \cdot L_{\text{bbox\_tIoU}} + \lambda_{\text{actionness}} \cdot L_{\text{actionness}} + \lambda_{\text{salient}} \cdot L_{\text{salient}}, \tag{14}$$

where λs are hyperparameters and

$$L_{\text{bbox\_L1}} = \sum_{l=1}^{N_{\text{dec}}} \sum_{i=1}^{N_q} L_{L1}(b_i, \hat{b}^l_{\pi(i)}), \tag{15}$$

$$L_{\text{bbox\_tIoU}} = \sum_{l=1}^{N_{\text{dec}}} \sum_{i=1}^{N_q} L_{\text{tIoU}}(b_i, \hat{b}^l_{\pi(i)}), \tag{16}$$

$$L_{\text{actionness}} = \sum_{l=1}^{N_{\text{dec}}} \sum_{i=1}^{N_q} L_{\text{focal}}(a_i, \hat{a}^l_{\pi(i)}), \tag{17}$$

where N_{dec} is the number of transformer decoder layers, and $\hat{b}^l_{\pi(i)}$ and $\hat{a}^l_{\pi(i)}$ are the temporal window and actionness score predicted by the l-th layer's intermediate output sequence of our transformer decoder from the i-th learnable query. The salient loss L_{salient} is calculated from both the TFPN and the transformer encoder outputs as

$$L_{\text{salient}} = \sum_{t=1}^{T^1} \Big(L_{\text{BCE}}(m_t, s_t^{\text{fpn}}) + L_{\text{BCE}}(m_t, s_t^{\text{enc}}) \Big), \tag{18}$$

where L_{BCE} means binary cross-entropy loss, $m_t = 1$ if there is an action at timestamp t, otherwise 0. s_t^{fpn} and s_t^{enc} are salience scores generated by our two salient heads.

Inference. During inference, we simply forward the video features through the network to obtain the predicted proposals, i.e. $\hat{P} = \{(\hat{b}_i, \hat{a}_i)\}_{i=1}^{N_q}$, where each proposal consists of a predicted temporal window and its corresponding actionness score, generated by the learned bounding box head and actionness head.

Table 1. Comparison with SOTA ZSTAPG methods. AVG refers to the average mAP (%) computed across different IoU thresholds: [0.3:0.1:0.7] for THUMOS14 and [0.5:0.05:0.95] for ActivityNet1.3. We could not reproduce the GAP results using the released code and received no author response. Thus, their reported mAP scores are listed but omitted from the comparison.

Split	Method	mAP@tIoU for THUMOS14						mAP@tIoU for ANet1.3			
		0.3	0.4	0.5	0.6	0.7	AVG	0.5	0.75	0.95	AVG
75%/25%	TadTR [20]‡	30.49	26.61	21.98	16.53	11.08	21.34	37.59	23.34	2.97	23.06
	EffPrompt [9]	39.70	31.60	23.00	14.90	7.50	23.30	37.60	22.90	3.80	23.10
	STALE [22]	40.50	32.30	23.50	15.30	7.60	23.80	38.20	25.20	6.00	24.90
	ZEETAD [25]§	47.30	–	29.70	–	11.50	29.70	45.50	28.20	6.30	28.40
	GAP [4] reported	52.30	44.20	32.80	22.40	12.40	32.90	47.60	32.50	8.60	31.80
	GAP [4] reproduced	43.63	34.48	24.75	14.90	7.50	24.86	44.14	29.93	6.65	29.00
	GAP [4]‡ adjusted	52.50	42.69	30.78	18.71	9.00	30.72	46.91	29.08	4.65	28.81
	ActionFormer [35]	**58.09**	**49.92**	38.79	27.14	15.57	37.90	48.57	**31.33**	6.67	31.13
	TP^2-DETR Direct Downsampling	55.69	47.05	35.34	24.21	13.37	35.13	47.30	30.22	6.59	30.09
	TP^2-DETR CNN-based (basic)	53.46	44.65	34.88	22.86	12.09	33.59	47.81	30.38	6.58	30.37
	TP^2-DETR CNN-based (enhanced)	51.95	43.05	31.48	21.21	12.14	31.96	**48.62**	31.32	**7.15**	**31.17**
	TP^2-DETR Transformer-based	57.81	49.36	**38.80**	**27.91**	**16.34**	**38.04**	48.41	31.28	7.06	31.05
50%/50%	TadTR [20]‡	28.82	24.22	18.91	13.41	8.41	18.76	35.59	21.45	2.30	21.37
	EffPrompt [9]	37.20	29.60	21.60	14.00	7.20	21.90	32.00	19.30	2.90	19.60
	STALE [22]	38.30	30.70	21.20	13.80	7.00	22.20	32.10	20.70	5.90	20.50
	GAP [4] reported	44.20	36.00	27.10	15.10	8.00	26.10	41.60	26.20	6.10	26.40
	GAP [4] reproduced	37.38	29.39	20.86	12.89	6.03	21.36	40.44	27.32	5.33	26.52
	GAP [4]‡ adjusted	50.00	40.06	28.64	17.08	8.04	29.11	45.67	28.53	3.97	28.18
	ActionFormer [35]	55.92	47.00	36.22	24.11	13.20	35.29	47.40	30.03	6.03	29.94
	TP^2-DETR Direct Downsampling	53.86	44.79	33.98	22.38	11.87	33.37	45.78	28.67	5.58	28.67
	TP^2-DETR CNN-based (basic)	51.17	42.22	31.37	19.66	9.73	30.83	46.29	29.22	5.64	29.12
	TP^2-DETR CNN-based (enhanced)	51.71	42.40	31.09	20.31	11.07	31.32	**48.18**	**30.68**	**6.37**	**30.46**
	TP^2-DETR Transformer-based	**55.95**	**47.82**	**37.34**	**25.89**	**14.84**	**36.37**	47.18	30.25	6.28	30.08

‡ w/o class_logits § w/o optical flow fully-supervised method

4. Experiments

We evaluate our model on two publicly available benchmarks: THUMOS14 TAL [8] and ActivityNet1.3 [7]. The THUMOS14 TAL dataset contains 413 untrimmed videos with temporal annotations across 20 action classes. It provides 200 validation videos and 213 test videos. The average video duration is 212 seconds, with an average of 15 action instances per video, and the action instance duration is 5 seconds on average. Following the protocol established by GAP, we use the same training/test sets to conduct our experiments. The ActivityNet1.3 dataset is much larger, consisting of 19,994 (training/validation/test: 10,024/4,926/5,044) untrimmed videos covering 200 action categories. Its average video duration is 180 seconds, but in a long-tailed distribution. While most are short clips, a sparse but influential number are very long videos (5 to 10 minutes). There are only 1.5 action instances per video, but the average instance duration is 50 seconds, reflecting a lower granularity in temporal action labeling.

