TALLFormer: Temporal Action Localization with a Long-memory Transformer

Feng Cheng[1] Gedas Bertasius[1]
{fengchan,gedas}@cs.unc.edu

Department of Computer Science, University of North Carolina at Chapel Hill

Abstract. Most modern approaches in temporal action localization divide this problem into two parts: (i) short-term feature extraction and (ii) long-range temporal boundary localization. Due to the high GPU memory cost caused by processing long untrimmed videos, many methods sacrifice the representational power of the short-term feature extractor by either freezing the backbone or using a small spatial video resolution. This issue becomes even worse with the recent video transformer models, many of which have quadratic memory complexity. To address these issues, we propose TALLFormer, a memory-efficient and end-to-end trainable **T**emporal **A**ction **L**ocalization transformer with **L**ong-term memory. Our long-term memory mechanism eliminates the need for processing hundreds of redundant video frames during each training iteration, thus, significantly reducing the GPU memory consumption and training time. These efficiency savings allow us (i) to use a powerful video transformer feature extractor without freezing the backbone or reducing the spatial video resolution, while (ii) also maintaining long-range temporal boundary localization capability. With only RGB frames as input and no external action recognition classifier, TALLFormer outperforms previous state-of-the-arts by a large margin, achieving an average mAP of 59.1% on THUMOS14 and 35.6% on ActivityNet-1.3. The code is public available[1].

1 Introduction

With the rapid growth of video media, video understanding has become an important area of computer vision. As a fundamental task in video understanding, Temporal Action Localization (TAL) aims to localize temporal boundaries and classify the actions for each action instance in a long untrimmed video.

Because many actions span long temporal extent (e.g., 50-100s), most prior approaches in TAL [27,25,53,6,2,62,33,23,45], divide this problem into two parts: (i) short-term feature extraction and (ii) long-range temporal boundary localization. As shown in Fig. 1, the first part involves sampling many consecutive short clips (e.g., each spanning 1-2 seconds) from a long untrimmed video and extracting short-term features from them. In the second part, the model uses the extracted features of all short-term clips (i.e., spanning the entire duration of an

[1] https://github.com/klauscc/TALLFormer.

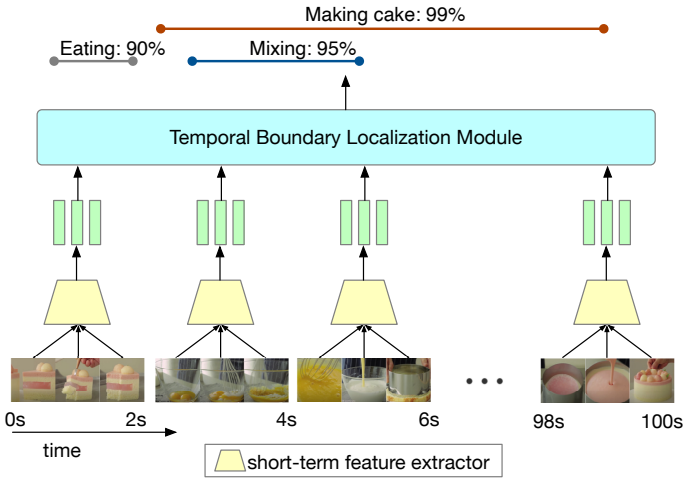

Fig. 1: A general framework for temporal action localization (TAL). The short-term feature extractor extracts the features for each short-term clip. Then, the long-term temporal boundary localization module uses the features of all short-term clips in the video to predict the action boundaries and categories. Due to excessive training time and GPU memory cost, many prior TAL methods restrict the representational power of the short-term feature extractor by either freezing the backbone or operating on small spatial video resolution. While effective at reducing the computational burden, these techniques also significantly degrade TAL performance.

untrimmed video) for predicting action boundaries and categories. Thus, based on these observations, it is natural to conclude that an ideal TAL model should consist of (i) a powerful short-term feature extractor and (ii) a precise temporal boundary localization module.

However, due to the high GPU memory cost needed to process long untrimmed videos, the majority of existing methods sacrifice the representational power of short-term feature extractor, by either freezing the backbone [27,25,53] or using a very small spatial video resolution (e.g., 96×96) [23,45]. While both of these techniques are highly effective at reducing GPU memory consumption, they also degrade the quality of extracted short-term features, which leads to a significantly lower TAL accuracy. For example, as shown in Table 1, in order to save memory, reducing spatial resolution from 168×168 to 112×112 leads to 2.3% mAP drop; using a smaller backbone also reduces the mAP by 2.1%; freezing the backbone leads to a severe drop in mAP ($\sim$8-9%).

