Long-term Pre-traning for Temporal Action Detection with Transformers

Jihwan Kim   **Miso Lee**   **Jae-Pil Heo**[†]

Sungkyunkwan University

{damien, dlalth557, jaepilheo}@skku.edu

Abstract

Temporal action detection (TAD) is challenging, yet fundamental for real-world video applications. Recently, DETR-based models for TAD have been prevailing thanks to their unique benefits. However, transformers demand a huge dataset, and unfortunately data scarcity in TAD causes a severe degeneration. In this paper, we identify two crucial problems from data scarcity: attention collapse and imbalanced performance. To this end, we propose a new pre-training strategy, Long-Term Pre-training (LTP), tailored for transformers. LTP has two main components: 1) class-wise synthesis, 2) long-term pretext tasks. Firstly, we synthesize long-form video features by merging video snippets of a target class and non-target classes. They are analogous to untrimmed data used in TAD, despite being created from trimmed data. In addition, we devise two types of long-term pretext tasks to learn long-term dependency. They impose long-term conditions such as finding second-to-fourth or short-duration actions. Our extensive experiments show state-of-the-art performances in DETR-based methods on ActivityNet-v1.3 and THUMOS14 by a large margin. Moreover, we demonstrate that LTP significantly relieves the data scarcity issues in TAD.

Introduction

Understanding videos has become fundamental for real-world video applications with the proliferation of videos being shared across online platforms. Initially, researchers concentrated on classifying actions in videos by using trimmed clips. However, this became expensive, leading to the development of Temporal Action Detection (TAD). TAD not only classifies actions correctly but also identifies the time boundaries in untrimmed videos.

End-to-end object DEtection with TRansformers (DETR) (Carion et al. 2020) has been prevailing in the literature of detection. DETR views detection as a set prediction for end-to-end detection without any human prior like non-maximum suppression or pre-defined anchors.

Introduction of DETR to TAD is natural for next research steps as DETR inherits strengths of transformer (Vaswani et al. 2017), bringing two valuable characteristics for TAD. First, it does not depend on pre-defined anchors. This merit is crucial for TAD since actions in videos have more diverse time lengths than object sizes. Second, it can directly learn
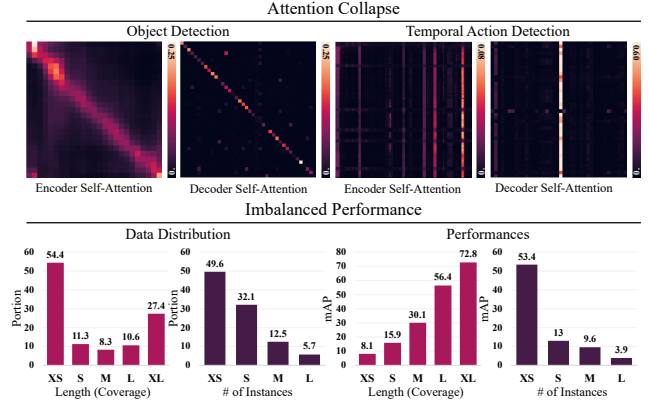

[†]Corresponding author

Figure 1: **Problems from data scarcity.** There are two main problems caused by the data scarcity in DETR for TAD: attention collapse and imbalanced performance. The first row shows the collapsed self-attention maps from encoder and decoder of DAB-DETR. The second depicts imbalanced performances in terms of action lengths (in Coverage) and the number of instances on ActivityNet-v1.3 from DETAD.

global relation without stacking multiple layers. It can relieve the problem from long-term dependency for TAD.

Despite the great benefits, transformers demand a large-scale dataset with sufficient inter- and intra-class diversities to learn dense relations without an inductive bias. Especially for DETR, the bipartite matching further strengthens the demand because of the label inefficiency from one-to-one matching. However, the current TAD benchmarks do not meet such a demand due to expensive labeling cost.

Consequently, directly employing DETR to TAD causes a severe degeneration for transformers. We empirically identify two primary problems from data scarcity: 1) attention collapse, 2) imbalanced performance, as depicted in Fig. 1. Firstly, the collapse of self-attention in transformer-based methods is the phenomenon to skip the self-attention layers since the model cannot learn expressive relations between video features and action queries from data (Dong, Cordonnier, and Loukas 2021). As in the upper of Fig. 1 unlike DETR for object detection, all queries of DETR for TAD attend a few key elements in which the gradients for them are vanishing (Noci et al. 2022), indicating that the self-attention quits learning. The collapse problem can be

arXiv:2408.13152v1 [cs.CV] 23 Aug 2024

caused by the two main aspects: model architecture (Dong, Cordonnier, and Loukas 2021; Zhai et al. 2023) and data diversity (Tang et al. 2021; Trockman and Kolter 2023). In this paper, we focus on the data perspective to cope with the collapse problem. Secondly, the model exhibits severe imbalanced performances in terms of action lengths and the number of action instances within a video. DETR does not use multi-scale division for label assignment, in which all queries are responsible for all temporal scales of actions. When this design faces insufficient data, it can be easier to fall in over-fitting to a major scale (usually to long actions).

To settle down the issue, we simply suggest that DETR be pre-trained with large-scale balanced data. Here, our focus is on 'the detector (i.e., DETR)', not on the feature extractor. Although there are many attempts to pre-train the feature extractor, there is no consideration for the detector. The pre-training for the detector should be oriented towards the temporal detection as it is designed for localization.

To this end, we develop a new pre-training strategy tailored for transformers of DETR, named Long-Term Pre-training (LTP). LTP has two main components: 1) class-wise synthesis, 2) long-term pretext tasks. Firstly, we synthesize long-form video features for pre-training from a large-scale video-classification dataset with trimmed clips to provide the model with sufficient and balanced data diversity. We build pre-training features based on action categories to minimize task discrepancy between the pretext task in pre-training and the downstream task (TAD) in fine-tuning. Although our way of synthesizing is closely related to previous pre-training methods, the key difference is in the class-wise pretext task, for which the model should detect all actions of the target class, instead of identifying a single action instance. Secondly, we further introduce two long-term pretext tasks, ordinal and scale conditions, to teach the model long-term dependency. For example, the model is required to identify exclusively "first-to-second" actions (ordinal) or "short-duration" actions (scale) among instances of the target category in synthesized videos during pre-training. These tasks impose specialized conditions on the basic pretext task, to further encourage DETR to learn long-term dependency and diverse relations in attention.

Our extensive experiments demonstrate that our pre-training remarkably alleviates the data scarcity problem. As a result, DETR models with LTP outperform the state-of-the-art DETR-based methods on two TAD benchmarks, ActivityNet-v1.3 and THUMOS14, by a large margin.

To sum up, our main contributions are as follows:

- To our knowledge, this is the first attempt to highlight the importance of pre-training DETR for TAD in mitigating the issues arising from limited data.