Zero-Shot Split Settings. We adopt the same protocol used by three existing methods [4, 9, 22] where the dataset is split into two settings: 75%/25% and 50%/50% for training/test. For both settings, we report the final performance as the average over ten different splits.

Evaluation Metrics. We use mean Average Precision (mAP) under multiple temporal Intersection over Union (tIoU) thresholds as the main evaluation metric, a standard criterion for measuring proposal quality for temporal action localization. In the general setting, AP is usually computed for each class independently, and the final mAP is averaged over all classes and thresholds. However, our model follows a two-stage design for ZSTAL, where the final classification is handled by vision-language models (VLMs). As a result, we intentionally design the proposal generation stage to be class-agnostic, where the model only predicts whether a segment is likely to contain any action (i.e., foreground), regardless of its semantic category. This design simplifies the proposal evaluation while remaining consistent with standard mAP computation practices, the details are shown in the supplementary material. We report mAP@[0.3:0.1:0.7] for THUMOS14 and mAP@[0.5:0.05:0.95] for ActivityNet1.3 for a pair comparison with existing methods [16].

Implementation Details. We adopt the visual encoder from pre-trained CLIP [26] (ViT-B/16) to extract video features with per-timestamp feature dimension $C = 512$ in the same manner as existing methods [4, 9, 22]. We use a snippet (aggregating 8 consecutive frames) as the timestamp unit for efficient computation. Our TFPN settings are shown in Table 2. Both the proposal generation heads (i.e., the bounding box head and the actionness head) are implemented using fully connected (FC) layers.

Table 2. Experimental settings. THU14 means THUMOS14, and ANet1.3 means ActivityNet1.3.

Item	THU14	ANet1.3
#pyramid layers	4	4
window size of Local MHSA	9	17
#Deformable DETR enc/dec layers	3/5	2/3
#reference points for enc/dec	4/4	4/6
#queries N_q	40	30
$\lambda_{salient}$	3	2

We use the AdamW [21] optimizer with a learning rate of 2×10^{-5} for the TFPN and 1×10^{-4} for the Deformable DETR. A multiplier of 0.1 is applied to the linear projection layers responsible for predicting reference points and sampling offsets. The weight decay is set to 1×10^{-4}. We train 35 epochs with a batch size 16 and a StepLR scheduler with a decay step at epoch 30.

Key hyperparameters for bipartite matching (Eq. 13) are $\alpha_{L1} = 5$, $\alpha_{tIoU} = 2$, and $\beta = 2$, and for training objectives (Eq. 14) are $\lambda_{\text{bbox\_L1}} = 5$, $\lambda_{\text{bbox\_tIoU}} = 2$, and $\lambda_{\text{actionness}} = 2$. Our method is implemented in PyTorch, and all experiments are conducted on a NVIDIA RTX 4080 GPU. Our model size is 43.35M, smaller than GAP's 53.34M, (Report Inference Speed here).

Results. Tables 1 shows the performance comparison among SOTA methods. To fairly compare them, we remove the semantic classification branch in GAP (denoted as GAP‡) to align with our model which does not rely on class-specific supervision. We reimplement TadTR by disabling its classification head and removing the computation and training involving the classification loss, and replace class logits with actionness scores for confidence prediction, which is consistent with the setup in TP^2-DETR.

In this case, TP^2-DETR consistently outperforms GAP‡ across all split settings, highlighting its robust generalization in the zero-shot scenario. All results are based solely on RGB-frame features extracted from CLIP (ViT-B/16), without incorporating additional modalities such as optical flow. Under this constraint, ZEETAD§ refers to a re-evaluated variant reproduced within the GAP framework.

Design of Salient Head. As described in the previous section, we adopt a unified MLP-based salient head applied to the encoder output, instead of using the multi-scale-aware CNN-based extension from GAP [4], which was originally designed for single-scale features. To assess the effectiveness of our design, we conduct experiments as shown in Table 3. Compared to applying the salient head only on the encoder output, the joint configuration yields consistently better performance, further validating the benefits of our proposed early supervision strategy.

Ablation Study. We show the performance of our components in Table 4. Our model introduces three core components: (1) a TFPN, (2) a multi-scale-aware salient head, and (3) auxiliary supervision for training stability. The auxiliary supervision includes both deep supervision via additional decoder-layer prediction heads and early supervision via a shared salient head applied to earlier stages. Because the salient head is functionally involved across two proposed components, we cannot isolate it from our model. From the experimental results, it is clear that every component consistently contributes to improve the performance.

Qualitative Comparison. We visualize the predicted results of a test video in Figure 4 and additional ones in the supplementary material. Specifically, we select the top-k class-agnostic proposals in two ways for visualization: (1) by highest actionness scores, following the GAP approach; and (2) by lowest bipartite matching cost, consistent with the matching process in the training stage. Here, k is set to the number of ground-truth action instances (N_{gt}) in each video. The visualizations show that TP^2-DETR produces more accurate and comprehensive coverage of ground-truth segments compared to both GAP and the baseline.

5. Conclusion

We propose TP^2-DETR, a model tailored to Deformable DETR for ZSTAPG. Experimental results demonstrate its effectiveness.