In parallel, we note that the recent introduction of powerful video transformer models [3,32] have achieved impressive results on various video understanding problems such as action recognition. However, these models have made the above-described GPU memory issues even worse. Due to the quadratic complexity of self-attention, video transformers require even more GPU memory than traditional CNNs. As a result, it is challenging to adapt these models to the TAL task, which

Table 1: We study several important factors for short-term feature extraction on THUMOS14: (i) spatial video resolution, (ii) transformer backbone complexity, and (iii) the number of frozen backbone stages. For these experiments, we use Swin [32], which consists of 4 stages where i frozen stages means that the first i stages in the backbone are frozen. We use DaoTAD [45] codebase to conduct these experiments. Additionally, we note that in all of these experiments, we incorporate Checkpointing [7] to reduce GPU memory usage. GPU Memory (Mem) is measured in Gigabytes. Based on these results, we note that **(a)** increasing the spatial resolution leads to large mAP improvement($\sim$2.3%) but also quadratic memory consumption, **(b)** using larger backbones also improves the mAP, and lastly, **(c)** freezing the backbone leads to a severe drop in performance ($\sim$8-9%).

(a) Spatial video resolution (SVR) analysis. The backbone is Swin-T with the first 2 stages frozen.

SVR	mAP(%)	Mem
112×112	52.8	**15**
168×168	**55.1**	34
224×224	-	OOM

(b) Studying the backbone complexity. Spatial resolution is 112×112. The first 2 stages of backbone are frozen.

Backbone	mAP(%)	Mem
Swin-T	52.8	**15**
Swin-S	53.3	17
Swin-B	**54.7**	20

(c) The number of frozen backbone stages (FBS). Spatial resolution is 112×112. The backbone is Swin-T.

FBS	mAP(%)	Mem
4	44.3	**3**
2	52.8	14
0	**53.8**	26

generally requires a lot of GPU memory even when using CNN-based models. The commonly used GPU memory saving techniques such as Checkpointing [7], Mixed Precision [34], can alleviate these computational issues. However, as shown in Table 1, even when using these techniques, the GPU memory cost of applying video transformers (e.g., VideoSwin [32]) to TAL is very large.

Thus, with these computational issues in mind, we propose TallFormer, a memory-efficient and end-to-end trainable **T**emporal **A**ction **L**ocalization Transformer with a **L**ong-memory mechanism. Our key observation is that most videos are highly redundant, i.e., their content changes little in most neighboring frames. This raises the question of whether every single frame from a long untrimmed video needs to be processed during each training iteration. Motivated by this observation, we design TallFormer to process only a fraction of randomly selected frames at each training iteration, which significantly reduces the training time and GPU memory requirements. For the remaining (i.e., not selected) video frames, the video features are sampled from long-term memory, which stores the features of all previously processed frames for that particular video. Note that the features from long-term memory do not have to be re-computed online, and they also do not require backpropagating the gradients, which makes long video processing much more efficient.

As the short-term feature extractor evolves throughout training, the video features in long-term memory are also evolving, i.e., the newly computed features for a given video are used to replace the old features in long-term memory.

Compared to previous TAL approaches, TallFormer has several main advantages. First, our model can be trained end-to-end on long, high spatial resolution videos beyond the constraints of finite GPU memory. Second, our framework is flexible as we can incorporate any state-of-the-art short-term video transformer model into TallFormer, thus, benefiting from future improvements in the video transformer design. Lastly, unlike many previous TAL methods [27,25,53,2,62,33] that rely on external action recognition classifiers, TallFormer is a unified framework that predicts action boundaries and categories with a single model. Despite being simpler, and only operating on RGB inputs, TallFormer achieves an average mAP of 59.1% on THUMOS14 and 35.6% on ActivityNet-1.3, thus, outperforming the current state-of-the-arts by 7.1% and 1.2% respectively.

2 Related Work

Action Recognition. Action recognition is a fundamental short-term modeling task in video understanding. With the success of deep learning, a vast array of methods [42,47,43,46,5,11,12,21,24,22,48,55] utilize 2D and 3D CNNs to achieve impressive performance on standard action recognition benchmarks [5]. Recently, Vision Transformer-based methods [3,32,10,56] have been shown to outperform previous CNN-based methods by a large margin. Due to the large scale pretraining on action recognition datasets, the pretrained models from this domain are widely used in temporal action localization as a short-term feature extractor. One limitation of modern video transformer models is that due to the quadratic memory complexity of self-attention [44], these models are slow to train and they require a lot of GPU memory. As a result, it is difficult to apply them to long-term modeling tasks such as temporal action localization.