- We propose a novel pre-training strategy, Long-Term Pre-training (LTP) for TAD with transformers. Class-wise synthesis and long-term pretext tasks provide diverse and balanced data and capability to learn long-term dependency while minimizing task discrepancy.

- Our extensive experiments demonstrate that LTP significantly relieves the data scarcity issues with a new SoTA performance on ActivityNet-v1.3 and THUMOS14.

Related Work

Temporal Action Detection

Temporal action detection (TAD) aims to find time intervals of actions of interest in untrimmed videos as well as classifying the instance. Pioneering methods (Yeung et al. 2016; Shou, Wang, and Chang 2016; Buch et al. 2017) have been made great advance in TAD during the past decade. As two-stage mechanism had been successful in object detection, numerous methods in TAD deployed multi-stage framework (Gao et al. 2017; Zhao et al. 2017; Xu, Das, and Saenko 2017; Kim and Heo 2019).

As the following methods, point-wise learning has been widely adopted to generate more flexible proposals without pre-defined time windows. SSN (Zhao et al. 2017) and TCN (Dai et al. 2017) expanded temporal context around a generated proposal to improve ranking performance. BSN (Lin et al. 2018) and BMN (Lin et al. 2019) grouped possible start-end pairs to build action proposals, then scored them for final localization predictions. BSN++ (Su et al. 2021) tackled the imbalance problem over temporal scales based on BSN. Recently, ActionFormer (Zhang, Wu, and Li 2022) deployed transformer-based encoder as multi-scale backbone network, and BRN (Kim et al. 2024) resolved the issue of multi-scale features for TAD.

Pre-training for Detection

End-to-end object DEtection with TRansformers (DETR) (Carion et al. 2020) firstly viewed object detection as a direct set prediction problem, and removed the need of human heuristics like non-maximum-suppression (NMS). However, transformers of DETR typically need 10 times more training iterations than the conventional detectors due to bipartite matching and dense attention.

To address the issue, there have been research efforts (Dai et al. 2021; Bar et al. 2022) for pre-training DETR. UP-DETR (Dai et al. 2021) is the first attempt for pre-training DETR to overcome data scarcity. UP-DETR introduced an instance-wise pre-training strategy, which finds the instance specified by the pooled features of a random region in the image. Also, DETReg (Bar et al. 2022) pointed out that the randomness of UP-DETR reduces the benefits of pre-training and to resolve this problem they devised object-centric selection in an unsupervised manner.

In the literature of TAD, there have been also pre-training approaches (Alwassel, Giancola, and Ghanem 2021; Xu et al. 2021a,b; Zhang et al. 2022; Kang et al. 2023) but all for the feature extractor. TSP (Alwassel, Giancola, and Ghanem 2021) and BSP (Xu et al. 2021a) have proposed boundary-sensitive pre-training strategies for the feature extractor. Also, LoFi (Xu et al. 2021b) devised fine-tuning strategies for the feature extractor with fewer clips and small images. On the other hand, PAL (Zhang et al. 2022) and SoLa (Kang et al. 2023) have introduced the methods for the task-specific temporal encoder for the feature extractor.

Recently, DETR has been introduced in TAD (Tan et al. 2021; Liu et al. 2021; Shi et al. 2022; Kim, Lee, and Heo 2023) as it prevails in object detection. However, there are no research attempts of pre-training DETR for TAD while
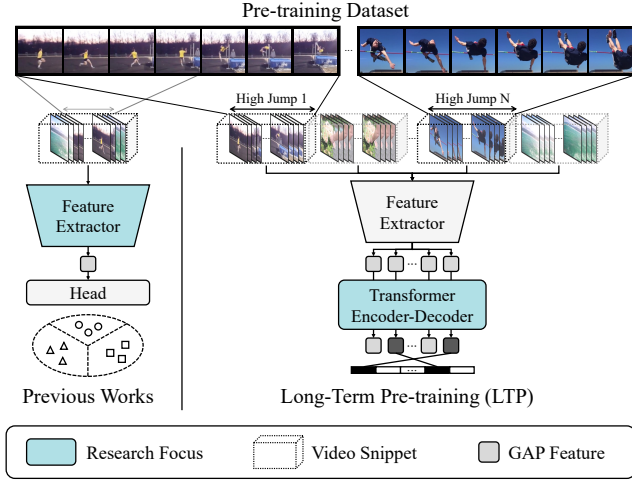
Figure 2: **Differences b/w previous and our pre-training.** Previous pre-training focused on the feature extractor. However, there has been no research conducted on pre-training DETR for TAD despite the issues on the data scarcity. The pretext tasks to train feature extractor and detector should be different. LTP is designed for class-wise localization from long-form videos, just like the downstream task.

the performance of DETR for TAD has been far behind conventional TAD methods due to the severe data scarcity. As for the feature extractor, the goal is to teach frame-wisely fine-grained temporal sensitivity with short videos (~2.6 seconds) as illustrated in Fig. 2. On the other hand, the pre-training for the detector should be oriented towards the temporal detection as it is designed for localization. Unlike previous pre-training methods of TAD, our pre-training method is specialized for pre-training of DETR for TAD.

Our Approach

In this section, we elaborate the details of our pre-training framework, Long-Term Pre-training (LTP). As illustrated in Fig. 3, LTP has two main components: 1) class-wise synthesis, 2) long-term pretext tasks. In class-wise synthesis, we construct features to form a localization problem by integrating target and non-target actions. Additionally, we introduce two kinds of conditions: ordinal and scale tasks.

Preliminary

DETR. DETR (Carion et al. 2020) is an architecture designed for end-to-end detection, comprising feature extractor, transformer encoder, and transformer decoder. The feature extractor captures general features from images, while the transformer encoder and decoder focus on encoding more specific features and decoding object information from features. The core of the transformer is the attention mechanism, which is defined formally as follows:

$$\text{Attention}(Q, K, V) = AV,$$
$$A = \text{softmax}(\frac{QK^{\top}}{\sqrt{d_k}}), \qquad (1)$$

where Q, K, V, d_k are query, key, value and dimension of the key, respectively. When we feed the same input into query, key, and value, it is commonly referred as self-attention mechanism. The transformer encoder in DETR consists of multiple layers of self-attention and feed-forward networks, refining the CNN features for the task.

On the other hand, the cross-attention mechanism employs the same key and value but with a different query. This enables the query to be represented by the input given as key and value. The transformer decoder in DETR includes several layers of self-attention, cross-attention, and feed-forward networks. Learnable embedding vectors are fed into self-attention networks as input and cross-attention networks as query, so called as decoder queries. For the cross-attention networks, encoder outputs are used as key and value. This allows the decoder to extract unique localization information from the encoder outputs for each query. Finally, DETR predicts the classes and the locations using the decoder outputs and bipartite matching.