Limitations. Although TP^2-DETR performs well on THUMOS14, its improvement on ActivityNet1.3 is less consistent. This suggests that the model may not generalize as effectively to longer or more complex temporal patterns. A contributing factor could be the preprocessing design in ActivityNet1.3, where each video is uniformly resized to a fixed number of timestamps, regardless of its original duration. For long videos, this may lead to substantial temporal information loss, especially when actions span extended periods. In contrast to approaches that apply sliding window inference to preserve finer temporal granularity, our model processes the entire video as a single sequence, which may limit its capacity to model long-range structures. Future work could explore adaptive resizing schemes or window-based inference strategies to mitigate this issue. Furthermore, enhancing the modeling of long-range dependencies—such as through adaptive temporal scaling or more granular control over the local window sizes in self-attention—may improve the model's robustness across

Table 3. Analysis of the design of the salient head. We compare various feature fusion types for the per-scale CNN-based design and our unified MLP-based variants. Salient scores at low temporal resolution layers are linearly upsampled to the largest resolution for fusion.

Design of salient heads	mAP@tIoU for THUMOS14, 50%/50% split					
	0.3	0.4	0.5	0.6	0.7	AVG
2nd head on TFPN, CNN-based per scale, max fusion	55.74	47.49	36.86	25.05	14.43	35.91
2nd head on TFPN, CNN-based per scale, mean fusion	**56.47**	47.68	36.64	24.51	13.26	35.71
2nd head on TFPN, CNN-based per scale, adaptive fusion	55.89	47.24	36.22	24.47	13.87	35.57
Single head on encoder, Unified MLP-based	55.02	47.06	36.59	25.17	14.72	35.71
2nd head on TFPN, both Unified MLP-based	55.95	**47.82**	**37.34**	**25.89**	**14.84**	**36.37**

Table 4. Ablation study on the effectiveness of each proposed component.

Method	mAP@tIoU for THUMOS14, 50%/50% split					
	0.3	0.4	0.5	0.6	0.7	AVG
Baseline (Deformable DETR)	51.37	42.55	31.12	19.09	8.78	30.58
+ Transformer-based TFPN	53.90	44.69	33.82	22.45	11.03	33.18
+ Deep Supervision (bounding box and actionness heads)	**55.96**	47.18	36.37	24.84	13.80	35.63
+ Early Supervision (MLP-based salient head)	55.95	**47.82**	**37.34**	**25.89**	**14.84**	**36.37**

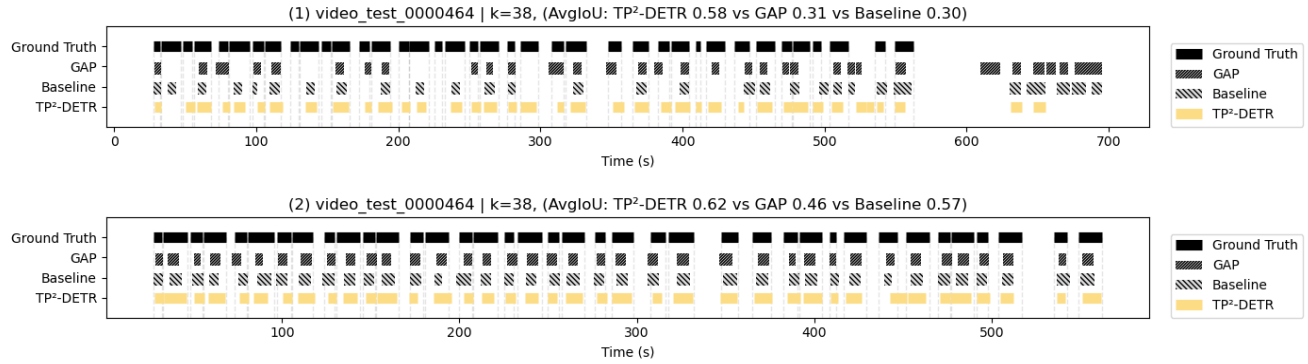


Figure 4. Qualitative comparison of the 50%/50% split. The THUMOS14 video 0000464 (class PoleVault) contains 38 action instances ($k = 38$). We select the top-k class-agnostic proposals in two ways: (1) by actionness scores (following the GAP approach) and (2) by bipartite matching using the same cost as in training. Our method better predicts the temporal bounding box than GAP and Deformable DETR (Baseline).

diverse action durations. Our current framework does not specifically analyze the impact of window size in the local multi-head self-attention used in the TFPN, which may influence its responsiveness to actions of varying lengths.

Future Work. Our model represents each timestamp as a single token, but in some efficient transformer designs such as EdgeViT [2], a token can cover larger and more context-aware temporal regions. This may help reduce temporal redundancy, enhance local context modeling, and improve efficiency, especially for long videos.

In addition, TP^2-DETR has not used any class-aware information, and the current design follows a two-stage paradigm, where proposal generation and classification are decoupled. Reformulating it as a one-stage method should be a promising direction to reduce the risk of error propagation between stages, as reported by STALE [22], and potentially improve both the efficiency and robustness of the overall system.

Furthermore, although our salient head is trained independently as an auxiliary component to facilitate early supervision, we observe that its predictions (i.e., salient logits) often overlap with those of the primary head (i.e., bounding box head). This implicit consistency suggests potential for cross-head information sharing, motivating a future study to explore mechanisms for shared or guided learning between these two components to further enhance proposal accuracy.

Acknowledgments

This work was supported by the National Science and Technology Council (NSTC) of Taiwan under Grant No. NSTC 115-2634-F-002-001.