Temporal Action Localization (TAL). Due to finite GPU memory constraints, most existing methods [27,25,53,6,2,62,33,41,60,1] use pre-extracted action recognition features as inputs to the TAL model. However, since those features are extracted using models [17,5] that are pretrained on different datasets, using these features for TAL often leads to suboptimal performance. To address these issues, recent methods AFSD [23] and DaoTAD [45] proposed end-to-end trainable frameworks. However, to fit into finite GPU memory, these models operate on very low spatial video resolutions (e.g., 96×96 and 112×112 respectively), which leads to a significant drop in TAL accuracy. To the best of our knowledge, none of the existing methods are capable of end-to-end training with both high spatial resolution and long temporal extent. We aim to address this issue by proposing a simple, end-to-end trainable, transformer-based TAL method that can operate on long high-resolution video inputs.

Besides end-to-end training ability, we also note that most TAL methods can be categorized into two groups: (i) single-stage detectors, and (ii) two-stage detectors that require external action recognition classifiers. One-stage detectors [26,29,59,50] perform action localization and classification at the same time. In comparison, the two-stage methods [27,25,53,23,14,57,30,13,2,40,36,38,63] only

predict action boundaries and then use the predictions of an external action recognition classifier to assign an action class to a given video segment. Despite the elegance and simplicity of one-stage methods, the two-stage methods typically have a much higher detection accuracy. In this work, we will show that even without relying on the external action recognition classifier, our TALLFormer still achieves state-of-the-art results on several major TAL benchmarks.

Memory-saving Techniques. Applying transformer-based methods to TAL poses many GPU memory challenges due to the quadratic memory complexity of self-attention. There are several general memory-saving techniques, including Gradient Checkpointing [7] and Mixed Precision [34], which reduce the GPU memory usage by about 50%. We note that our proposed approach is complementary to these techniques. In fact, we use Gradient Checkpointing [7] in many of our experiments, thus, demonstrating that our proposed method works well in conjunction with these prior memory-saving techniques.

Furthermore, we note that several methods from Natural Language Processing (NLP) such as LinFormer [49] and Performer [9] propose to reduce the memory complexity of standard self-attention by approximating the attention using low-rank matrix decomposition. While being effective in NLP, those approximation methods work poorly when applied to video recognition [35].

3 TALLFormer

Given an untrimmed video $V = \{x_t\}_{t=1}^{T} \in \mathbb{R}^{C \times T \times H \times W}$ with T RGB frames, our TALLFormer model aims to predict a set of action instances $\Phi_V = \{\phi_m\}_{m=1}^{M}$ where M is the number of action instances in V. Each action instance $\phi_m = (s_m, e_m, c_m, p_m)$ is a four-element tuple that represents the start timestamp of action, end timestamp of action, action class and probability of this instance respectively.

As shown in Fig. 2, TALLFormer consists of four components: (i) a short-term Transformer encoder, (ii) a long memory module, (iii) a temporal consistency module and (iv) a temporal boundary localization module. First, we randomly sample a subset of short video clips, and process them using the short-term Transformer encoder. The remaining features are directly sampled from long-term memory, which stores previously computed features of all frames for that particular video input. Afterward, all of these features (i.e., from the short-term Transformer encoder and long-term memory) are fed into a temporal consistency module that effectively fuses them in order to map them to a similar feature space, i.e., to alleviate potential issues caused by differing feature distributions from the feature extractor and long-term memory. Lastly, the temporal boundary localization module processes these features and produces temporal boundaries and action categories for each detected action instance. We now describe each of these components in more detail.