DETR for TAD. In TAD, we deploy a 3D CNN as the feature extractor, which is pre-trained on Kinetics (Kay et al. 2017). Note that the 3D CNN is fixed while training the transformer encoder and decoder of DETR. To extract the video features, each video is fed into the 3D CNN followed by global-average pooling for spatial dimensions so that only the temporal dimension remains. When it comes to architectures of DETR, we deploy two representative architectures, DAB-DETR (Liu et al. 2022) and deformable-DETR (Zhu et al. 2021) in this paper. Decoder queries stand for action instances, called action queries. Therefore, the decoder receives the refined video features from the encoder and relates them with action queries. Finally, the output of the decoder passes through classification and regression heads, then final detection results are produced.

Class-wise Synthesis

Motivation. In order to provide sufficient and balanced data especially for relieving imbalanced performances over scales, we utilize large-scale action recognition dataset like Kinetics (Kay et al. 2017). Also, long-form video features are vital to build a localization pretext task for transformer in DETR. Here, we aim to minimize the task discrepancy between pre-training and fine-tuning (TAD) while making large-scale balanced data via class-wise synthesis.

Synthesis. As with earlier pre-training approaches (Alwassel, Giancola, and Ghanem 2021; Xu et al. 2021a), we must create video features suitable for localization since action recognition videos lack background regions.

To achieve this, we bring in class-wise synthesis to reduce task differences. We start by picking a single target action category from the Kinetics classes. Then we randomly select videos from other classes to make a background template, which holds non-target actions. Once the template is set, we randomly choose videos from the target class. By placing these chosen target category videos into the background template, we create synthesized video features for the pretext task.
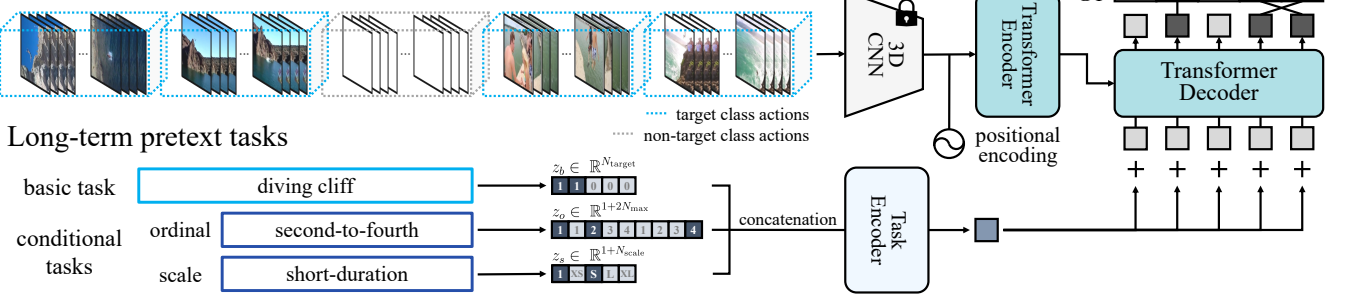
Formally, let us denote the set of categories in Kinetics

Figure 3: **Overall procedure of Long-Term Pre-trainig (LTP).** LTP has two main components: 1) class-wise synthesis, 2) long-term pretext tasks. Class-wise synthesis aims to minimize the task discrepancy by building training features to localize based on categories. Moreover, the conditional tasks aim to learn the long-term dependency by ordinal or scale conditions.

as $K = \{1, ..., N_K\}$. Each video in Kinetics only contains a single action category among K. Also, we have a pre-trained feature extractor $F(\cdot)$, which fixed across both pre-training and fine-tuning phases. Initially, a random selection is made from K to determine the target category k^*. In order to construct a background template B with only non-target actions, we randomly sample N_{bg} videos of other classes except for the target category as below:

$$B = [F(v_1), F(v_2), ..., F(v_{N_{bg}})], \quad k_{v_i} \neq k^*, \quad (2)$$

where $F(v_i)$ is temporal features of the video v_i after global-average pooling, $[\cdot]$ means concatenation along time axis, and k_{v_i} is the category of the video v_i.

After forming the template, we also randomly sample N_t videos of the target class as below:

$$T = \{F(v_1), F(v_2), ..., F(v_{N_t})\}, \quad k_{v_i} = k^*, \quad (3)$$

where $\{\cdot\}$ indicates a set of features.

With the sampled target features T, we firstly temporally crop each video feature into a random size. Then we insert the cropped features of the sampled target videos into the background template B at random temporal positions. We utilize all N_t target features and allow overlapping.

Long-term Pretext Tasks

Motivation. Alongside the basic task to localize actions of the target class, we introduce two types of long-term pretext tasks: ordinal task, and scale task. These tasks are designed to reduce the degree of the attention collapse problem. The tasks encourage the model to learn long-term dependency, and prevent it from attending to a few of temporal features.

Basic Task. To appropriately guide the model in our pre-training, we need to define the desired objective by specifying the target class. This target category is represented through one-hot encoding, denoted as the one-hot vector z_b for indicating the target class for the basic pretext task. The dimension of z_b is in $\mathbb{R}^{N_{\text{target}}}$, where N_{target} corresponds to the total number of action classes in K.

Conditional Tasks. In addition to the basic task, we introduce two types of pretext tasks: ordinal and scale pretext tasks. Firstly, the ordinal task involves identifying actions falling within specified ordinal number ranges. For instance, the model can be trained to identify the third-to-fifth actions of the target class. The ordinal task prompts the model to understand the relative positioning of actions within the video sequence. This requires the model to consider not only individual actions but also their sequence, encouraging the learning that extend beyond neighboring actions.

Specifically, the start ordinal number and range length are randomly selected based on the current number of target actions within the synthesized features. Note that ordinals are assigned exclusively to target class actions. Through the ordinal task, the DETR decoder's action queries are prompted to consider other queries for locating the ordinal position. This fosters the model to learn the global relationships within the video.

Secondly, the scale pretext task focuses on localizing actions corresponding to defined temporal scales. Understanding scale involves how relatively long an action lasts in the video sequence. This comprehension prompts the model to understand the speed of actions compared to the neighboring actions and the global context.

Concretely, we define four scales: extra-short (XS), short (S), long (L), and extra-long (XL). These scales are determined by the action's duration-to-total-video-length ratio (r). XS actions are characterized by $r < 0.25$, S by $0.25 \leq r < 0.50$, L by $0.50 \leq r < 0.75$, and XL by $0.75 \leq r$. To designate the scale, we randomly select one target action from the synthesized features and assign the task with the corresponding scale.

Similar to the basic task, we introduce conditions through one-hot encoding. Formally, let us denote the one hot vectors for the ordinal and scale tasks as z_o, and z_s, respectively. In the scale pretext task, the one-hot vector z_s has a length of 5. The initial channel of z_s indicates whether to apply the scale condition or not, while the remaining four channels specify the target scale among the four available scales. For instance, $[1, 0, 1, 0, 0]$ describes the intent to identify short-duration actions within the target class. Here, we only assign a single scale for the condition.