References

[1] Josiah Aklilu, Xiaohan Wang, and Serena Yeung-Levy. Zero-shot Action Localization via the Confidence of Large Vision-Language Models. arXiv:2410.14340, 2024. 2

[2] Zekai Chen, Fangtian Zhong, Qi Luo, Xiao Zhang, and Yanwei Zheng. EdgeViT: Efficient visual modeling for edge computing. In Proceedings of the 17th International Conference on Wireless Algorithms, Systems, and Applications (WASA), pages 393–405, 2022. 8

[3] François Chollet. Xception: Deep learning with depthwise separable convolutions. In CVPR, pages 1800–1807, 2017. 4

[4] Jia-Run Du, Kun-Yu Lin, Jingke Meng, and Wei-Shi Zheng. Towards Completeness: A Generalizable Action Proposal Generator for Zero-Shot Temporal Action Localization. In ICPR, pages 252–267, 2024. 2, 4, 5, 6, 7

[5] Chaolei Han, Hongsong Wang, Jidong Kuang, Lei Zhang, and Jie Gui. Training-free zero-shot temporal action detection with vision-language models. arXiv:2501.13795, 2025. 3

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In CVPR, pages 770–778, 2016. 2, 3

[7] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A large-scale video benchmark for human activity understanding. In CVPR, pages 961–970, 2015. 2, 6

[8] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. http://crcv.ucf.edu/THUMOS14/, 2014. 2, 6

[9] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting Visual-Language Models for Efficient Video Understanding. In ECCV, pages 105–124, 2022. 2, 6, 7

[10] Chen Ju, Zeqian Li, Peisen Zhao, Ya Zhang, Xiaopeng Zhang, Qi Tian, Yanfeng Wang, and Weidi Xie. Multi-modal prompting for low-shot temporal action localization. arXiv:2303.11732, 2023. 2

[11] Ho-Joong Kim, Jung-Ho Hong, Heejo Kong, and Seong-Whan Lee. TE-TAD: Towards Full End-to-End Temporal Action Detection via Time-Aligned Coordinate Expression. In CVPR, pages 18837–18846, 2024. 2

[12] Jihwan Kim, Miso Lee, and Jae-Pil Heo. Self-feedback detr for temporal action detection. In ICCV, pages 10252–10262, 2023. 2

[13] Jihwan Kim, Miso Lee, Cheol-Ho Cho, Jihyun Lee, and Jae-Pil Heo. Prediction-Feedback DETR for Temporal Action Detection. In AAAI, pages 4266–4274, 2025. 2

[14] Benedetta Liberatori, Alessandro Conti, Paolo Rota, Yiming Wang, and Elisa Ricci. Test-time zero-shot temporal action localization. In CVPR, pages 18720–18729, 2024. 2

[15] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning Salient Boundary Feature for Anchor-free Temporal Action Localization. In CVPR, pages 3319–3328, 2021. 2

[16] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. BMN: Boundary-Matching Network for Temporal Action Proposal Generation. In ICCV, pages 3888–3897, 2019. 2, 6

[17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In CVPR, pages 936–944, 2017. 3, 4

[18] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In ICCV, pages 2980–2988, 2017. 3, 5

[19] Shuming Liu, Chen-Lin Zhang, Chen Zhao, and Bernard Ghanem. End-to-End Temporal Action Detection with 1B Parameters Across 1000 Frames. In CVPR, pages 18591–18601, 2024. 1

[20] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Shiwei Zhang, Song Bai, and Xiang Bai. End-to-End Temporal Action Detection With Transformer. IEEE TIP, 31:5427–5441, 2022. 1, 2, 3, 5, 6

[21] Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In ICLR, 2019. 7

[22] Sauradip Nag, Xiatian Zhu, Yi-Zhe Song, and Tao Xiang. Zero-Shot Temporal Action Detection via Vision-Language Prompting. In ECCV, pages 681–697, 2022. 2, 6, 7, 8

[23] Sayak Nag, Orpaz Goldstein, and Amit K. Roy-Chowdhury. Semantics Guided Contrastive Learning of Transformers for Zero-shot Temporal Activity Detection. In WACV, pages 6232–6242, 2023. 3

[24] OpenAI. GPT-4 Technical Report. arXiv:2303.08774, 2024. 3

[25] Thinh Phan, Khoa Vo, Duy Le, Gianfranco Doretto, Donald A. Adjeroh, and Ngan Le. ZEETAD: Adapting Pretrained Vision-Language Model for Zero-Shot End-to-End Temporal Action Detection. arXiv:2311.00729, 2023. 1, 2, 3, 6

[26] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. arXiv:2103.00020, 2021. 2, 3, 7

[27] Dingfeng Shi, Yujie Zhong, Qiong Cao, Jing Zhang, Lin Ma, Jia Li, and Dacheng Tao. ReAct: Temporal Action Detection with Relational Queries. In ECCV, pages 324–344, 2022. 2, 3

[28] Dingfeng Shi, Yujie Zhong, Qiong Cao, Lin Ma, Jia Li, and Dacheng Tao. TriDet: Temporal Action Detection with Relative Boundary Modeling. In CVPR, pages 18857–18866, 2023. 2, 4, 5

[29] Jing Tan, Jiaqi Tang, Limin Wang, and Gangshan Wu. Relaxed Transformer Decoders for Direct Action Proposal Generation. In ICCV, pages 13506–13515, 2021. 2, 3

[30] Jing Tan, Xiaotong Zhao, Xintian Shi, Bin Kang, and Limin Wang. PointTAD: Multi-Label Temporal Action Detection with Learnable Query Points. In NeurIPS, page 1111, 2022. 2

[31] Mengmeng Xu, Chen Zhao, David S. Rojas, Ali Thabet, and Bernard Ghanem. G-TAD: Sub-Graph Localization for

Temporal Action Detection. In CVPR, pages 10153–10162, 2020. 2

[32] Shen Yan, Xuehan Xiong, Arsha Nagrani, Anurag Arnab, Zhonghao Wang, Weina Ge, David Ross, and Cordelia Schmid. UnLoc: A Unified Framework for Video Localization Tasks. In ICCV, pages 13577–13587, 2023. 2

[33] Le Yang, Houwen Peng, Dingwen Zhang, Jianlong Fu, and Junwei Han. Revisiting anchor mechanisms for temporal action localization. IEEE TIP, 29:8535–8548, 2020. 2

[34] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. CoCa: Contrastive Captioners are Image-Text Foundation Models. Transactions on Machine Learning Research (TMLR), 2022. 2

[35] Chenlin Zhang, Jianxin Wu, and Yin Li. ActionFormer: Localizing Moments of Actions with Transformers. In ECCV, pages 492–510, 2022. 1, 2, 4, 5, 6

[36] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In ICLR, 2021. 1

[37] Yuhan Zhu, Guozhen Zhang, Jing Tan, Gangshan Wu, and Limin Wang. Dual DETRs for Multi-Label Temporal Action Detection. In CVPR, pages 18559–18569, 2024. 2

TP^2-DETR: Unlocking Deformable DETR for Zero-Shot Temporal Action Proposal Generation with Temporal Feature Pyramids

Supplementary Material

6. Additional Experimental Results

We present the detailed experimental results below to provide additional information.

