DCAPS: DUAL CROSS-ATTENTION COUPLED WITH STABILIZER FOR FEW-SHOT COMMON ACTION LOCALIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The goal of this paper is to localize action instances in a long untrimmed query video using just meager trimmed support videos representing a common action whose class information is not given. In this task, it is crucial not only to correctly align a temporal segment (proposal) of the query video and the support videos, but also to increase the compatibility among the support videos. The latter has been understudied, even though the context (e.g., background, camera angle) varies across the support videos. To address both points, we design a dual cross-attention coupled with a stabilizer (DCAPS). First, we develop an attention mechanism by cross-correlation, and apply it independently to each support video (with the query videos) in order to manage the heterogeneity among the support videos. Next, we devise an embedding stabilizer to increase the compatibility among the support videos. Then, the cross-attention is used again here to make the stabilized support videos attend and enhance the query proposals. Finally, we also develop a relational classifier head based on the query and support video representations. Hence, our contributions better utilize a few support videos for representing query proposals and thus attain precise common action localization. We show the effectiveness of our work with the state-of-the-art performance in benchmark datasets (ActivityNet1.3 and THUMOS14), and analyze each component extensively.

1 INTRODUCTION

Temporal action localization Escorcia et al. (2016); Gao et al. (2017); Wang et al. (2017); Nguyen et al. (2018); Singh & Lee (2017); Lee et al. (2021b); Luo et al. (2020) have been widely studied in fully or weakly-supervised manner. However, they require collecting massive videos labeled by action classes, and also only detect the action classes observed in training. Whereas, we aims to temporally localize action instances in a long untrimmed query video based on the a few trimmed support videos describing a common action class. As the testing action class is unseen in training and no ground-truth class cue is given, the only cue is the commonality of the few support videos.

In the few-shot perspective, due to the scarcity of support videos, aligning the support videos to the context of the query video is greatly crucial to truly exploit the action cues described in the support videos. Feng et al. (2018) re-weighted the query and support features and summed them up. Then, sequential representation is obtained by gating out time-steps where the query and support videos are unrelated. Under a more practical setting for the query videos with multiple action instances and more than one support video, Yang et al. (2020); Nag et al. (2021) manipulated the self-attention mechanism (Vaswani et al., 2017). Yang et al. (2020) exploited a series of the attention modules for iterative alignment of query and support features. Using a linear classifier fine-tuned for the given support features as a prototype, Nag et al. (2021) transformed the prototype to the query's context.

The iterative attention or fine-tuning the classifier for every set of supports is promising, but can increase the computational cost. Also, as exemplified in Fig. 1, though the support videos represent a common action class, their context (*e.g* background, camera angle) can be different. Hence, when all the support videos are transformed to query context simultaneously, the support videos cluttered by background are overly suppressed although they include useful information (Fig. 1(a)). In contrast, as in Fig. 1(b), we can leverage the support videos through attending each support video individually.

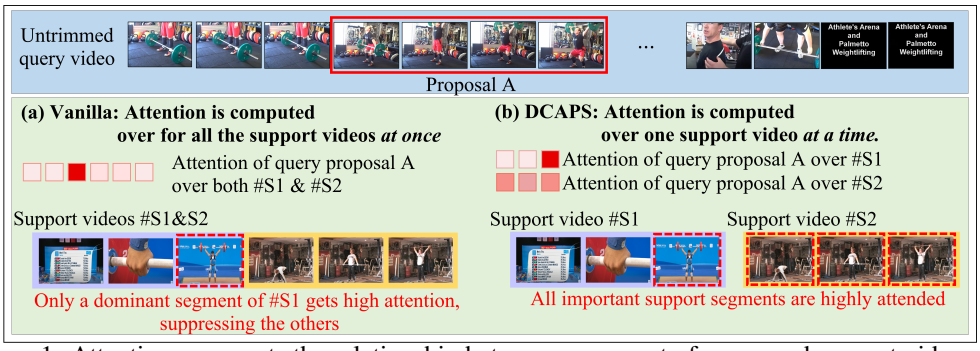


Figure 1: Attention represents the relationship between a moment of query and support videos. (a) Vanilla: attention of a query proposal is obtained simultaneously for the segments of all the support videos. (b) DCAPS (ours): the attention is computed for the segments of one support video, at a time. While important segments of support video #2 cannot be attended by the related query proposal in (a), they are appropriately transformed to the context of the query in (b).

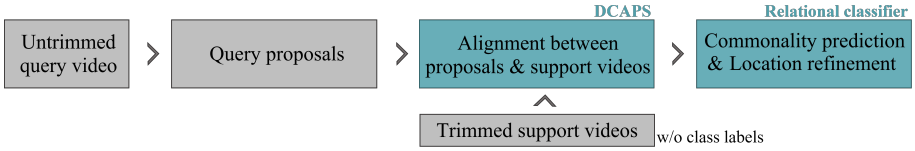


Figure 2: Overall process of few-shot common action localization in our work.

Motivated by this, we propose a novel few-shot common action localization method. Overall process is outlined in Fig. 2. First, the long untrimmed query video is split to query proposals, and they are aligned with the support videos. Then, the proposals representing the common action class are detected with temporal location refinement.

For the query-support alignment, we develop Dual Cross-Attention coupled with embedding Stabilizer (DCAPS). In DCAPS, the cross-correlation with a learnable weight matrix is exploited for an attentional transform of the support and query features. Instead of attending heterogeneous features reciprocally with a single cross-attention weight matrix (Lee et al., 2021a; Lu et al., 2016), two weight matrices are used for query-to-support and support-to-query attention, respectively. We first transform support videos to query context by its cross-correlation with the query features. As aforementioned, the support videos are individually attended to effectively emphasize the most informative features in each support video. Then, for stable complementary use of all the individually attended support video features in the following query attention, we design an embedding stabilizer that makes the attended support video features aligned with each other in the latent space. With the stabilized support video features, we attend the query video features by cross-correlation. Detailed description is in Sec. 3.2.

For the commonality prediction and refinement, we design a relational classifier which consists of an action classifier and an auxiliary relational module. (Sec. 3.3). To prevent overfitting and boost performance, the latter matches the support and query video features using pseudo-label cues as we avoid the use of class labels here. In testing, this module is not used.

Our major contributions: (i) We develop a cross-attention mechanism to enhance the representation of queries by support videos and vice versa. (ii) We attend each support video via query video individually and design a stabilizer to make the attended support features more compatible. (iii) We develop a relational classifier consisting of an action classifier and an auxiliary relational module. The latter is only needed during training. (iv) Extensive experimental analysis is done for the components of our method on two benchmark datasets, where we achieve state-of-the-art performance.

2 Related Work

Temporal action localization The goal is to predict the temporal boundary and the label of action instances in untrimmed videos. In the fully-supervised case where temporal annotations are given during training, several works tried to obtain better temporal proposals (Heilbron et al., 2016; Gao et al., 2017; Lin et al., 2018), while others improved temporal searching Yeung et al. (2016); Yu & Yuan (2015) or classifiers (Shou et al., 2017). R-C3D (Xu et al., 2017) proposed an end-to-end

trainable activity detector based on Faster-RCNN (Ren et al., 2015). GNNs (Zhao et al., 2021; Xu et al., 2020b) are recently used to capture the temporal relationship among proposals/snippets. In the weakly-supervised setting, as only video-level annotations are given, most methods have aimed to obtain discriminative snippet-level activations and conduct post-processing to localize action instances. A co-activity similarity loss to enforce the feature similarity for video pairs with a common class was introduced in (Paul et al., 2018), and videos are segmented into interpretable fragments in (Jain et al., 2020). Some works focused on distinguishing action and near-action/background snippets exploiting variational auto-encoder (Shi et al., 2020), entropy maximization (Luo et al., 2020), and an additional class-agnostic model (Ma et al., 2021). Also, to better localize difficult snippets, multi-modal (audio-visual) fusion (Lee et al., 2021a) or a contrastive loss with easy foreground/background snippets (Zhang et al., 2021) are devised. Although those fully- and weakly-supervised methods attained large progress in this field, the learned models can only localize activity categories observed in the training dataset.