Moreover, when the maximum number of target action in-

stances within a video in pre-training is denoted as $N_{\max}$, the one-hot vector z_o encompasses $2N_{\max} + 1$ channels. Likewise, the first channel serves as an indicator for conditioning. The subsequent $N_{\max}$ channels following the indicator mean the first ordinal number for the range, while the final $N_{\max}$ channels designate the second ordinal number. For example, with $N_{\max}$ set to 4, the one-hot vector for the ordinal task, seeking second-to-fourth actions within the target class, would be $[1, 0, 1, 0, 0, 0, 0, 0, 1]$.

To establish conditional tasks alongside the basic pretext, we concatenate all one-hot vectors for both basic and conditional pretext tasks along the channel axis. Whether to condition or not is assigned with a probability p_{cond} in uniform distribution. We also employ a task encoder $E(\cdot)$ consisting of three fully-connected layers with ReLU activations. This task encoder $E(\cdot)$ aligns the dimensionality of the one-hot vector to match the action queries of DETR. These projected vectors are then added to DETR decoder's action queries. Formally, let the decoder's action queries be q. The conditioned queries, denoted as q', are defined as follows:

$$q' = q \oplus E([z_b, z_o, z_s]), \qquad (4)$$

where $\oplus$ is element-wise summation, and $[\cdot]$ is concatenation along the channel axis.

Objectives

DETR. We follow the objectives in DETR (Carion et al. 2020). Let us denote the ground-truths, and the M predictions as $y, \hat{y} = \hat{y}_i{}_{i=1}^M$, respectively. For the bipartite matching between the ground-truth and prediction sets, we define the optimal matching $\hat{j}$ with the minimal cost and search from all possible permutation of $\{1, ..., M\}$ denoted by J_M as follows:

$$\hat{j} = \arg\min_{j \in J_M} \sum_i^M \mathcal{L}_{\text{match}}(y_i, \hat{y}_{j(i)}), \qquad (5)$$

where $L_{\text{match}(y_i, \hat{y}_{j(i)})}$ is a pair-wise matching cost between y_i and the prediction with the index from $j(i)$, which outputs the index i from the permutation j.

Next, let us denote each ground-truth action as $y_i = (c_i, t_i)$, where c_i is the target class label with the background one $\emptyset$, and t_i is the time intervals. For the prediction with the index $j_{(i)}$, we define the probability of the class c_i as $\hat{p}_{j(i)}(c_i)$ and the predicted time intervals as $\hat{t}_{j(i)}$. Then we formulate the main objective as following:

$$\mathcal{L}_{\text{DETR}}(y, \hat{y}) = \sum_{i=1}^M [-\log \hat{p}_{\hat{j}(i)}(c_i) + \mathbb{1}_{c_i \neq \emptyset} \mathcal{L}_{\text{reg}}(t_i, \hat{t}_{\hat{j}(i)})], \quad (6)$$

where $\hat{j}$ is the optimal assignment from Eq. 5, and $\mathcal{L}_{\text{reg}}(t_i, \hat{t}_{j(i)})$ is the regression loss between the ground-truth t_i and the prediction $\hat{t}$ with the index $j(i)$.

Pre-training. In LTP, class-wise synthesis assigns the target category to the model, so that the DETR model to perform binary classification. The class-agnostic approach enhances model generality as in class-agnostic object detection (Maaz et al. 2022). To achieve this, we adopt the identical bipartite matching objective as DETR, as depicted in

Eq. 6, with binary classification labels. This configuration enables the model to address the localization task in a class-agnostic manner. Apart from classification, the remaining components of the objective remain consistent with $\mathcal{L}_{\text{DETR}}$.

In our conditional tasks, we delineate actions to locate based on ordinal and scale criteria. These conditions serve to narrow down the scope, aiming to focus on specific target actions based on the criteria. For instance, specifying a search for first-to-fourth actions solely involves the target actions, excluding non-target ones from the ordinal count. Similarly, in cases where the scale condition pertains to identifying short-duration actions, the model must pinpoint short actions within the target class, not short instances from other classes. Consequently, the basic task consistently serves as the objective, while conditional tasks are randomly enabled or disabled with a probability of p_{cond}.

Experiments

Implementation Details

In this section, we briefly describe the implementation details. We recommend to refer to the thorough elaboration for the details provided in the supplementary materials.
Datasets. Our experiments are conducted on three human action benchmarks: Kinetics-400 (Kay et al. 2017), THUMOS14 (Jiang et al. 2014) and ActivityNet-v1.3 (Fabian Caba Heilbron and Niebles 2015).

Feature Extractor. We use the features of I3D (Carreira and Zisserman 2017) or TSN (Wang et al. 2016) pre-trained on Kinetics. The way of extracting follows the backbone model.

DETR Architectures. In this paper, we introduce two types of DETR baselines: 1) DAB-DETR (Liu et al. 2022), 2) Deformable-DETR (Zhu et al. 2021) in a temporal version. These baselines do not have any additional module so that we can solely show the benefits of LTP. The configure of DAB-DETR follows Self-DETR while that of Deformable-DETR is aligned with TadTR. all other configurations follow the DETR baselines (Liu et al. 2021; Shi et al. 2022; Kim, Lee, and Heo 2023) for fair comparison.

Pre-training Set-up. For pre-training, we utilize the Kinetics-400 dataset (Kay et al. 2017), the same dataset on which the feature extractor is pre-trained, which contains approximately 300K videos. We set the condition probability p_{cond} for long-term pretext tasks as 0.5.. We train the DETR model with a mini-batch size of 256 by using AdamW with initial learning rate of 1.0×10^{-4} during 15 epochs.

Fine-tuning Set-up. As down-stream tasks, the pre-trained models are fine-tuned on ActivityNet and THUMOS14. The configuration of fine-tuning follows the baselines.

Main Results

Comparison with the State-of-the-Art. We compare LTP applied on DAB-DETR (Liu et al. 2022), Deformable-DETR (Zhu et al. 2021) and Self-DETR (Kim, Lee, and Heo 2023) with the state-of-the art methods. Table. 1 demonstrates that our LTP significantly improves the performances of various DETR models on both benchmarks.