6.1. Visualization of Prediction Quality

Figure 5 shows the visualization of prediction quality. We plot scatter charts of inference results from a random 50%/50% split of the THUMOS14 dataset to compare our TP^2-DETR method with the baseline Deformable DETR. Each point represents a predicted proposal, where its horizontal position indicates the confidence score (i.e., actionness score) and the vertical position represents its best temporal IoU with the ground truth. Since high-quality proposals typically appear in the top-right region (indicating both high confidence and accurate localization), TP^2-DETR demonstrates a significantly denser distribution in that area compared to the baseline.

6.2. Additional Qualitative Comparisons

Figure 6 shows two additional sets of qualitative comparisons in the same format as Figure 4 but under different numbers of action instances. Figure 7 shows the precision-recall chart of the dense video (`video_test_0000464`) from which we calculate the mAP value.

6.3. Visualization of Salient Logits

As mentioned in Section 5, our salient head is trained independently as an auxiliary component to facilitate early supervision, but we observe that its predictions (i.e., salient logits) often overlap with those of the primary head (i.e., bounding box head). Figure 8 shows several examples.

6.4. Comparison of THUMOS14 and ActivityNet1.3

As discussed in Section 4, the two datasets exhibit significant differences. Figure 9 shows the distributions of action durations. THUMOS14's action durations are significantly shorter than those of ActivityNet1.3. Figure 10 shows the distributions of relative action durations of the two datasets. The THUMOS14 distribution highly concentrates in the first bin (0-5%), while the ActivityNet1.3 distribution has a U shape.

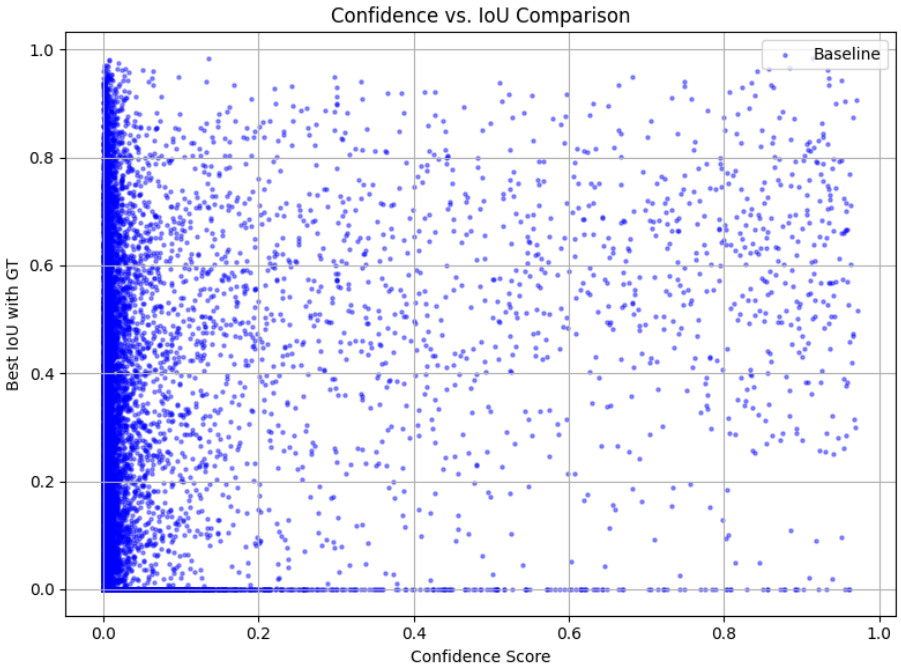
6.5. Class-Agnostic mAP Computation

Algorithm 2 outlines the detailed steps for computing the class-agnostic mAP reported in our manuscript.

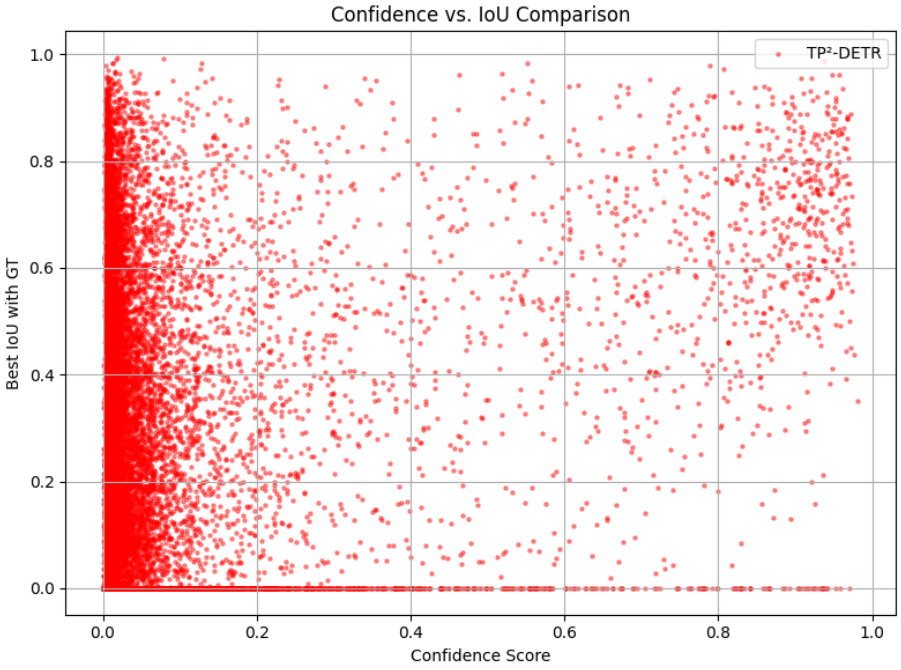
Algorithm 2: Class-Agnostic mAP Computation for Proposal Generation