Few-shot temporal action localization Yang et al. (2018) pioneered few-shot action localization, where a few (or only one) positive and several negative labeled videos steer the localization via an end-to-end meta-learning strategy. Xu et al. (2020a) also temporally localized an action from a few positive labeled and several negative labeled videos. They adopted a region proposal network Ren et al. (2015) to produce proposals with flexible boundaries. Zhang et al. (2020) performed few-shot action localization where video-level annotations are needed. They constructed a multi-scale feature pyramid to directly produce temporal features at variable scales. Unlike those works, few-shot common action localization is less tied up with the need for labels. It localizes the action instances in a long untrimmed query video according to the commonality between the query and support inputs. Assuming a query with only one common action instance, Feng et al. (2018) computed the probabilities for starting, ending, inside, or outside of action instances at every time-step, and decided the window with the highest joint probability as the action instance. Extending to the query videos with multiple action instances of the same action class, Yang et al. (2020) generated action proposals, and then classified the proposals and regressed their temporal locations. Nag et al. (2021) set a linear classifier itself as a prototype, which needs fine-tuning by support videos for every target action, then the prototype is cross-attended by query proposals using multiple self-attentions.

Cross-attention To leverage the relationship between two heterogeneous representations, diverse cross-attention schemes have been devised. Inspired by self-attention (scaled dot-product of key, query, and value), (Wei et al., 2020) applied the scaled dot-product operation for the concatenation of image and text features for VQA. Also, Kim et al. (2021) attended student using key and value of teacher for knowledge propagation. Several works exploited the cross-correlation between the heterogeneous representation as the attention weights for image and sentence matching in visual question answering (VQA) (Lee et al., 2018; Kim et al., 2017), query and prototype matching in prototypical few-shot learning (Hou et al., 2019), and audio-visual fusion (Praveen et al., 2022). Here, we exploit the cross-attention to improve the matching between query and support video clips as well as to better utilize multiple support videos in the context of few-shot action localization.

3 METHOD

In this section, we describe the proposed method including DCAPS (Dual Cross-Attention couPled with Stabilizer) and relational classifier for few-shot common action localization.

Problem statement Given an untrimmed query video V_Q and L trimmed support videos $\{V_S^1, V_S^2, \ldots, V_S^L\}$, the goal is to train a network (which consists of backbone g, proposal-net h, DCAPS and relational classifier f) to temporally localize action instances in the testing query videos based on the commonality of the testing support videos whose action classes are same and unseen during training. Note that any ground-truth class cues are not given even during training.

In training, we resort to a meta-learning strategy. Here, the action classes in the training set (C_{train}) and those of the testing set (C_{test}) have no overlapping. Also, to simulate the few-shot configuration of support and query videos that will be encountered at the testing phase, we exploit episode-based training. Specifically, in a training iteration, we compose an episode as a tuple of a query and L support videos $\{(V_Q, V_S^1, V_S^2, \ldots, V_S^L)\}$ from a randomly selected class of C_{train}. In our work, we set L to 1 or 5. Formally, the objective function is represented by

$$\arg\min_{g,h,f} \mathbb{E}_{(V_Q,\{V_S^l\}_{l=1}^L)\sim C_{\text{train}}} [\, \mathcal{L}(\mathcal{Y}, f(h(g(V_Q)), g(\{V_S^l\}_{l=1}^L)))\,], \tag{1}$$

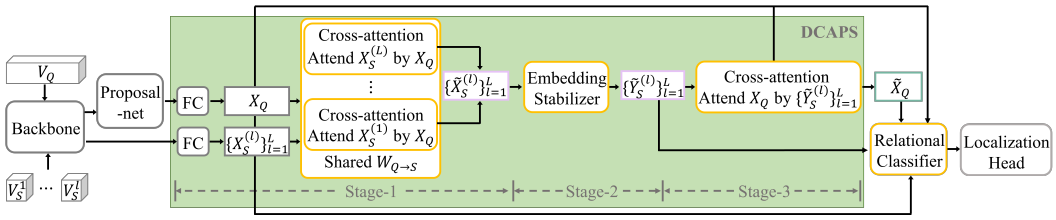


Figure 3: An overview of the proposed method. The segments of lth support video, $X_S^{(l)}$, are attended by query proposals X_Q, individually. Next, all the attended support segments are stabilized and give attention to the query proposals. $\tilde{X}_Q$ is the resulting attended query proposals. Then, the relational classifier predicts commonality scores of the proposals for the action of support videos, and the localization head finalizes action localization removing redundant proposals.

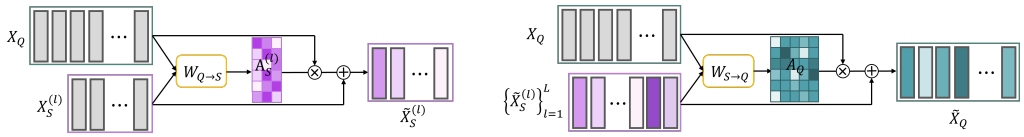


Figure 4: Cross-attention. Left: attention of lth support segments by the query proposals. Right: attention of the query proposals by all the attended support segments.

where $\mathcal{Y}$ denotes the set of temporal positions for the ground-truth action segments of the interest in V_Q. $\mathcal{L}$ is the loss function.

3.1 Overall Framework

Fig. 3 depicts the overall framework of our method. To obtain the initial representations of the input query and support videos, we follow the preprocessing of (Yang et al., 2020). The backbone network g (Tran et al., 2015) generates video representations for each input. Then, for the query video, the proposal subnet h (Xu et al., 2017) yields potential temporal action instances $Q = \{q_i\}_{i=1}^{N_Q}$, called action proposals, with diverse temporal lengths (details are in Appendix D). N_Q is the number of the proposals, and q_i denotes i^{th} action proposal representation. For every l^{th} support video, we uniformly split each support video into N_S temporal segments by $S^{(l)} = \{s_i^{(l)}\}_{i=1}^{N_S}$. Then, DCAPS aligns Q and $S^{(l)}$'s. Next, in the relational classifier, the action classifier detects the action proposals with the target common action, and yields their temporal offsets to refine the start and end time of the proposals. In training, learning an auxiliary relational module in parallel with the action classifier is beneficial. In testing, the auxiliary relational module is discarded, and the localization head suppresses redundant proposals which is explained in Sec. 4.1.

3.2 DCAPS

The blue box of Fig. 3 illustrates three stages of DCAPS (More details are in Appendix E): attending support videos with the query video, stabilizing representation of support videos, and attending the query video with the stabilized support videos. Here, as the query proposal q and the support segment s have different temporal granularity, we use different cross-attention weights for query to support and support to query, respectively. We first introduce the cross-attention mechanism to individually attend support features. Next, we explain the (embedding) stabilizer devised to increase the compatibility between the individually attended support features. Then, we present the way to attend the features of query proposals via the cross-attention with all the stabilized support features. These enhanced features serve as crucial inputs to the following relational classifier (Sec. 3.3).

Cross-attention for support videos: We first enhance the support features in consideration of the context of the query video. To this end, we develop a cross-attention mechanism. When we suppose a vanilla approach such as Fig 1(a), the attention weight is obtained at once based on the relationship between the query proposals and all the support videos. In this case, a support video, which is relatively less relevant to the query video than the other support videos, gets tiny attention weights and cannot represent its action information in the context of the query video. As depicted in Fig. 4 (left), to utilize the query's context faithfully, we individually attend the support videos.

Table 1: Pairwise cosine dist. between two support segment features from different support videos ($L = 5$).

Features	$\{\tilde{X}_S^{(l)}\}$ (before stabilizer)	$\{\tilde{Y}_S^{(l)}\}$ (after stabilizer)
Avg cosine dist.	0.018	0.005

Table 2: Pairwise cosine dist. between positive and negative query proposal features in the same video.