Method	THUMOS14						ActivityNet-v1.3			
	0.3	0.4	0.5	0.6	0.7	Avg.	0.5	0.75	0.95	Avg.
DETR-based Methods										
RTD-Net (Tan et al. 2021)	68.3	62.3	51.9	38.8	23.7	49.0	47.21	30.68	8.61	30.83
ReAct (Shi et al. 2022)	69.2	65.0	57.1	47.8	35.6	55.0	49.60	33.00	8.60	32.60
DAB-DETR (Liu et al. 2022)	70.5	64.3	53.9	39.3	23.8	50.3	50.62	32.59	7.43	32.40
DAB-DETR + LTP (Ours)	75.2	69.5	60.4	48.4	32.8	57.3	52.71	35.60	9.08	34.92
Self-DETR (Kim, Lee, and Heo 2023)	74.6	69.5	60.0	47.6	31.8	56.7	52.25	33.67	8.40	33.76
Self-DETR + LTP (Ours)	75.3	69.4	60.2	48.5	32.9	57.3	53.05	35.69	8.93	35.09
Deformable-DETR (Zhu et al. 2021)	71.5	66.0	55.5	43.5	29.6	53.2	50.05	33.44	9.81	33.27
Deformable-DETR + LTP (Ours)	73.8	69.4	60.6	48.0	33.7	57.1	52.86	36.27	**10.88**	35.81
TadTR[†] (Liu et al. 2021)	74.8	69.1	60.1	46.6	32.8	56.7	49.91	33.43	8.77	33.23
TadTR[†] + LTP (Ours)	75.5	69.7	60.5	47.3	33.4	57.3	52.61	36.15	9.73	35.46
Encoder-only Methods										
ActionFormer (Zhang, Wu, and Li 2022)	82.1	77.8	71.0	59.4	43.9	66.8	53.50	36.20	8.20	35.60
ActionFormer + LTP (Ours)	82.3	78.3	72.1	59.8	44.2	67.3	54.72	37.88	9.01	36.72
TriDet[†] (Shi et al. 2023)	83.6	80.1	72.9	62.4	47.4	69.3	54.06	36.45	7.90	35.56
TriDet[†] + LTP (Ours)	**84.2**	**81.0**	**73.5**	**63.0**	**48.8**	**70.1**	**55.22**	**37.98**	8.79	**37.03**

Table 1: **Comparison results with the transformer-based models on THUMOS14 and ActivityNet-v1.3.** '†' indicates our reproduced versions.

B	O	S	THUMOS14				ActivityNet-v1.3			
			0.3	0.5	0.7	Avg.	0.5	0.75	0.95	Avg.
·	·	·	70.5	53.9	23.8	50.3	52.62	32.59	7.43	32.40
✓	·	·	71.5	55.9	29.9	53.7	52.29	34.08	8.41	33.88
✓	✓	·	72.2	57.8	29.7	54.2	52.62	34.99	8.80	34.61
✓	·	✓	73.6	59.0	33.7	56.5	52.83	35.20	9.03	34.79
✓	✓	✓	75.2	60.4	32.8	57.3	52.71	35.60	9.08	34.92

Table 2: **Ablation on pretext tasks.** 'B', 'O', 'S' indicate Base, Ordinal, and Scale tasks, respectively.

Synthesis	0.50	0.75	0.95	Avg.
none	52.62	32.59	7.43	32.40
instance-wise	50.35	32.54	8.26	32.62
class-wise	52.71	35.60	9.08	34.92

Table 3: **Alternative to synthesis.** Experiments are conducted on ActivityNet-v1.3 with DAB-DETR.

On THUMOS14, the DETR models with LTP have reached the state-of-the-art performance. Especially for APs at high IoU thresholds such as 0.6 and 0.7, LTP remarkably improves the scores. We conjecture that long-term pretext tasks in the pre-training enable the model to precise localization. Interestingly for ActivityNet, the DETR-based models with LTP improve the performances with a large margin. Similar to the results of THUMOS14, LTP has significantly improved the APs at high IoU thresholds especially at 0.95.

Moreover, the table shows the results of the encoder-only models. We found that they also suffer from the data scarcity problem. As in the table, LTP significantly improves the performance by alleviating the issues from the data scarcity problem.

Conditional Tasks. We have conducted experiments on ActivityNet with DAB-DETR to prove the fidelity of each conditional task of LTP. Table. 2 shows the results of the ablation study on pretext tasks of LTP on THUMOS14 and ActivityNet with DAB-DETR. Each conditional task shows a clear improvement than only using the basic task. This demonstrates that the conditional pretext tasks can help for learning long-term dependency allowing for more precise localization. Also, the performance gain brought from the scale condition is slightly larger than the ordinal one. We conjecture that the scale condition effectively eases the problem of imbalanced performances over scales.

Synthesis. In order to validate the benefits of class-wise synthesis in LTP, we further study on another alternative for synthesis. One of the promising synthesis ways is the instance-wise method as in UP-DETR (Dai et al. 2021). To this end, we randomly sample temporal intervals from videos. Then the features of the interval are given as the condition, added to the action queries after projection in the same way as LTP.

Table. 3 shows the results for the alternative to class-wise synthesis on ActivityNet with DAB-DETR. As shown, the instance-wise synthesis does not bring noticeable performance gain for the DETR model. This is mainly because instance-wise synthesis for TAD handles a relatively short-term task than class-wise one so that it can suffer from a large task discrepancy between pre-training and fine-tuning. Therefore, we claim that class-wise synthesis is the key point for pre-training of DETR for TAD.

Complementary to TSP. Table. 4 shows the results of LTP with TSP (Alwassel, Giancola, and Ghanem 2021), a representative pre-training method for the feature extractor, on ActivityNet-v1.3 with DAB-DETR. The table demonstrates

Method	0.50	0.75	0.95	Avg.
I3D	52.62	32.59	7.43	32.40
I3D + LTP	52.71	35.60	9.08	34.92
TSP	50.74	32.89	7.86	32.83
TSP + LTP	53.80	35.95	8.45	35.42

Table 4: **Complementary to TSP.** It shows LTP is complementary to TSP on ActivityNet-v1.3 with DAB-DETR.

Method	Coverage					# of Instances			
	XS	S	M	L	XL	XS	S	M	L
BMN	8.7	24.4	35.0	55.2	70.7	53.4	16.1	9.8	5.1
GTAD	8.4	27.4	39.0	59.2	71.9	55.7	17.7	9.5	2.9
RCL	11.8	27.8	39.4	58.0	71.1	55.8	19.4	13.6	5.1
DAB-DETR	8.1	15.9	30.1	56.4	72.8	53.4	13.0	9.6	3.9
+ LTP	12.3	20.0	30.3	55.4	74.3	54.3	16.1	15.7	10.3

Table 5: **Imbalanced performances.** We conduct DETAD analysis on ActivityNet-v1.3 based on DAB-DETR.

that the performance improvement achieved by TSP persists in the integration of TSP and LTP, suggesting that the benefits of LTP complement pre-training the feature extractor.

Analysis

Imbalanced Performances. We observed that the model demonstrates significant imbalances in performance, primarily attributed to the limited data, particularly concerning the lengths of actions. Table. 5 shows the sensitivity analysis of DETAD (Alwassel et al. 2018) in terms of coverage (relative scale to video) and the number of instances. It illustrates the imbalanced performances of DAB-DETR, indicating a higher likelihood of the model predicting long actions. As a result, the performances for smaller scales and videos with numerous short instances are notably low.