Input:
G: Ground-truth proposals across all videos
P: Predicted proposals with actionness scores (i.e., confidence scores)
T: Set of tIoU thresholds
Output:
mAP: Mean Average Precision across all tIoU thresholds

foreach threshold t_{IoU} in T **do**
  Sort P by actionness scores in descending order;
  Initialize an empty set $matched\_gt$ to store matched ground truths;
  foreach prediction p in P **do**
    if exists $g \in G$ in same video s.t. $g \notin matched\_gt$ and IoU$(p, g) \geq t_{IoU}$ **then**
      Mark p as True Positive (TP);
      Add g to $matched\_gt$;
    end
    else
      Mark p as False Positive (FP);
    end
  end
  Compute precision and recall from TP/FP labels;
  Compute $\text{AP}_{t_{IoU}}$ as area under Precision-Recall curve;
end
Compute $mAP = \frac{1}{|T|} \sum_{t_{IoU} \in T} AP_{t_{IoU}}$;



(a) Baseline (Deformable DETR)



(b) TP^2-DETR

Figure 5. Visualizations of prediction quality (confidence vs. IoU with ground truth). Each point represents a predicted proposal. TP^2-DETR shows a denser concentration in the top-right region, indicating higher-quality proposals in terms of both localization and confidence. Results are reported on THUMOS14 using the 50%/50% split (split_id = 0).

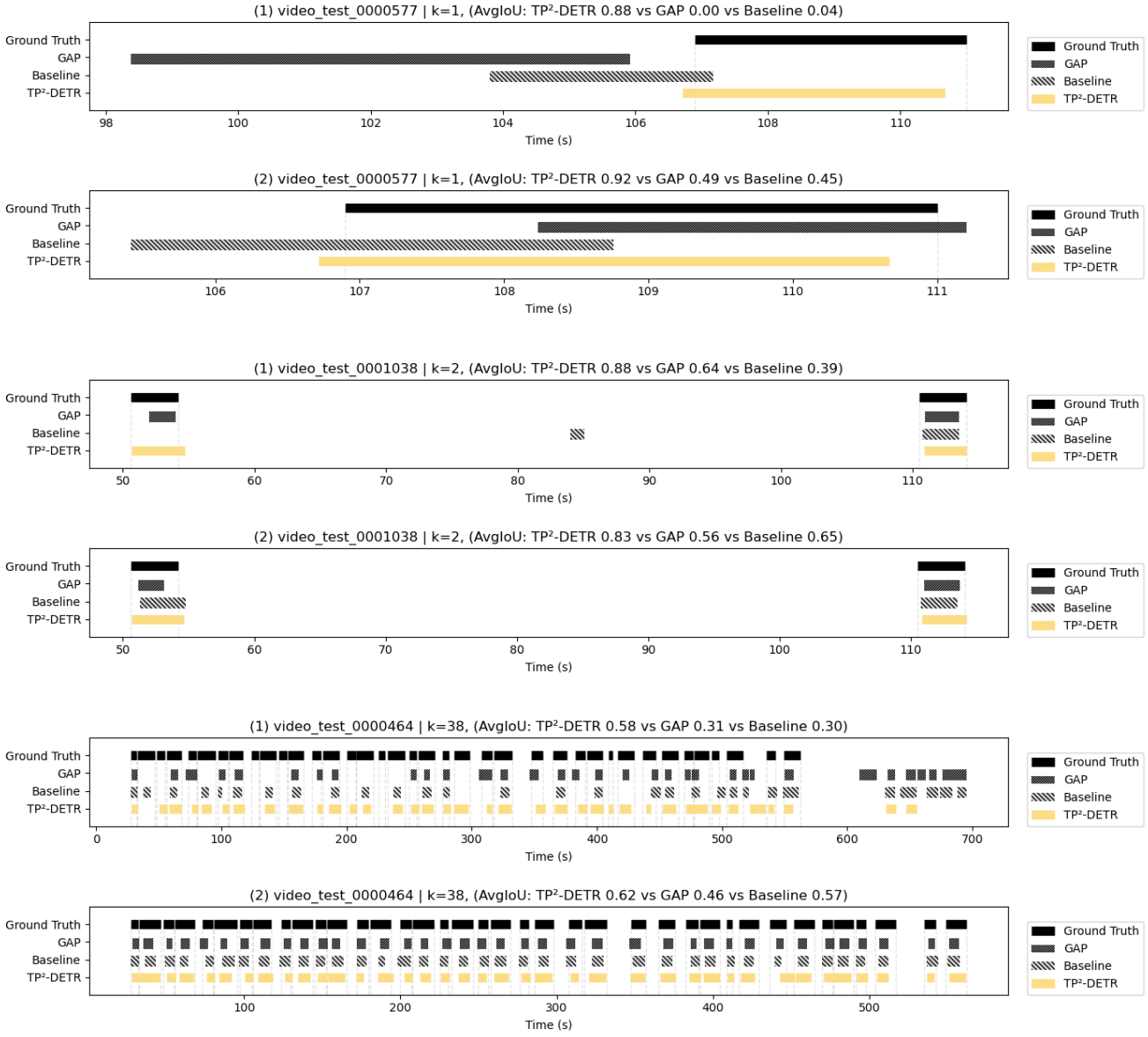


Figure 6. Visualizations of class-agnostic proposals as qualitative results. We present examples from sparse cases ($k = 1$, $k = 2$) to dense cases ($k = 38$), visualizing the top-k proposals selected in two ways: (1) by actionness scores (following the GAP approach) and (2) by bipartite matching using the same cost as in training. In addition to the intuitive metric AvgIoU for each case, we also provide a precision-recall curve for the dense video (`video_test_0000464`) to illustrate the mAP behavior. All results are from the 50/50 split (split_id = 0) on THUMOS'14.