Features	w/o stabilizer	w/ stabilizer
Avg cosine dist.	0.091	0.104

In specific, for lth support video, we first encode the backbone features of the query proposals Q to $X_Q \in \mathbb{R}^{d \times N_Q}$ and the support segments $S^{(l)}$ to $X_S^{(l)} \in \mathbb{R}^{d \times N_S}$, with a fully-connected (fc) layer. The columns of X_Q and $X_S^{(l)}$ represent the encoded d-dimensional features of the query proposal and support segment, respectively. After that, we compute the cross-correlation of X_Q and $X_S^{(l)}$ to measure their relevance. To reduce the gap of the heterogeneity between the query and support videos, we use a learnable weight matrix $W_{Q \to S} \in \mathbb{R}^{d \times d}$ and compute the cross-correlation as

$$\Lambda_S^{(l)} = X_Q^T W_{Q \to S} X_S^{(l)} \tag{2}$$

where $\Lambda_S^{(l)} \in \mathbb{R}^{N_Q \times N_S}$. Note that each column of X_Q and $X_S^{(l)}$ are l_2-normalized before the cross-correlation computation, and the learnable weight $W_{Q \to S}$ and the fc layer are shared across all l's *i.e.* all the support videos.

In the cross-correlation matrix, a high correlation coefficient means that the corresponding proposal and segment features are highly relevant. Accordingly, ith row of $\Lambda_S^{(l)}$ corresponds to the relevance of ith query proposal to the N_S support segments. Then, from row-wise soft-max of $\Lambda_S^{(l)}$, we can obtain cross-attention weights $A_S^{(l)}$ where each row represents the relative relevance of a query proposal to the support segments.

The attention-weighted proposal features are summed to the corresponding support segment feature. This is to ensure that the meaningful action information of the support video is well-preserved while applying the cross-attention. Formally, the attended support segment features $\tilde{X}_S^{(l)}$ are obtained by

$$\tilde{X}_S^{(l)} = X_Q A_S^{(l)} + X_S^{(l)}. \tag{3}$$

Note that, through the attention weights, the query proposal injects its context to the support segments proportionally to their relevance *i.e.* more to the highly relevant support segments. With this, the support segment features are enhanced to better guide the information about the target action to the query proposals in the later stage of attending queries of Fig. 4 (right). We provide the detailed structure of the cross-attention in the supplementary materials.

Stabilizer: As the support videos represent a common action, the support segment features should be closely located in a latent feature space. However, the support videos may be inherently heterogeneous due to the difference of background, camera angle, etc. Moreover, the cross-attention can also cause heterogeneity among support segments. If a support segment is the only informative (but not much) segment in a support video, the cross-attention overly highlights it. Otherwise, when most support segments are moderately important in another support video, they are evenly attended. These heterogeneities can degenerate the effect of attention on query videos.

Therefore, before attending the query proposals with the support segments from different support videos in the next stage, we re-calibrate the attended support segment features. To this end, we pass all the attended support segments through a common multi-layer perceptron (MLP) with bottleneck-structured two fc layers, t_1 and t_2, with ReLU activation.

$$\tilde{\mathbf{y}}_{S,i}^{(l)} = t_1(\text{ReLU}(t_2(\tilde{\mathbf{x}}_{S,i}^{(l)}))) \tag{4}$$

where $\tilde{\mathbf{x}}_{S,i}^{(l)}$ and $\tilde{\mathbf{y}}_{S,i}^{(l)}$ are the features of ith temporal segment of lth support video before and after passing through the stabilizer, respectively.

Table 1 shows the averaged pairwise cosine distance between support segment features in different support videos when $L = 5$. After passing through the MLP, the features get closer together in the latent feature space (see the reduced distances), and this promotes the compatibility of the attended support segment features from different support videos. Hence, we call this MLP a stabilizer. Also, in Table 2, the attended query proposal features are more discriminative when the attended support segment features are stabilized. More analyses are in Sec. 4.3 and Appendix C.3& C.4.

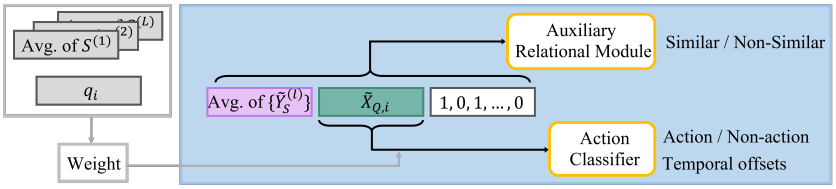


Figure 5: Relational classifier consists of auxiliary relational module and action classifier.

Cross-attention for query: In this stage, we attend query videos with the support videos, using the cross-attention again. This is shown in Fig. 4 (right). To mine the richer information for target action from all the support videos, we attend query proposal features using the entire stabilized support segment features $\tilde{Y}_S = \{\tilde{Y}_S^{(1)}, \tilde{Y}_S^{(2)}, \ldots, \tilde{Y}_S^{(L)}\} \in \mathbb{R}^{d \times LN_S}$. Similar to attending support features, we exploit the cross-attention to obtain the attended query proposals $\tilde{X}_Q$ with a learnable weight matrix $W_{S \to Q} \in \mathbb{R}^{d \times d}$ by

$$\tilde{X}_Q = \tilde{Y}_S A_Q + X_Q \tag{5}$$

where A_Q is attention weight computed by row-wise soft-max of cross-correlation, $\Lambda_Q = \tilde{Y}_S^T W_{S \to Q} X_Q$. From this cross-attention, the query proposals, which have high relevance to crucial support segments, better delineate the target action. The cross-attention presented here may resemble the one proposed in (Lee et al., 2021a). But dissimilar with our dual cross-attention, to fuse the different modalities, (Lee et al., 2021a) attended them reciprocally with a single learned weight matrix. Whereas, our cross-attention between the support videos and the query video is applied at different stages, learning two weight matrices (query-to-support and vice versa).

3.3 Relational Classifier

Up to this point, we described how DCAPS generates better representations of query proposals and support segments. Here, we explain the way to obtain the final decision from the attended representations. Fig. 5 depicts our relational classifier including an action classifier and an auxiliary relational module. The auxiliary module facilitates learning how to understand the relationship between query proposals and a target action with pseudo action class cues. As this auxiliary relational module is discarded in the deployed network, no action class label is required during testing.

Auxiliary relational module: In episodic few-shot learning, an auxiliary classifier can be co-trained to categorize an input into one of the ground-truth classes rather than the target classes of an episode. This is beneficial for feature extractors to prevent overfitting, stabilize the training, and boost the performance (Oreshkin et al., 2018; Kang et al., 2021).

However, since the ground-truth action classes are not available in our task, we utilize pseudo action classes. As in Fig. 5, we develop the auxiliary relational module which takes as an input the concatenation of the attended query proposal (green), support prototype (purple), and a pseudo-class identifier (white). The support prototype is the average of all the attended support segments. The pseudo-class identifier is a multi-hot vector with 1 for the pseudo action classes present in the support videos and 0 otherwise. As action class annotations are unavailable, the pseudo action classes are simply obtained by k-means clustering of the pre-trained backbone features extracted from all the ground-truth positive action instances in the training set. Then, these pseudo action classes can provide the cue to capture the commonality of the support videos with the same action across episodes (its effect is addressed in Appendix C.1). Hence, joint training with the auxiliary module assists the action classifier to learn to more correctly find the common actions in the query video without getting distracted by the non-target actions. The auxiliary relational module predicts whether the query proposal and the prototype are similar or not. Note that this module is only needed for training and discarded in the testing phase as the training and testing classes are mutually exclusive.

Action classifier: The action classifier consists of two parallel fc layers. Taking only the query proposal features $\tilde{X}_Q$ as inputs, the action classifier computes two outputs. The first is soft-max activated and decides if a query proposal contains target action or not. The second regresses the temporal offsets from the corresponding ground-truth. In addition, following (Yang et al., 2020), each attended query proposal is weighted by the distances between the non-attended query proposal and support video features (video-wise average of original support segment features).