However, the integration of LTP into DAB-DETR significantly alleviates the imbalanced performances, particularly in extra-short (XS) and short (S) scales. Of greater interest, the performances for videos with medium (M) and large (L) numbers of action instances exhibited remarkable improvement, achieving a new state-of-the-art score. This is primarily attributed to LTP preventing the model from overfitting to the action distribution of ActivityNet, which predominantly consists of a single long instance in the video.

Attention Collapse. Fig. 4 shows self-attention maps from the last layers of the DAB-DETR encoder ((a), (c)), and decoder ((b), (d)) for test samples of ActivityNet-v1.3. We observe that DETR for TAD severely suffers from the attention collapse problem. As in the attention maps ((a), (b)) of Fig. 4, all queries of DETR without LTP attend a few key elements. However, the attention maps ((c), (d)) of the model with LTP do not exhibit collapsed relations; instead, tokens show correlations with themselves or neighboring tokens.

Furthermore, we quantify the diversity of self-attention maps according to (Dong, Cordonnier, and Loukas 2021; Kim, Lee, and Heo 2023). The diversity $d(A)$ for the at-
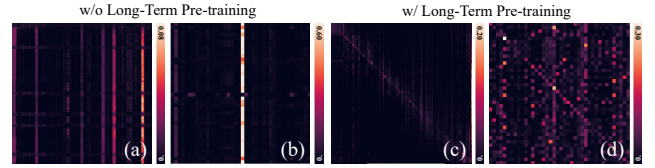

Figure 4: **Attention maps.** It shows self-attention maps from the last layers of the DAB-DETR encoder ((a), (c)) and decoder ((b), (d)) in test samples of ActivityNet-v1.3.
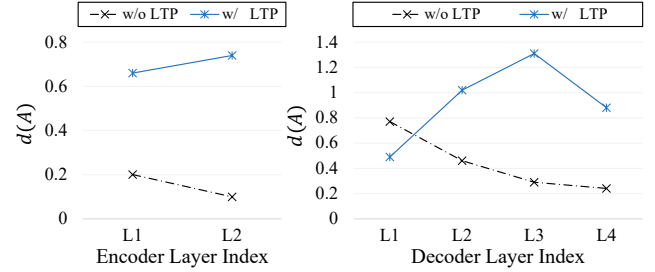

Figure 5: **Diversity of self-attention maps.** To analyze the effect of our pre-training for the attention collapse, we measure the diversity defined in Eq. 7 of the self-attention maps.

tention map A is the measure of the closeness between the attention map and a rank-1 matrix as defined as below:

$$d(A) = \|A - \mathbf{1}a^\top\|, \text{ where } a = \arg\min_{a'}\|A - \mathbf{1}a'^\top\|, \quad (7)$$

where $\|\cdot\|$ denotes the ℓ_1,ℓ_∞-composite matrix norm, a, a' are column vectors of the attention map A, and $\mathbf{1}$ is an all-ones vector. Note that the rank of $\mathbf{1}a^\top$ is 1, and therefore, a smaller value of $d(A)$ means A is closer to a rank-1 matrix.

Fig. 5 shows the diversity on each layer of the encoder and decoder for DAB-DETR models with and without LTP. The diversity is measured by the average value over all videos in the test set of ActivityNet-v1.3. As the model depth gets deeper, the diversity of the baseline decreases close to 0. However, the diversity of LTP keeps high, signifying that LTP eases the attention collapse. While our pre-training method does not employ a direct solution to the collapse, such as Self-DETR (Kim, Lee, and Heo 2023), the results suggest that furnishing the model with sufficient data diversity can alleviate the problem and lead to a more substantial performance improvement described in Table. 1.

Conclusion

In this paper, we have proposed a new pre-training strategy, Long-Term Pre-training (LTP) tailored for DETR in TAD, to migrate the problems caused by data scarcity. LTP has two main components: 1) class-wise synthesis, 2) long-term pretext tasks. Class-wise synthesis is to construct video features based on a target category to minimize the task discrepancy. In addition, two conditional pretext tasks foster the DETR model to learn long-term dependency. The extensive experiments have demonstrated that LTP relieves the problems of data scarcity with the remarkably improved performances.

References

Alwassel, H.; Giancola, S.; and Ghanem, B. 2021. Tsp: Temporally-sensitive pretraining of video encoders for localization tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3173–3183.

Alwassel, H.; Heilbron, F. C.; Escorcia, V.; and Ghanem, B. 2018. Diagnosing error in temporal action detectors. In *Proceedings of the European conference on computer vision (ECCV)*, 256–272.

Bar, A.; Wang, X.; Kantorov, V.; Reed, C. J.; Herzig, R.; Chechik, G.; Rohrbach, A.; Darrell, T.; and Globerson, A. 2022. Detreg: Unsupervised pretraining with region priors for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14605–14615.

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

Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; and Zagoruyko, S. 2020. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, 213–229. Springer.

Carreira, J.; and Zisserman, A. 2017. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6299–6308.

Dai, X.; Singh, B.; Zhang, G.; Davis, L. S.; and Qiu Chen, Y. 2017. Temporal context network for activity localization in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, 5793–5802.

Dai, Z.; Cai, B.; Lin, Y.; and Chen, J. 2021. Up-detr: Unsupervised pre-training for object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 1601–1610.

Dong, Y.; Cordonnier, J.-B.; and Loukas, A. 2021. Attention is not all you need: Pure attention loses rank doubly exponentially with depth. In *International Conference on Machine Learning*, 2793–2803. PMLR.

Fabian Caba Heilbron, B. G., Victor Escorcia; and Niebles, J. C. 2015. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 961–970.

Gao, J.; Yang, Z.; Chen, K.; Sun, C.; and Nevatia, R. 2017. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE International Conference on Computer Vision*, 3628–3636.

Jiang, Y.-G.; Liu, J.; Roshan Zamir, A.; Toderici, G.; Laptev, I.; Shah, M.; and Sukthankar, R. 2014. THUMOS Challenge: Action Recognition with a Large Number of Classes. http://crcv.ucf.edu/THUMOS14/.

Kang, H.; Kim, H.; An, J.; Cho, M.; and Kim, S. J. 2023. Soft-Landing Strategy for Alleviating the Task Discrepancy Problem in Temporal Action Localization Tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6514–6523.

Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, C.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; et al. 2017. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*.

Kim, J.; Choi, J.; Jeon, Y.; and Heo, J.-P. 2024. Boundary-Recovering Network for Temporal Action Detection. *arXiv preprint arXiv:2408.09354*.

Kim, J.; and Heo, J. 2019. Learning Coarse and Fine Features for Precise Temporal Action Localization. *IEEE Access*, 7: 149797–149809.