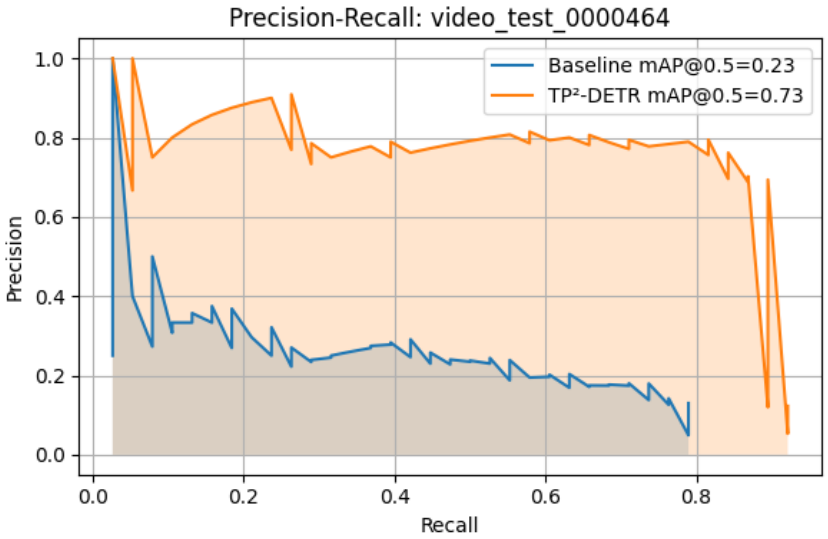


Figure 7. Precision-recall curve of the `0000464` video, whose proposal visualization results are shown in Figure 6. Here we show its precision-recall curve of the 38 actionness to illustrate the mAP behavior.

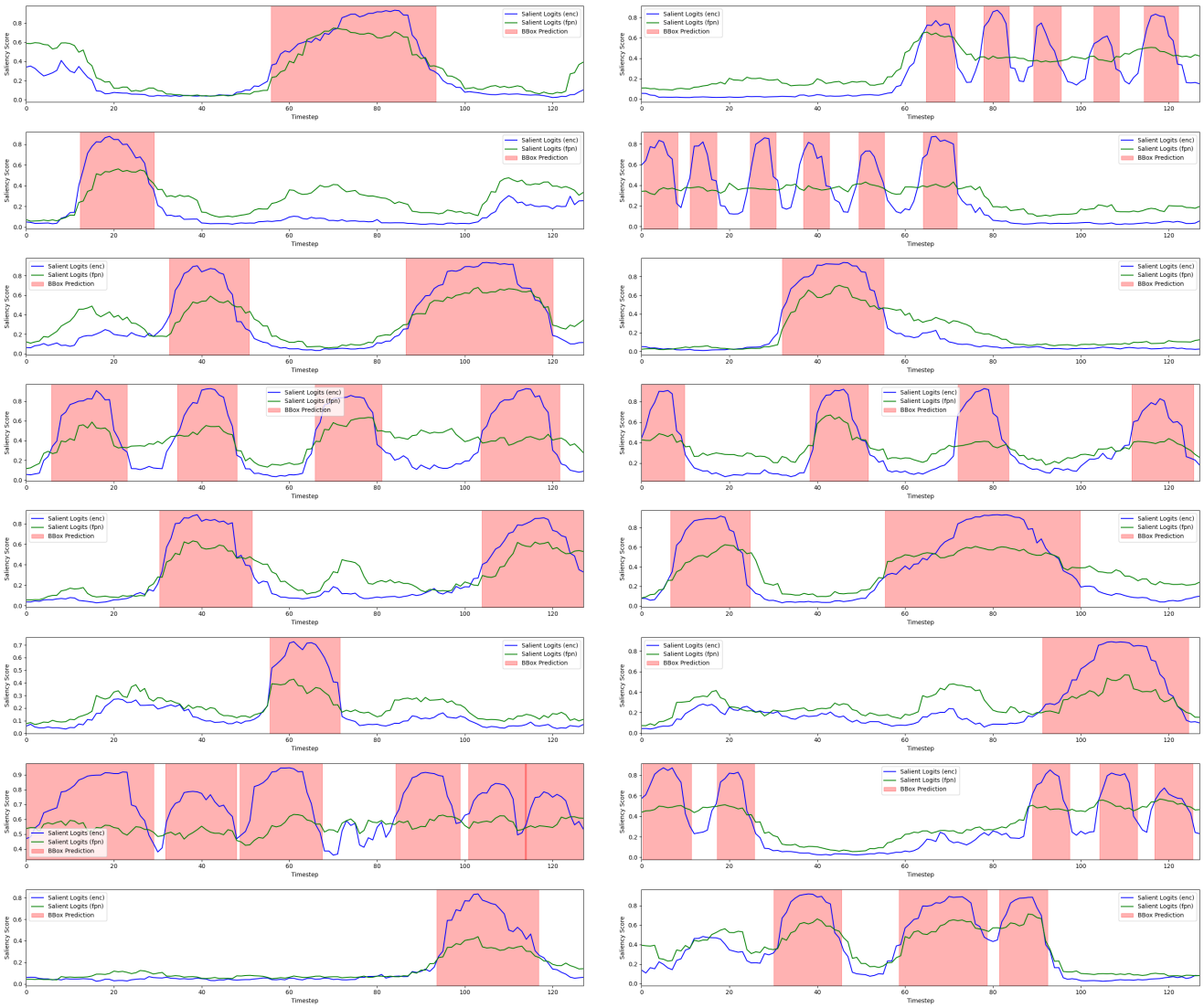


Figure 8. Visualization of salient logits (blue: from encoder, green: from temporal FPN) and predicted proposals (red) from matched queries for a randomly sampled training batch of videos on THUMOS'14 using the 50/50 split. Each subfigure shows the predicted saliency scores (y-axis) across timesteps (x-axis). The visualizations show that salient peaks—particularly those from the encoder output—often align with the predicted segments, despite the salient head being trained independently. This supports our hypothesis of potential cross-head consistency, which may be leveraged for future improvements.