3.4 LOSS FUNCTIONS

To optimize the entire network (backbone + DCAPS + relational classifier), as in (Yang et al., 2020), we use loss terms for both support-agnostic and -conditioned parts. In the support-agnostic part, the binary cross-entropy loss $\mathcal{L}_{\text{cls}}^{\text{ag}}$ predicts if the proposal contains any activity or not, and the smooth l_1 regression loss $\mathcal{L}_{\text{reg}}^{\text{ag}}$ optimizes the relative displacement between proposals and ground-truths.

In the auxiliary relational module of the support-conditioned part, we simply use the binary cross-entropy loss $\mathcal{L}_{\text{aux}}$ to predict if the proposal and support videos are similar or not. In the action classifier of the support-conditioned part, similar to the support-agnostic part, the classification loss $\mathcal{L}_{\text{cls}}^{\text{co}}$ predicts if the proposal includes the same common action of support videos, and the regression loss $\mathcal{L}_{\text{reg}}^{\text{co}}$ optimizes the relative displacement between the proposals and the ground-truths. Also, we design a pairwise ranking loss to add constraints to the action classifier. Considering a pair of proposals q_i and q_j, where at least one of them is a positive query proposal, we let the proposal with the larger IoU (intersection over union) for the ground-truth action instance have a higher soft-max score for the action class. Formally, the pairwise ranking loss $\mathcal{L}_{\text{rank}}$ is represented by

$$\mathcal{L}_{\text{rank}} = \frac{1}{N_{\text{pair}}} \sum_{(i,j)} (\Delta\text{IoU}_{ij} - \Delta p_{ij}^{\text{action}})^2 \tag{6}$$

where N_{pair} is the number of considered proposal pairs. ΔIoU_{ij} is the difference between IoUs of q_i and q_j with their corresponding ground-truths, and $\Delta p_{ij}^{\text{action}}$ is the difference of soft-max predictions for the proposals on the action class from the action classifier. Note that to relax the relationship between IoU and soft-max prediction, we give temperature to the soft-max predictions in $\mathcal{L}_{\text{rank}}$. Finally, total loss $\mathcal{L} = \mathcal{L}_{\text{cls}}^{\text{ag}} + \mathcal{L}_{\text{reg}}^{\text{ag}} + \mathcal{L}_{\text{cls}}^{\text{co}} + \mathcal{L}_{\text{reg}}^{\text{co}} + \mathcal{L}_{\text{aux}} + \lambda\mathcal{L}_{\text{rank}}$ (more details in Appendix A).

4 EXPERIMENTS

4.1 DATASETS AND EVALUATION

To evaluate few-shot common action localization, we use the revised versions (Feng et al., 2018; Yang et al., 2020) of ActivityNet1.3 (Caba Heilbron et al., 2015) and THUMOS14 (Idrees et al., 2017). In the revised, there are two cases depending on queries: single-instance and multi-instance. We address 1- or 5-shot settings in both cases. For meta-learning strategy, the entire action classes of each dataset are split to 80% for training, 10% for validation, and 10% for testing. For a fair comparison, we follow the data configuration of (Yang et al., 2020), which will be described in the following paragraphs. Further details are in Appendix D.

Common single-instance: For both datasets, videos with multiple actions are divided into independent videos where each video contains just one action instance and background. Then, videos longer than 768 frames are discarded. If a video is selected as a support video, its foreground action instance is only used as the support input. For ActivityNet1.3, there are 10,035 and 2,483 videos for training and validation+testing, each. The average video length is 89.0s. For THUMOS14, there are 3,580 and 775 training and validation+testing videos, respectively. The average length is 11.4s.

Common multi-instance: In real-world scenarios, the lengths of query videos are usually not constrained, and the query videos may contain multiple action instances. Hence, we exploit the original videos of ActivityNet1.3 and THUMOS14 without any processing for query videos. Support videos are still trimmed ones. For ActivityNet1.3, the numbers of videos are 6,747 and 1,545 for training and validation+testing, respectively. The average video length is 148.2s. For THUMOS14, there are 1,664 training videos and 323 validation+testing videos. Average video length is 230.6s.

Inference: In testing, the backbone-generated query proposals are refined by non-maximum suppression (NMS) with a threshold 0.7. Also, following (Yang et al., 2020), if the query video is longer than 768 frames, we generate the multi-scale segments (Shou et al., 2016). We slide windows with sizes of 512 and 768 frames along the temporal axis with 75% overlap. The generated proposals of the windows go through the NMS to remove redundant proposals. Then, the selected proposals are fed into our DCAPS and relational classifier. Finally, we perform NMS (threshold 0.3) for the regressed proposals based on the outputs of the relational classifier to remove redundant ones.

Evaluation: We measure the temporal action localization performance with mean Average Precision (mAP). A prediction is correct when it has the correct foreground/background classification and has IoU with its ground-truth larger than a threshold. The threshold is set to 0.5 unless specified.

Table 3: Comparison with state-of-the-arts in terms of mAP@0.5.

Method	*ActivityNet1.3*				*THUMOS14*			
	Single-instance		Multi-instance		Single-instance		Multi-instance	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
(Feng et al., 2018)	43.5	n/a	31.4	n/a	34.1	n/a	4.3	n/a
(Hu et al., 2019)	41.0	45.4	29.6	38.9	-	42.2	-	6.8
(Yang et al., 2020)	53.1	56.5	42.1	43.9	48.7	51.9	7.5	8.6
(Yang et al., 2021)	57.5	60.6	**47.8**	48.7	-	-	-	-
(Nag et al., 2021)	55.1	63.0	44.1	48.2	49.2	54.3	7.3	10.4
Ours	**61.3** $\pm$ **1.1**	**65.1** $\pm$ **1.2**	47.2 $\pm$ 1.0	**52.2** $\pm$ **0.8**	**53.3** $\pm$ **1.3**	**57.9** $\pm$ **1.0**	**9.2** $\pm$ **0.3**	**14.0** $\pm$ **0.4**

Table 4: Computational cost of query-support fusion modules.

	(Yang et al., 2020)		(Nag et al., 2021)		DCAPS (Ours)	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
#Param / MACs	5.25M / 0.36G	5.25M / 0.47G	0.53M / 0.66G	0.53M / 3.28G	1.05M / 0.10G	1.31M / 0.27G

4.2 Comparative assessment

We report the performance of compared methods from the literature. Due to the variance of k-means clustering of the pseudo label generation, we report the average of three runs for our method.

For AcitivityNet1.3, the left of Table 3 demonstrates the comparative results on both common single- and multi-instance cases when $L = 1$ or 5. (Feng et al., 2018; Yang et al., 2020; Nag et al., 2021) were developed for the common temporal action localization (our task) in videos, and (Hu et al., 2019) was for the common object detection in images. Compared to them, our method shows notably higher performance for all the settings ((Feng et al., 2018) was designed to use one support video). Specifically, in the single-instance case, we outperform those methods by at least 6.2% and 2.1% in the 1- and 5-shot settings, respectively. In the multi-instance, the margins are at least 3.1% and 4.0% for 1- and 5-shot, each. Note that (Yang et al., 2021) was developed for object-level common temporal action localization. Although object-level localization is more challenging, it requires further information to learn their model such as the coordinates of foreground object bounding boxes. Hence, they have advantages in the frame-level localization task. Even so, our method yields better results than (Yang et al., 2021) in most of the settings. From the results of the 1-shot setting, we show the effectiveness of our cross-attention and the relational classifier. Also, in the 5-shot setting, the score margins to the existing works are larger than those of 1-shot in both single- and multi-instance. This means that the individual attention of support videos with stabilizer is helpful to mine the knowledge for a common action from multiple support videos. Unlike ActivityNet1.3, THUMOS14 includes shorter action instances which make correct localization more difficult. Nevertheless, we outperform the compared methods over all the settings.

4.3 Component Analysis

We analyze our method on ActivitNet1.3. More studies are in Appendix C and F.