Kim, J.; Lee, M.; and Heo, J.-P. 2023. Self-Feedback DETR for Temporal Action Detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 10286–10296.

Lin, T.; Liu, X.; Li, X.; Ding, E.; and Wen, S. 2019. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 3889–3898.

Lin, T.; Zhao, X.; Su, H.; Wang, C.; and Yang, M. 2018. Bsn: Boundary sensitive network for temporal action proposal generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 3–19.

Liu, S.; Li, F.; Zhang, H.; Yang, X.; Qi, X.; Su, H.; Zhu, J.; and Zhang, L. 2022. DAB-DETR: Dynamic Anchor Boxes are Better Queries for DETR. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Liu, X.; Wang, Q.; Hu, Y.; Tang, X.; Zhang, S.; Bai, S.; and Bai, X. 2021. End-to-end Temporal Action Detection with Transformer. *arXiv preprint arXiv:2106.10271*.

Maaz, M.; Rasheed, H.; Khan, S.; Khan, F. S.; Anwer, R. M.; and Yang, M.-H. 2022. Class-agnostic object detection with multi-modal transformer. In *European Conference on Computer Vision*, 512–531. Springer.

Noci, L.; Anagnostidis, S.; Biggio, L.; Orvieto, A.; Singh, S. P.; and Lucchi, A. 2022. Signal propagation in transformers: Theoretical perspectives and the role of rank collapse. *Advances in Neural Information Processing Systems*, 35: 27198–27211.

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

Shi, D.; Zhong, Y.; Cao, Q.; Zhang, J.; Ma, L.; Li, J.; and Tao, D. 2022. React: Temporal action detection with relational queries. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*, 105–121. Springer.

Shou, Z.; Wang, D.; and Chang, S.-F. 2016. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1049–1058.

Su, H.; Gan, W.; Wu, W.; Qiao, Y.; and Yan, J. 2021. Bsn++: Complementary boundary regressor with scale-balanced relation modeling for temporal action proposal generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 2602–2610.

Tan, J.; Tang, J.; Wang, L.; and Wu, G. 2021. Relaxed Transformer Decoders for Direct Action Proposal Generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 13526–13535.

Tang, Y.; Han, K.; Xu, C.; Xiao, A.; Deng, Y.; Xu, C.; and Wang, Y. 2021. Augmented shortcuts for vision transformers. *Advances in Neural Information Processing Systems*, 34: 15316–15327.

Trockman, A.; and Kolter, J. Z. 2023. Mimetic initialization of self-attention layers. In *International Conference on Machine Learning*, 34456–34468. PMLR.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, L.; Xiong, Y.; Wang, Z.; Qiao, Y.; Lin, D.; Tang, X.; and Van Gool, L. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, 20–36. Springer.

Xu, H.; Das, A.; and Saenko, K. 2017. R-c3d: Region convolutional 3d network for temporal activity detection. In *Proceedings of the IEEE international conference on computer vision*, 5783–5792.

Xu, M.; Pérez-Rúa, J.-M.; Escorcia, V.; Martinez, B.; Zhu, X.; Zhang, L.; Ghanem, B.; and Xiang, T. 2021a. Boundary-sensitive pre-training for temporal localization in videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7220–7230.

Xu, M.; Perez Rua, J. M.; Zhu, X.; Ghanem, B.; and Martinez, B. 2021b. Low-fidelity video encoder optimization for temporal action localization. *Advances in Neural Information Processing Systems*, 34: 9923–9935.

Yeung, S.; Russakovsky, O.; Mori, G.; and Fei-Fei, L. 2016. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2678–2687.

Zhai, S.; Likhomanenko, T.; Littwin, E.; Busbridge, D.; Ramapuram, J.; Zhang, Y.; Gu, J.; and Susskind, J. M. 2023. Stabilizing transformer training by preventing attention entropy collapse. In *International Conference on Machine Learning*, 40770–40803. PMLR.

Zhang, C.; Yang, T.; Weng, J.; Cao, M.; Wang, J.; and Zou, Y. 2022. Unsupervised pre-training for temporal action localization tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14031–14041.

Zhang, C.-L.; Wu, J.; and Li, Y. 2022. Actionformer: Localizing moments of actions with transformers. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IV*, 492–510. Springer.

Zhao, Y.; Xiong, Y.; Wang, L.; Wu, Z.; Tang, X.; and Lin, D. 2017. Temporal action detection with structured segment networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 2914–2923.

Zhu, X.; Su, W.; Lu, L.; Li, B.; Wang, X.; and Dai, J. 2021. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Additional Details

Feature Extractor. We use the features of I3D or TSN pre-trained on Kinetics. We follow video pre-processing process as in I3D, which uses 25 FPS for frame extraction and re-sizes the frames so that they have 256 pixels in their longer side (width or height). We extract temporal features with 16-frame non-overlapping windows, and use the final outputs with global average pooling (GAP). Each temporal feature is corresponding to 16 frames since the final temporal resolution of I3D is $1/16$ of the input video. Note that the feature extractor is fixed during pre-training and fine-tuning. The way of extracting follows the backbone model.

DETR Architectures. In this paper, we introduce two types of DETR baselines: 1) DAB-DETR, 2) Deformable-DETR in a temporal version. These baselines do not have any additional module so that we can solely show the benefits of LTP. The configure of DAB-DETR follows Self-DETR while that of Deformable-DETR is aligned with TadTR. The hidden dimension of DETR models is 256 as the DETR baselines for TAD. The number of queries is 40. Also, the number of the encoder and decoder layers is 2, and 4, respectively. all other configurations follow the DETR baselines (Liu et al. 2021; Shi et al. 2022; Kim, Lee, and Heo 2023) for fair comparison.

Pre-training Set-up. For pre-training, we utilize the Kinetics-400 dataset, the same dataset on which the feature extractor is pre-trained, which contains approximately 300K videos. The length of the temporal features to synthesize is 192, which is the same as that of ActivityNet-v1.3. Also, the maximum number of action instances $N_{\max}$ is 12 since the temporal length of each video is 16. We set the condition probability p_{cond} for long-term pretext tasks as 0.5. When two videos of the same target class are overlapped in class-wise synthesis, we group them as a single long action instance. We train the DETR model with a mini-batch size of 256 by using AdamW with initial learning rate of 1.0×10^{-4} during 15 epochs. The learning rate is scheduled by cosine annealing with a warm-up of 5 epochs.