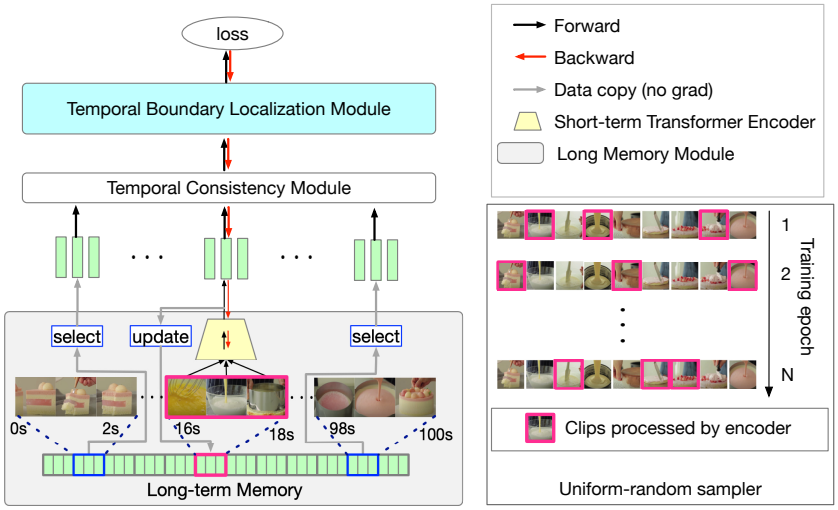

Fig. 2: An illustration of our proposed TALLFormer model. Our method consists of four high-level components: (i) a short-term Transformer encoder, (ii) a long memory module, (iii) a temporal consistency module and (iv) a temporal boundary localization module. The short-term Transformer encoder only extracts features from a few randomly sampled video clips. The rest of the features are sampled from long-term memory. All features are then fed into a temporal consistency module to ensure smooth feature fusion. Lastly, the temporal boundary localization module outputs temporal boundaries and action categories for each action instance. Afterward, the features extracted by the short-term encoder are used to update the corresponding features in long-term memory.

3.1 Short-term Transformer Encoder

Our Shor-term Transformer Encoder considers many consecutive short clips (i.e., spanning 1-2 seconds) from a long untrimmed video. In order to avoid computing dense features for every single clip, we randomly sample a fixed number of such clips and feed them into our encoder.

Formally, for each input video V we divide it into N_c non-overlapping clips $c = \{c_m\}_{m=1}^{N_c}$ where $c_m \in \mathbb{R}^{L_c \times H \times W \times 3}$. The input video is shifted at most L_c frames to ensure the clip-division changes at each epoch. A uniform sampler first samples the indices $I \in \mathbb{R}^{N_s}$ of clips that will be processed by the encoder. The indices of the remaining (i.e., not sampled) clips are denoted as $I' \in \mathbb{R}^{N_c - N_s}$. The encoder then processes each sampled clip c_i to extract low-dimensional features $f_{c_i} \in \mathbb{R}^{L_f \times C_f}$ to produce features $f_I^{(s)} = \{f_{c_{I_1}}, f_{c_{I_2}}, ..., f_{c_{I_{N_s}}}\} \in \mathbb{R}^{N_s \times L_f \times C_f}$.

Note that during each training iteration, the Transformer encoder only processes a small fraction of clips from the whole input video. The remaining clips are sampled from the long memory module (described in Sec. 3.2). This enables TALLFormer to be trained end-to-end on long high spatial resolution videos without (i) reducing the spatial video resolution, (ii) freezing the backbone, or (iii) resorting to a weak short-term feature extraction backbone. We use the

recent VideoSwin [32] as our short-term Transformer encoder, which achieved impressive results on several popular action recognition benchmarks [5,15].

3.2 Long Memory Module

Our proposed Long Memory Module (LMM) enables TALLFormer to be trained on long and high-resolution videos. Inspired by [8,58], we propose LMM to cache the features computed by our short-term Transformer encoder for all short-term video clips. For the remaining clips (denoted by the indices I') that are not processed by the short-term Transformer encoder, LMM samples the features $f_{I'}^{(l)} \in \mathbb{R}^{(N_c - N_s) \times L_f \times C_f}$ from long-term memory. Following this step, we then update long-term memory with the features $f_I^{(s)}$ extracted by the short-term Transformer encoder. Note that before training, we initialize the LMM with the features extracted by our short-term Transformer encoder.

Such a scheme works well in the TAL setting because the short-term Transformer encoder is already pretrained on a large-scale external action recognition dataset (e.g., Kinetics) and thus, it evolves more slowly than the other modules in the network (i.e., it uses a smaller learning rate than the other parts of the network). Thus, "approximating" short-term features with the features from LMM provides large efficiency gains (both in terms of training time and GPU memory), while still achieving excellent TAL accuracy, which we demonstrate in our experimental section. Compared to prior methods [51,16,54,58] that use memory bank as auxiliary information, our LMM serves as an approximation to the short-term encoder. Both the features from LMM and short-term encoder are directly used to produce the final predictions of our method.