Efficiency of DCAPS: We analyze the efficiency of our DCAPS comparing with (Yang et al., 2020; Nag et al., 2021). In Table 4, for query-support fusion modules, we compute the number of parameters and MACs per a single query and 1 or 5 support videos. Compared to (Yang et al., 2020), the proposed DCAPS requires significantly lower computation costs in both 1- and 5-shot settings. Though Nag et al. (2021) uses a small number of parameters, their iterative back-propagation to fine-tune prototype for given query and support videos drastically increases MACs. Whereas, DCAPS is highly efficient in both parameters and MACs. Hence, DCAPS effectively and efficiently utilizes a few support videos for common action localization.

Impact of cross-attention: We first evaluate the efficacy of our cross-attention. To this end, we compare our DCAPS (D-0 in Table 5) to the progressive cross-attention (Yang et al., 2020) and the multi-head cross-attention with fine-tuned (50-100 iterations) prototype Nag et al. (2021). For a fair comparison, the pairwise ranking loss and auxiliary relational module are not used (the stabilizer is also not used in the 1-shot setting) in D-0. DCAPS yields mAP gain of at least +3.6% and +0.5% in 1-shot (55.1% *vs* 58.7%) and 5-shot (63.0% *vs* 63.5%), respectively. Hence, we conclude our dual cross-attention more effectively enhances the query and support videos in this task.

Cross-attention individual *vs* aggregated: To study the effectiveness of the individual cross-attention on multiple support videos, we compare our individual cross-attention approach with the aggregated cross-attention. In the aggregated one, we concatenate all the support segments of the

Table 5: Analysis on the relational classifier.

Method	$\mathcal{L}_{\text{rank}}$	Auxiliary rel. module	mAP (%)	
			1-shot	5-shot
D-0			58.7	63.5
D-1	✓		60.2	64.1
D-2		✓	59.7	63.8
D-3	✓	✓	61.3	65.1

Table 6: Analysis on the stage-1 individual cross-attention comparing with the aggregated cross-attention on the 5-shot setting.

	Cross-att (support)	Single-inst.	Multi-inst.
w/o Stabilizer	Aggregated	57.2	44.5
	Individual	60.9	49.4
w/ Stabilizer	Aggregated	57.4	43.4
	Individual	65.1	52.2

Table 7: Analysis on the stabilizer for average pairwise cosine distance of segment features from different support videos ($L = 5$). $\{\tilde{X}_S^{(l)}\}$ and $\{\tilde{Y}_S^{(l)}\}$ are features before and after the stabilizer.

	Baseline	l_1-norm	l_2-norm	Instance norm	Batch norm	Stabilizer (Ours)	
						1fc	2fc
$\{\tilde{X}_S^{(l)}\}$	0.038	0.201	0.041	0.104	0.117	0.021	0.018
$\{\tilde{Y}_S^{(l)}\}$	n/a	0.201	0.041	0.103	0.058	0.011	0.005
mAP (%)	56.7	39.9	54.1	52.8	55.7	63.5	65.1

entire support videos, and simultaneously attend them via query proposals. Also, for more extensive analysis, we compare the individual and aggregated cross-attentions under with or without the stabilizer. As in Table 6, regardless of the use of the stabilizer, the aggregated attention degrades the localization performance, and the stabilizer has no effect on the aggregated attention in the single-instance case. Whereas the individual cross-attention shows better performance than the aggregated one improving mAP by 3.7% when the stabilizer is not used (w/o stabilizer), this gap is further increased to 7.7% when the stabilizer is used. A similar tendency is observed in the multi-instance case as well. Therefore, we conclude that independent of the stabilizer, the dual cross-attention itself is useful to thoroughly mine the action information from each support video.

Stabilizer: Based on the performance boost (60.9% to 65.1%) by the stabilizer in Table 6, we can infer that the stabilizer helps to improve the compatibility of support segment features from different support videos. To identify this, we measure the pairwise cosine distances between two support segment features from different support videos, before and after passing through the stabilizer. The average of these distances is reported in Table 7. In terms of the averaged cosine distance, we verify our stabilizer by replacing it with w/o any processing on the attended support features (baseline) and several normalization schemes: simple l_1-, l_2-normalization, and normalization with learnable parameters (batch normalization (Ioffe & Szegedy, 2015) or instance normalization (Ulyanov et al., 2016)). The l_1- or l_2-normalization does not affect the cosine distance but removes the support-specific useful information (resulting in low performance). Though the batch and instance normalization techniques reduce the averaged cosine distance, the performance gain is marginal. Whereas, with quite low averaged cosine distances, 1 or 2 fc stabilizers outperform all the compared methods.

Relational classifier: Here, we verify the auxiliary relational module and the pairwise ranking loss by ablating each. To show their effect, we compare each ablated version to the baseline D-0 without any of them in Table 5. The pairwise ranking loss gives performance improvement by 1.5% and 0.6% for 1- and 5-shot, respectively (D-1). Hence, this loss lets the proposals with larger IoUs to ground-truths get higher action scores. From the result of D-2, we also see that learning the auxiliary module in parallel to the action classifier is useful to boost performance in both settings. And, combining both works the best (D-3).

5 Conclusions

We proposed the dual cross-attention coupled with stabilizer (DCAPS) and the relational classifier for few-shot common action Localization. DCAPS increases the effect of the cross-attention by individually attending each support video, coupling the individually attended ones using the stabilizer, and then attending the query video via all the stabilized support videos. In the relational classifier, we designed the pairwise ranking loss which makes more precise action localization of the action classifier. Learned in parallel with the action classifier, the auxiliary relational module with the pseudo-class labels prevents the network from overfitting to each training episode. This module is discarded in testing. Each component of our method is analyzed and validated through extensive experiments. Finally, we achieved the state-of-the-art on ActivityNet1.3 and THUMOS14.

REFERENCES

Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015.

Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *ECCV*, 2016.

Yang Feng, Lin Ma, Wei Liu, Tong Zhang, and Jiebo Luo. Video re-localization. In *ECCV*, 2018.

Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *ICCV*, 2017.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 2010.

Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Fast temporal activity proposals for efficient detection of human actions in untrimmed videos. In *CVPR*, 2016.

Ruibing Hou, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Cross attention network for few-shot classification. In *NeurIPS*, 2019.

Tao Hu, Pascal Mettes, Jia-Hong Huang, and Cees GM Snoek. Silco: Show a few images, localize the common object. In *ICCV*, 2019.

Haroon Idrees, Amir R Zamir, Yu-Gang Jiang, Alex Gorban, Ivan Laptev, Rahul Sukthankar, and Mubarak Shah. The thumos challenge on action recognition for videos "in the wild". *Computer Vision and Image Understanding*, 155:1–23, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Mihir Jain, Amir Ghodrati, and Cees G. M. Snoek. ActionBytes: Learning from trimmed videos to localize actions. In *CVPR*. 2020.

Dahyun Kang, Heeseung Kwon, Juhong Min, and Minsu Cho. Relational embedding for few-shot classification. 2021.

Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

Hanul Kim, Mihir Jain, Jun-Tae Lee, Sungrack Yun, and Fatih Porikli. Efficient action recognition via dynamic knowledge propagation. In *ICCV*, 2021.

Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Hadamard product for low-rank bilinear pooling. In *ICLR*, 2017.

Jun-Tae Lee, Mihir Jain, Hyoungwoo Park, and Sungrack Yun. Cross-attentional audio-visual fusion for weakly-supervised action localization. In *ICLR*. 2021a.

Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *ECCV*, 2018.

Pilhyeon Lee, Jinglu Wang, Yan Lu, and Hyeran Byun. Weakly-supervised temporal action localization by uncertainty modeling. In *AAAI*. 2021b.

Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*, 2018.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *NeurIPS*, 2016.

Zhekun Luo, Devin Guillory, Baifeng Shi, Wei Ke, Fang Wan, Trevor Darrell, and Huijuan Xu. Weakly-supervised action localization with expectation-maximization multi-instance learning. In *ECCV*. 2020.

Junwei Ma, Satya Krishna Gorti, Maksims Volkovs, and Guangwei Yu. Weakly supervised action selection learning in video. In *CVPR*. 2021.