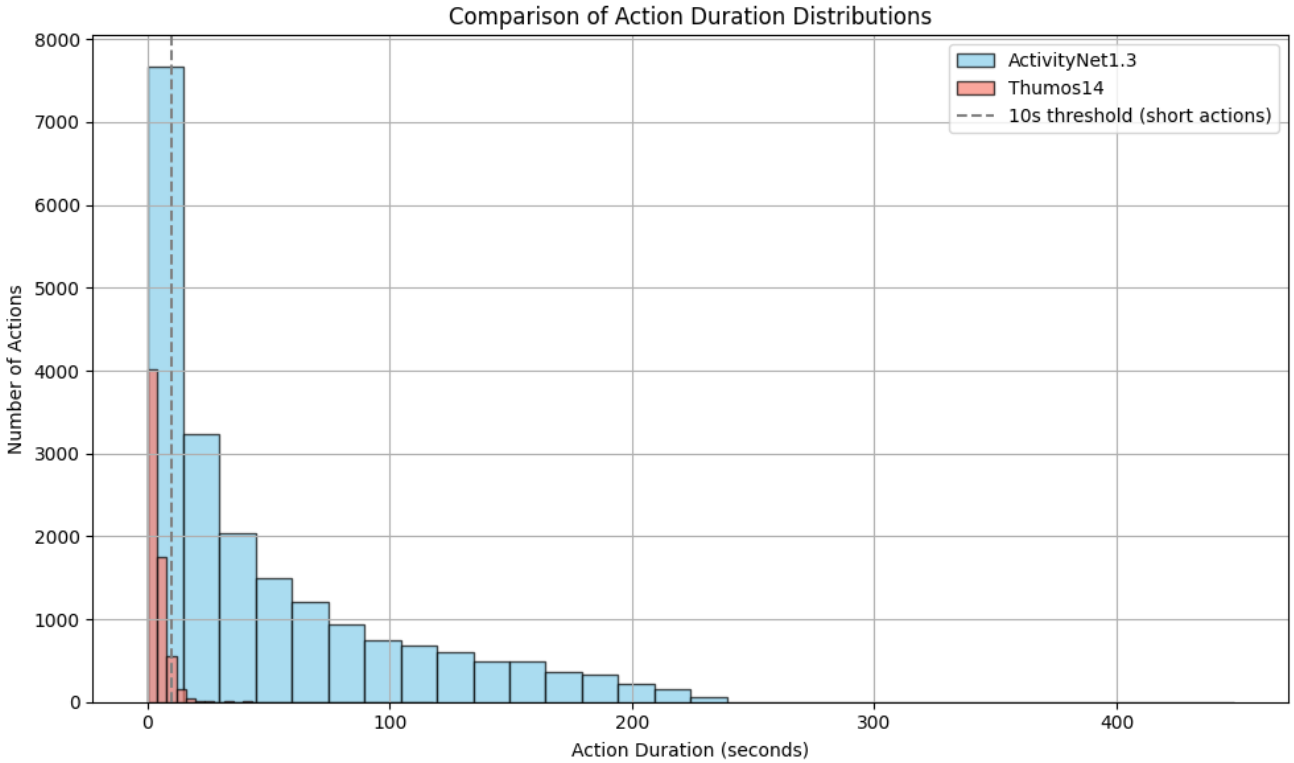


Figure 9. Histogram of action durations (seconds).

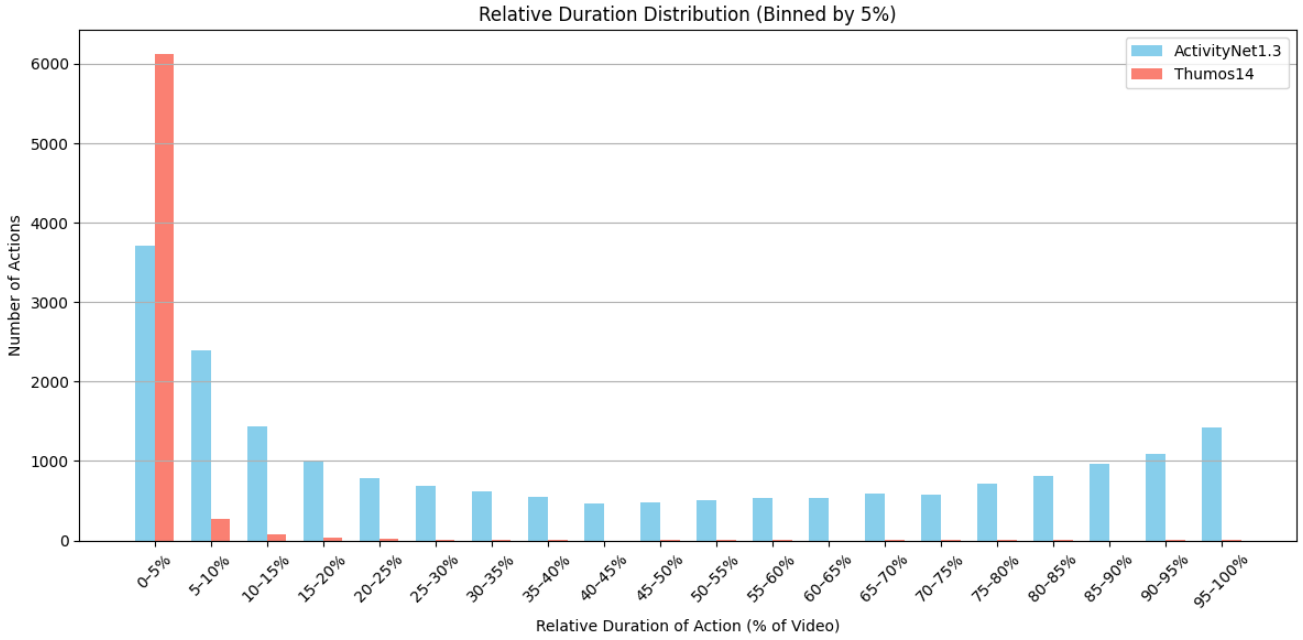


Figure 10. Relative view of action duration distributions on THUMOS14 and ActivityNet1.3. The x-axis indicates the relative duration of each action instance as a percentage of the corresponding video length. THUMOS14 exhibits a high concentration of short actions, with the majority lasting under 5% of video length. This observation aligns with our motivation to enhance short-action localization.