Fine-tuning Set-up. As down-stream tasks, the pre-trained models are fine-tuned on ActivityNet and THUMOS14. The configuration of fine-tuning follows the baselines. Note that the parameters of the classification head are not initialized with the pre-train weights because the number of classes does not match. Also, we do not add any vector to the action queries as condition when fine-tuning. The lengths of the temporal features are 192, and 128 for ActivityNet-v1.3, and THUMOS14, respectively. We fine-tune the DETR model with a mini-batch size of 16 by using the AdamW optimizer for both benchmarks. As for ActivityNet, we train the model with the learning rate of 1.0×10^{-4} during 20 epochs. Also, we schedule the learning rate by cosine annealing with a warm-up of 5 epochs. On THUMOS14, we train the model with the learning rate of 2.0×10^{-4} during 120 epochs. We schedule the learning rate multiplied by 0.1 at 80 and 100 epochs, respectively.

Details for the Synthesis Ablation. In Table. 2 of the paper, we have conducted experiments for an alternative to

Method	Coverage					# of Instances			
	XS	S	M	L	XL	XS	S	M	L
BMN	8.7	24.4	35.0	55.2	70.7	53.4	16.1	9.8	5.1
GTAD	8.4	27.4	39.0	59.2	71.9	55.7	17.7	9.5	2.9
RCL	11.8	27.8	39.4	58.0	71.1	55.8	19.4	13.6	5.1
w/o LTP	12.1	19.4	29.2	52.3	70.7	51.7	15.6	15.2	8.8
w/ LTP	13.2	22.9	33.8	56.3	74.7	55.2	18.3	16.3	9.4

Table A1: **DETAD analysis.** We conduct DETAD analysis on ActivityNet-v1.3 based on Deformable-DETR.

the class-wise synthesis of LTP. We employ the way of UP-DETR as the alternative synthesis, named 'instance-wise' in the table. To be more specific, we first randomly sample features from a video in the dataset. Then we randomly select an interval to temporally crop the features as the target. The cropped features are average-pooled and added to the action queries as a condition. The DETR model is trained to find the interval from the entire video as UP-DETR. As shown in Table. 2 of the paper, we found that this way is not valid for TAD.

Details for the Experiments with TSP. In Table. 5 of the paper, we have validated the complementary benefits with TSP (Alwassel, Giancola, and Ghanem 2021), a pre-training method for the feature extractor. To this end, we first extract TSP features of the Kinetics-400 dataset. We use the official codes from (Alwassel, Giancola, and Ghanem 2021) with a stride 16, which means a non-overlapping set-up. After feature extraction, we pre-train and fine-tune the DAB-DETR model as done in the paper.

Additional Results

DETAD Analysis for Deformable-DETR. Table. A1 shows the results of the DETAD (Alwassel et al. 2018) analysis on ActivityNet-v1.3 with Deformable-DETR. Similar to the results of DAB-DETR in Table. 4 of the paper, LTP significantly improves the performances of extra-small (XS) and small (S) actions. Moreover, the performances for the middle (M) and large (L) numbers of actions instances are largely boosted by LTP.

Data Scarcity. To further analyze the benefits of LTP under data scarcity, we conducted additional experiments using a reduced amount of training data from ActivityNet-v1.3 with DAB-DETR. To achieve this, we randomly sampled and utilized training data ranging from 5% to 90%, while maintaining the entirety of the test set.

Fig. A2 presents the results in mAP based on the DAB-DETR model, both with and without LTP. DETR with LTP consistently outperforms the baseline without LTP in terms of mAP for all configurations. Despite the rapid degradation in the performance of DAB-DETR without LTP, the model with LTP exhibits a much slower decrease in performance. This demonstrates that long-term pre-training effectively mitigates the issue of data scarcity.

Qualitative Results. Fig. A1 illustrates test samples in ActivityNet-v1.3 based on DAB-DETR models with and
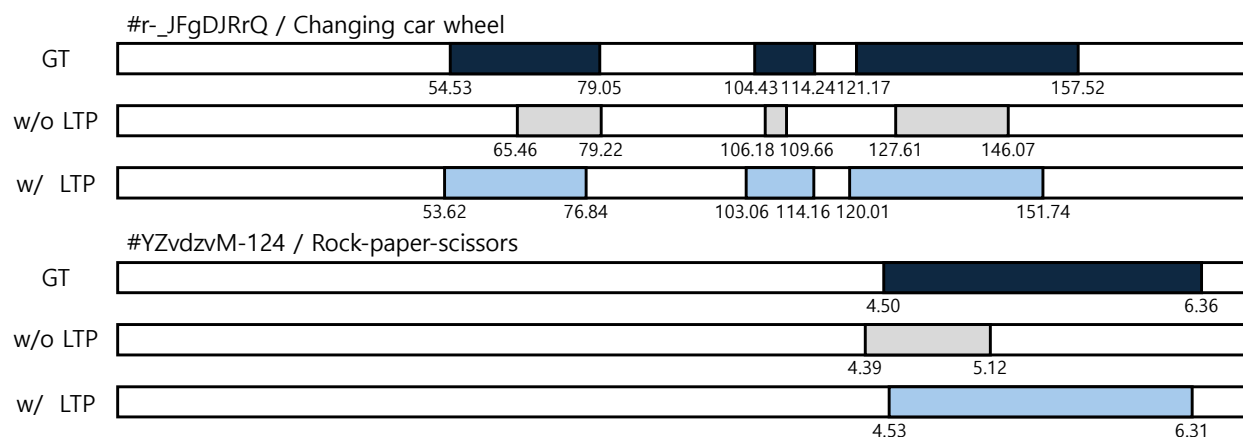
Figure A1: **Qualitative results with and without LTP.** The figure depicts test samples on ActivityNet-v1.3 with DAB-DETR.
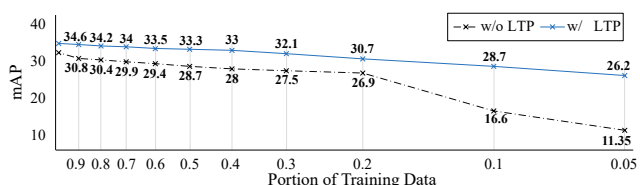

Figure A2: **Data scarcity.** We conducted experiments for training data ablation of ActivityNet-v1.3 with DAB-DETR.
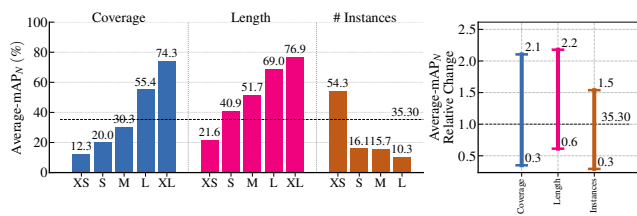

Figure A3: **DETAD analysis on ActivityNet.** It shows the DETAD (Alwassel et al. 2018) sensitivity analysis on ActivityNet with DAB-DETR.

without LTP. As shown, DAB-DETR with LTP outputs more precise results.

Visualization of DETAD Analysis for DAB-DETR. Fig. A3 depicts the results of DETAD (Alwassel et al. 2018) sensitivity analysis for DAB-DETR on ActivityNet-v1.3.