Sauradip Nag, Xiatian Zhu, and Tao Xiang. Few-shot temporal action localization with query adaptive transformer. In *BMVC*, 2021.

Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network. In *CVPR*. 2018.

Boris N Oreshkin, Pau Rodriguez, and Alexandre Lacoste. Tadam: Task dependent adaptive metric for improved few-shot learning. In *NeurIPS*, 2018.

Sujoy Paul, Sourya Roy, and Amit K. Roy-Chowdhury. W-TALC: Weakly-supervised temporal activity localization and classification. In *ECCV*. 2018.

R Gnana Praveen, Wheidima Carneiro de Melo, Nasib Ullah, Haseeb Aslam, Osama Zeeshan, Théo Denorme, Marco Pedersoli, Alessandro L Koerich, Simon Bacon, Patrick Cardinal, et al. A joint cross-attention model for audio-visual fusion in dimensional emotion recognition. In *CVPR*. 2022.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

Baifeng Shi, Qi Dai, Yadong Mu, and Jingdong Wang. Weakly-supervised action localization by generative attention modeling. In *CVPR*. 2020.

Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *CVPR*, 2016.

Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *CVPR*, 2017.

Krishna Kumar Singh and Yong Jae Lee. Hide-and-seek: Forcing a network to be meticulous for weakly-supervised object and action localization. In *ICCV*. 2017.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. UntrimmedNets for weakly supervised action recognition and detection. In *CVPR*. 2017.

Xi Wei, Tianzhu Zhang, Yan Li, Yongdong Zhang, and Feng Wu. Multi-modality cross attention network for image and sentence matching. In *CVPR*, 2020.

Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *ICCV*, 2017.

Huijuan Xu, Ximeng Sun, Eric Tzeng, Abir Das, Kate Saenko, and Trevor Darrell. Revisiting few-shot activity detection with class similarity control. In *arXiv preprint arXiv:2004.00137*, 2020a.

Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *CVPR*, 2020b.

Hongtao Yang, Xuming He, and Fatih Porikli. One-shot action localization by learning sequence matching network. In *CVPR*, 2018.

Pengwan Yang, Vincent Tao Hu, Pascal Mettes, and Cees GM Snoek. Localizing the common action among a few videos. In *ECCV*, 2020.

Pengwan Yang, Pascal Mettes, and Cees GM Snoek. Few-shot transformation of common actions into time and space. In *CVPR*, 2021.

Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *CVPR*, 2016.

Gang Yu and Junsong Yuan. Fast action proposals for human action detection and search. In *CVPR*, 2015.

Can Zhang, Meng Cao, Dongming Yang, Jie Chen, and Yuexian Zou. Cola: Weakly-supervised temporal action localization with snippet contrastive learning. In *CVPR*, 2021.

Da Zhang, Xiyang Dai, and Yuan-Fang Wang. Metal: Minimum effort temporal activity localization in untrimmed videos. In *CVPR*, 2020.

Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In *ICCV*, 2021.

A Details of the loss functions

To facilitate the follow-up works, we provide more detailed description for the loss functions.

As the support-agnostic part, the proposal subnet yields two intermediate outputs for each query proposal. First, softmax probability vector $(p^{\text{ag}}, r^{\text{ag}})$ where p^{ag} and r^{ag} are the probabilities that a query proposal belongs to the activity and non-activity classes, respectively. Second, the offset vector $(\Delta t_1^{\text{ag}}, \Delta t_2^{\text{ag}})$ is the temporal offset of start and end times to the corresponding ground-truth. Then, with the ground-truth binary vector $(\bar{p}^{\text{ag}}, \bar{r}^{\text{ag}})$ and the ground-truth regression offeset $(\Delta \bar{t}_1^{\text{ag}}, \Delta \bar{t}_2^{\text{ag}})$, we compute two losses:

$$\mathcal{L}_{\text{cls}}^{\text{ag}} = \frac{1}{N^{\text{ag}}} \sum_n -\bar{p}_n^{\text{ag}} \log(p_n^{\text{ag}}) - \bar{r}_n^{\text{ag}} \log(r_n^{\text{ag}}) \tag{7}$$

$$\mathcal{L}_{\text{reg}}^{\text{ag}} = \frac{1}{N^{\text{ag}}} \sum_n \eta(\Delta t_{1,n}^{\text{ag}} - \Delta \bar{t}_{1,n}^{\text{ag}}) + \eta(\Delta t_{2,n}^{\text{ag}} - \Delta \bar{t}_{2,n}^{\text{ag}}) \tag{8}$$

where N_{ag} is the number of the query proposals before NMS, and η is the smooth l_1 function defined by

$$\eta(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1, \\ |x| - 0.5 & \text{otherwise.} \end{cases} \tag{9}$$

Similar to the support-agnostic, the support-conditioned losses are computed for the two outputs of the action classifiers $(p^{\text{co}}, r^{\text{co}})$ and $(\Delta t_1^{\text{co}}, \Delta t_2^{\text{co}})$:

$$\mathcal{L}_{\text{cls}}^{\text{co}} = \frac{1}{N^{\text{co}}} \sum_n -\bar{p}_n^{\text{co}} \log(p_n^{\text{co}}) - \bar{r}_n^{\text{co}} \log(r_n^{\text{co}}) \tag{10}$$

$$\mathcal{L}_{\text{reg}}^{\text{co}} = \frac{1}{N^{\text{co}}} \sum_n \eta(\Delta t_{1,n}^{\text{co}} - \Delta \bar{t}_{1,n}^{\text{co}}) + \eta(\Delta t_{2,n}^{\text{co}} - \Delta \bar{t}_{2,n}^{\text{co}}) \tag{11}$$

where N_{co} is the number of the query proposals after NMS, and $(\bar{p}^{\text{co}}, \bar{r}^{\text{co}})$ and $(\Delta \bar{t}_1^{\text{co}}, \Delta \bar{t}_2^{\text{co}})$ denote ground-truths.

Finally, with the pairwise ranking loss $\mathcal{L}_{\text{rank}}$ of eq. (4) and the binary cross-entropy loss $\mathcal{L}_{\text{aux}}$ for the auxiliary relational module, the total loss $\mathcal{L}$ is represented by

$$\mathcal{L} = \mathcal{L}_{\text{cls}}^{\text{ag}} + \mathcal{L}_{\text{reg}}^{\text{ag}} + \mathcal{L}_{\text{cls}}^{\text{co}} + \mathcal{L}_{\text{reg}}^{\text{co}} + \mathcal{L}_{\text{aux}} + \lambda \mathcal{L}_{\text{rank}} \tag{12}$$

where λ, the loss weight of $\mathcal{L}_{\text{rank}}$, is experimentally set to 0.3 for all experiments.

B Performance at various IoU thresholds

We show more quantitative results of the proposed method and (Yang et al., 2020; Nag et al., 2021) in terms of mAP scores by varying the IoU threshold from 0.5 to 0.9 in Tables 8 and 9 for ActivityNet1.3 and THUMOS14, respectively.