Overall, compared to standard end-to-end training, TALLFormer only needs to process a fraction of input clips, which saves the memory and computational cost by a rate of $r = \frac{N_s}{N_c}$. This then allows us to (i) use a powerful transformer-based feature extractor without freezing its backbone or reducing the spatial video resolution and (ii) still maintain the ability to precisely localize long-range temporal boundaries of actions. Note that during inference, we extract all features using a short-term Transformer encoder (i.e., without using LMM).

3.3 Temporal Consistency Module

Due to different feature distributions between (i) the online extracted Transformer features $f_I^{(s)}$ and (ii) LMM-cached offline features $f_{I'}^{(l)}$, we need to reduce temporal inconsistency among clip-level features across the whole input video. To be more precise, the features that are processed online (i.e., using our short-term Transformer encoder) are extracted using the latest short-term encoder. In contrast, most clip-level features stored in the LMM are extracted using the same short-term Transformer encoder but from the previous iterations. Thus, the short-term features associated with different clips might have different feature distributions, which can potentially degrade TAL performance. To address this issue, we propose a simple, yet effective Temporal Consistency Module (TCM).

The idea is to make the features from both sources more consistent by allowing them to interact with each other. Due to the effectiveness of standard self-attention to capture global long-range dependencies, we design TCM as an L attention layer subnetwork. Formally, given the video features $g = [f_I^{(l)}; f_{I'}^{(s)}]$, the TCM refines the features using three Transformer layers:

$$h^{(i)} = \text{TransformerLayer}(h^{(i-1)}) \tag{1}$$

where $i \in [1, L]$ is the layer index, $h^{(0)} = g$ and $h^{(L)}$ is the refined features of TCM. The TransformerLayer uses relative positional encoding as in Swin [31], GELU [18] activation, and Droppath [19].

Conceptually, our self-attention-based TCM subnetwork allows our model to refine potentially inconsistent features by incorporating temporal information from the entire untrimmed input video into feature vectors associated with individual video clips. In our experimental section, we demonstrate the effectiveness of such long-range TCM module.

3.4 Temporal Boundary Localization Module

The Temporal Boundary Localization Module (TBLM) utilizes the features of all clips produced by the TCM to predict the action boundaries and categories. The TBLMs in most existing methods [27,25,53,2,62,33,23,41,60] especially for difficult datasets (e.g., ActivityNet [4]) are two-stage detectors that require external action classification classifiers, which is costly and cumbersome. Our analysis into this problem reveals that the reason that many prior methods rely on external classifiers is because of a weak short-term encoder. Specifically, as discussed above, many prior methods have to either freeze the backbone or use small spatial video resolution in order to save GPU memory. This then leads to poor action classification performance, which requires these methods to adapt an external action recognition classifier. In contrast, we note that TALLFormer utilizes a strong short-term encoder while achieving strong performance on both action classification and localization using a one-stage TBLM.

We build upon the existing methods [23,45] by simply adding a shared linear action recognition classifier to each action proposal. For datasets where each video only contains one action category such as ActivityNet [4], we add the linear classifier on the temporally averaged features of the TCM with a 50% dropout. Due to the strong representational power of TALLFormer, this simple modification achieves a high action classification accuracy and thus, eliminates the necessity for an external action recognition classifier. We use the same loss functions as the previous methods [23,45] and Focal Loss [28] for the added linear action recognition layer. See more details in Appendix. A.2.

4 Experiments

4.1 Datasets and Evaluation Metrics

Datasets. We conduct our evaluations on the two commonly-used benchmark datasets THUMOS14 [20], and ActivityNet-1.3 [4]. THUMOS14 contains 200

untrimmed validation videos and 213 untrimmed testing videos with temporal annotations from 20 categories. ActivityNet-1.3 contains 15,000 videos for training and 5,000 videos for validation. Additionally, we also evaluate on the large-scale HACS-Segment [61] dataset, which contains 38K untrimmed videos for training and 6K for validation. Following previous works [25,23,45], on THUMOS14, we train on the validation set and evaluate on the test set. On ActivityNet-1.3 and HACS-Segment we train on the training set and evaluate on the validation set.

Evaluation Metrics. As is standard, we use mean Average Precision (mAP) to report our results. The Intersection over Union (IoU) thresholds are set to $[0.3 : 0.1 : 0.7]$ for THUMOS14 and $[0.5 : 0.05 : 0.95]$ for ActivityNet-1.3 and HACS-Segment.