Table 8: The mAPs (%) at different IoU thresholds and Avg mAP across the IoU thresholds on ActivityNet1.3.

			mAP@IoU					
			0.5	0.6	0.7	0.8	0.9	Avg
Single-inst.	1-shot	(Yang et al., 2020)	53.1	40.9	29.8	18.2	8.4	29.5
		(Nag et al., 2021)	55.1	45.2	35.5	25.3	13.2	32.5
		Ours	61.3	55.6	33.8	22.9	8.0	35.8
	5-shot	(Yang et al., 2020)	56.5	47.0	37.4	21.5	11.9	34.9
		(Nag et al., 2021)	63.0	54.5	44.2	30.9	15.8	38.4
		Ours	65.1	55.6	40.7	26.2	13.6	40.3
Multi-inst.	1-shot	(Yang et al., 2020)	42.1	36.0	18.5	11.1	7.0	22.9
		(Nag et al., 2021)	44.1	37.8	29.5	21.4	11.5	25.8
		Ours	47.2	45.3	27.8	9.5	5.0	26.9
	5-shot	(Yang et al., 2020)	43.9	37.4	20.2	13.4	7.7	24.5
		(Nag et al., 2021)	48.2	39.1	29.7	22.5	12.8	28.2
		Ours	52.2	46.8	26.1	17.8	12.2	31.0

Table 9: The mAPs (%) at diverse IoU thresholds and Avg mAP across the IoU thresholds on THUMOS14.

			mAP@IoU					
			0.5	0.6	0.7	0.8	0.9	Avg
Single-inst.	1-shot	(Yang et al., 2020)	48.7	36.7	19.8	8.3	3.7	23.7
		(Nag et al., 2021)	49.2	36.9	24.3	16.5	10.1	27.2
		Ours	53.3	45.6	27.7	8.2	2.8	27.5
	5-shot	(Yang et al., 2020)	51.9	42.7	24.4	17.7	10.1	29.3
		(Nag et al., 2021)	54.3	43.6	35.8	24.5	12.2	31.6
		Ours	57.9	49.9	29.3	16.3	6.2	31.9
Multi-inst.	1-shot	(Yang et al., 2020)	7.1	3.2	2.8	1.9	0.7	3.1
		(Nag et al., 2021)	7.3	4.2	3.1	2.0	1.5	3.7
		Ours	9.2	6.7	4.6	2.5	0.6	4.7
	5-shot	(Yang et al., 2020)	8.6	5.6	3.8	2.5	1.7	4.4
		(Nag et al., 2021)	10.4	7.1	5.7	4.8	2.9	5.4
		Ours	14.0	11.1	8.6	5.7	2.3	8.3

C More Analysis

C.1 Adequacy to the number of pseudo action labels

In the auxiliary relational module, we generate the multi-hot indicator based on pseudo action labels and concatenate it to the features of a query proposal and a support prototype. To see the multi-hot indicator effectiveness, in Table 10, we report the mAP scores by varying the number of pseudo action classes (k) and w/o the indicator as well. In 1-shot setting, the auxiliary relational module without the multi-hot indicator ('w/o indicator') works decently, compared to D-1 of Table 5 (60.2% and 64.1% for 1- and 5-shot, respectively). When k is 80, because of too few pseudo-classes, performance is slightly lower than the 'w/o indicator' for the single-instance case. This performance drop is larger in the multi-instance case. However, for all other settings, as k gets larger, the multi-hot indicator yields larger performance gains overall. Though the support videos represent a common action, there is diversity from background or action details. Hence, it is beneficial to distinguish the support videos with more pseudo labels. This is more crucial to the multi-instance case. Considering computation cost, we set k to 160 in other experiments. Similarly, we set it to 16 for the THUMOS14 dataset, which makes sense as this dataset has 10 times fewer action classes than the ActivityNet 1.3 dataset.

Table 10: Effect of the multi-hot pseudo action indicator by varying the number of pseudo action classes or removing it.

		w/o Indicator	k			
			80	160	240	320
Single Inst.	1-shot	60.2	59.2	61.3	62.1	61.5
	5-shot	64.7	63.8	65.1	65.4	65.7
Multiple Inst.	1-shot	45.1	42.5	47.2	47.0	47.4
	5-shot	49.2	47.1	52.2	52.8	53.2

Table 11: Robustness to the number of support videos in testing for the single-instance case.

		No. testing supports				
	Shots	5	4	3	2	1
(Yang et al., 2020)	1	48.1	48.5	48.9	53.3	53.1
	5	56.5	54.8	51.7	47.9	48.1
Ours	1	57.7	57.8	58.1	57.8	61.3
	5	65.1	64.7	64.8	63.1	60.5

C.2 Robustness to number of testing support videos

Now we evaluate the robustness to the number of testing support videos. Specifically, in Table 11, we verify the 1- or 5-shot trained models by varying the numbers of support videos from 1 to 5. Both 1-shot models of our method and (Yang et al., 2020) are fit to one support video, and hence their performance is degraded for more than one testing support video. Note that, here our 1-shot model does not include the stabilizer, so the dual cross-attention is applied without the stabilizer in this setting. Nevertheless, our method shows less performance drop compared to (Yang et al., 2020). The drop for five testing support videos is 3.6% in ours and is 5.0% in (Yang et al., 2020). For the 5-shot setting, as the numbers of the testing support videos are reduced, the performance largely drops in (Yang et al., 2020). Contrarily, our 5-shot model robustly performs even for two testing support videos. However, as the 1-shot model is better tuned for one support video, it shows a better result than the 5-shot model for testing of one support video. Hence, we deduce that our DCAPS and relational classifier are also robust to diverse numbers of testing support videos.

C.3 Complete our method *vs* variants w/o stabilizer for pairwise cosine distance in support videos

To analyze the effect of the stabilizer further, we compare our complete method and two variants without the stabilizer: i) 'seperate cross-att. w/o stab.': different attention weights for query-to-support and support-to-query attentions. ii) 'shared cross-att. w/o stab.': reciprocally shared attention weight for query-to-support and support-to-query attentions. In both, the support videos are individually attended. Table 12 compares our method with the two variants in terms of the average of the pairwise cosine distance where the each support segment pair is sampled from different support videos as in Table 1. Compared with ours (0.005 after stabilizer), 0.024 and 0.038 is further large (in degree, 5.73 *vs* 12.57 and 15.84).

Also, even the support segment features before the stabilizer of our complete method shows the lower pairwise cosine distances (0.018) than the two variants. Since the model is end-to-end learned, adding the stabilizer affects the distribution of the support segment features before passing the stabilizer as well. After passing through the stabilizer, the support segment features are further densely located in latent space. Therefore, the MLP stabilizer helps our individual cross-attention (stage 1) be learned stably, and it results in the attended support segment features densely distributed in latent space.

Table 12: Pairwise cosine dist. between two support segment features from different support videos ($L = 5$).

Method	Ours		i) Seperate cross-att. w/o stab.	ii) Shared cross-att. w/o stab.
Features	$\{\tilde{X}_S^{(l)}\}$ (before stabilizer)	$\{\tilde{Y}_S^{(l)}\}$ (after stabilizer)	$\{\tilde{X}_S^{(l)}\}$	$\{\tilde{X}_S^{(l)}\}$
Avg cosine dist.	0.018	0.005	0.024	0.038
mAP (%)	65.1		60.9	56.7

Table 13: Pairwise l_2 dist. between two support segment features from different support videos ($L = 5$).

Features	$\{\tilde{X}_S^{(l)}\}$ (before stabilizer)	$\{\tilde{Y}_S^{(l)}\}$ (after stabilizer)
Avg l_2 dist.	15.12	6.91

Table 14: Pairwise l_2 dist. between positive and negative query proposal features in the same video.

Features	w/o stabilizer	Ours
Avg l_2 dist.	24.21	29.34

C.4 Pairwise l_2 distance analysis on the stabilizer

To more convince the effect of stabilizer, we analyze the stabilizer in different distance metric, l_2. In specific, we first perform pairwise l_2 distance analysis for support video and query proposal features, respectively. 1) we provide the l_2 distance version of Tables 1 and 2. Similar to Table 1, we show the average pairwise (two paired support segments are in different support videos) distances before or after passing through the stabilizer in Table 13. 2) Also, akin to Table 2, Table 14 shows the pairwise average l_2 distances where pairs are positive and negative query proposals. In both Tables, we can see the similar tendency to the cosine distance analysis.

Then, we compare the proposed method and 'w/o stabilizer individual' according to the l_2 distance of the attended support video (not segment-level) features to the attended positive query proposal. An attended support video feature is obtained by averaging the attended support segment features in the support video. As demonstrated in Table 15, compared to the 'w/o stabilizer individual,' the average pairwise l_2 distance is smaller to positive queries and larger to negative queries in our complete method. Therefore, the stabilizer induces to obtain more discriminative attended query proposal features which helps to learn the following relation classifier more robustly.

C.5 Ablation study on relational classifier

We provide the ablation study of Table 5 for a more simple baseline where the proposed DCAPS is replaced with 'AggAtt w/o stb.' In AggAtt w/o stb, the aggregated cross-attention is applied on support videos w/o the stabilizer, and then query proposals are enhanced by the attended support videos. Hence, AggAtt w/o stb is a naive two-stage cross-attention without our key contributions (individual cross-attention in support videos and stabilizer) for query-support alignment. The Table 16 shows the result. Like Table 5, we can see that the proposed $\mathcal{L}_{\text{rank}}$ and auxiliary relational module ($\mathcal{L}_{\text{aux}}$) are effective themselves to common action localization.

D Experimental details

We implemented our method based on the official PyTorch source code of (Yang et al., 2020). Also, following the experimental setting of (Yang et al., 2020), train our network using Adam optimizer with an initial learning rate of 1e-5, decayed by 0.1 every 25k iterations. The batch size is set as 1. For a fair comparison, we use the same backbone network and proposal subnet with those of (Yang et al., 2020). In specific, we employ the C3D network (Tran et al., 2015) as the backbone which is pre-trained on Sport-1M (Karpathy et al., 2014) dataset. For query proposal generation, we adopt

Table 15: Pairwise l_2 dist. from video-level support features to positive or negative query proposal features.

Features	w/o stabilizer		Ours	
Query proposals	Positive	Negative	Positive	Negative
Avg l_2 dist.	16.67	26.20	10.14	31.86

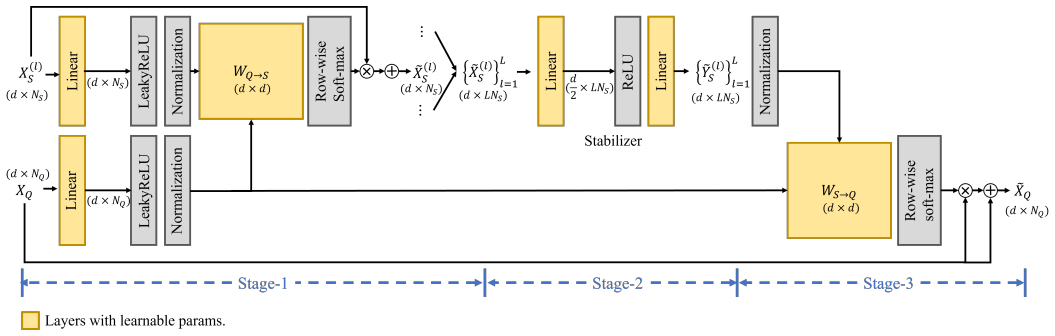
Table 16: Analysis on the relational classifier.

Method	$\mathcal{L}_{\text{rank}}$	Auxiliary rel. module	mAP (%)	
			1-shot	5-shot
AggAtt w/o stb-0			52.7	55.1
AggAtt w/o stb-1	✓		53.6	56.4
AggAtt w/o stb-2		✓	54.0	56.9
AggAtt w/o stb-3	✓	✓	54.3	57.2

the proposal subnet of R-C3D (Xu et al., 2017) to obtain class-agnostic action proposals. Also, for the DCAPS and relational classifier, we used the Xavier (Glorot & Bengio, 2010) initializer. As in (Yang et al., 2020), the proposal score threshold is set as 0.7, and the number of proposals after NMS is 128 in training and 300 in validation and testing. On top of the backbone, we construct the DCAPS and relational classifier.

In the auxiliary relational module, the k-means clustering for the pseudo action label generation is implemented by `scikit-learn` library of version 1.0.1 in the default hyperparameter setting. In consideration of the randomness of the k-means clustering, we run three times for the reported scores for our method and its variants in all the Tables.

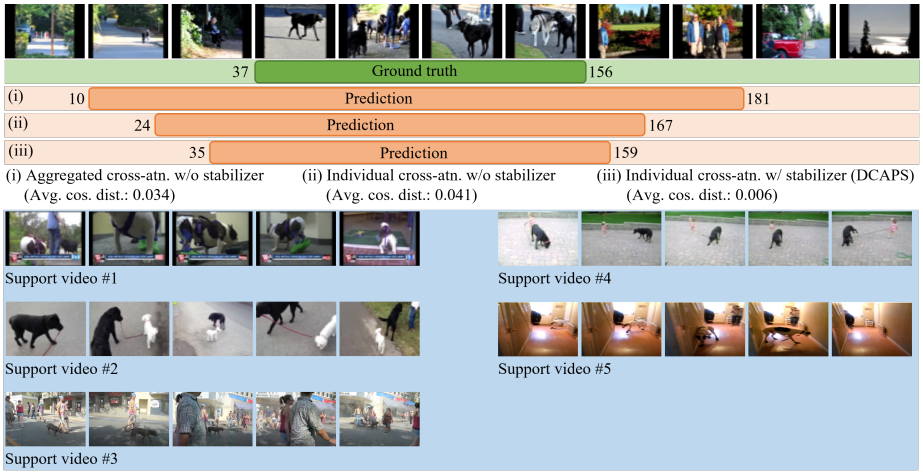
E Detailed structure of DCAPS



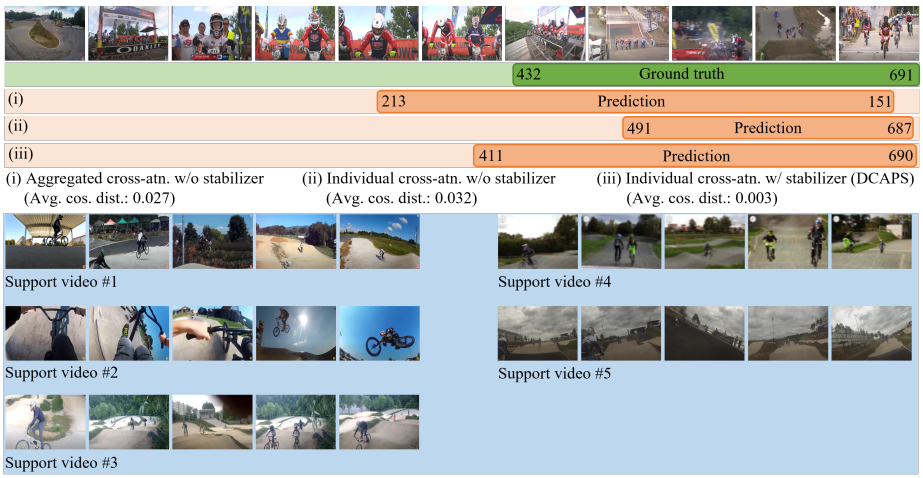
The figure above describes the detailed structure of 3-stage attention mechanism of the proposed DCAPS. First, the backbone features of the query proposal and support segments are embedded by different linear layers, and the query proposal features individually attend the segment features of each support video at the first stage. Then, when the number of support videos are larger than 1, the stabilizer of the second stage increases the compatibility of the individually attended features of the support videos. Lastly, at the third stage, all the attended support video features simultaneously attend the query proposal features via the second cross-attention matrix. The final attended query proposal features are fed into the following relational classifier.

F Qualitative results

We visualize predicted action instances in Fig. 6. (ii) shows better results than (i). Hence, the individual cross-attention is effective to attend multiple support videos. And, the stabilizer increases the compatibility of different support videos, yielding the most accurate results as exemplified in (iii). We also provide the average of the cosine distances between the support segment features (after the support attention in (i) and (ii), after the stabilizer in (iii)). In both examples, (iii) shows the smallest average cosine distance.



(a) Walking the dog



(b) BMX

Figure 6: Qualitative results. (i) aggregated cross-attention w/o stabilizer, (ii) individual cross-attention w/o stabilizer and (iii) individual cross-attention w/ stabilizer (DCAPS, our final attention module). Each of the ground-truths and predicted action instances is colorized with the frame indices of start and end times. Average of the cosine distances between the support segment features (after the support attention in (i) and (ii), after the stabilizer in (iii)) is provided, as well.