4.2 Implementation Details

The flexibility of our framework allows us to consider any transformer-based model as our short-term feature extractor. Due to its superior accuracy, we adopt Video Swin Transformer [32] pretrained on Kinetics-400 [5]. The number of layers L in TCM is set to 3 and Droppath rate is 0.1. Our temporal boundary localization module (TBLM) is designed using the techniques from DaoTAD [45] and AFSD [23] for THUMOS14 and ActivityNet-1.3 respectively. Unlike previous methods that operate on (i) RGB and (ii) optical flow frames inputs, our TALLFormer only uses RGB frames. The frames are resized to 256×256 and cropped to 224×224 unless stated otherwise.

During training, we apply common data augmentations on both datasets, including random crop, random horizontal flipping, random rotate and other photometric distortions such as random brightness and contrast. For the other training and inference details, we follow DaoTAD for THUMOS14 and AFSD for ActivityNet-1.3 with minor modifications as below. The inference and all the other settings are kept the same. Gradient Checkpointing [7] is applied to all our models. Our models are trained on $4\times$ RTX A6000 GPUs.

THUMOS14. We extract RGB frames with 15fps. Because 99.5% of action instances in the validation set span less than 32 seconds, we consider 480 frames as inputs. The batch size is set to 4 on each GPU. Since in DaoTAD, clip features are temporally downsampled by 8, but in Swin Transformer the temporal downsampling rate is only 2, we add two Convolutional layers with stride 2 before the TCM to keep the same temporal downsampling rate as in DaoTAD.

ActivityNet-1.3. As is done in prior work [23], we resize all videos to 768 RGB frames. The batch size is set to 1 on each GPU. For simplicity, we remove the boundary consistency learning module in AFSD. Our model is trained for 10 epochs instead of 16 as used in the original AFSD.

4.3 Comparison with Short-term and Long-term Baselines

We next conduct a thorough empirical study investigating the importance of short-term vs. long-term modeling for the TAL task. We focus our comparisons

Table 2: Comparing TALLFormer with several of our own short-term and long-term baselines on THUMOS14. We use a powerful video Swin-B as the Feature Extractor (**FE**) for all models. All models operate on videos with a Spatial Resolution (**SR**) of 224×224. LT-Frozen is a long-term baseline that uses a frozen feature extractor but long Temporal Support (**TS**). ST-E2E is a short-term end-to-end trainable baseline with an unfrozen backbone but short temporal support. TALLFormer provides the best trade-off between short-term and long-term modeling among these baselines.

Mem Cap	Model Type	Short-term			Long-term		mAP(%)			
(GB/GPU)		FE	E2E	SR	TS(s)	Fps	0.3	0.5	0.7	Avg.
12	LT-Frozen	Swin-B	✗	224	32	15	70.2	55.4	29.4	52.7
	ST-E2E	Swin-B	✓	224	8	8	63.1	46.6	16.0	42.9
		Swin-B	✓	224	32	2	55.4	32.2	8.4	31.9
		Swin-B	✓	224	4	15	50.5	33.9	13.4	32.8
	TALLFormer	Swin-B	✓	224	32	15	**76.1**	**63.1**	**34.2**	**59.0**
32	LT-Frozen	Swin-B	✗	224	32	15	70.2	55.4	29.4	52.7
	ST-E2E	Swin-B	✓	224	32	8	72.7	59.8	33.3	56.3
		Swin-B	✓	224	12	15	72.8	58.6	30.0	55.1
	TALLFormer	Swin-B	✓	224	32	15	**76.0**	**63.2**	**34.5**	**59.2**

on three of our baselines, which are compared under the same finite GPU memory constraints, i.e., either 12GB (RTX 3080) or 32GB (Tesla V100).

LT-Frozen: For this Long-Term modeling baseline, we use a powerful yet frozen Video Swin-B as the feature extractor. A similar strategy of freezing the feature extractor is commonly used in many prior methods [27,25,53] as the GPU memory savings from freezing the backbone enable long-range temporal modeling needed by TAL. All models are trained under a finite GPU memory constraint.

ST-E2E: Unlike LT-Frozen baseline, the Short-Term End-to-End trainable baseline uses a Swin-B feature extractor (*not frozen*) that operates on 224×224 video frame inputs. While benefiting from end-to-end trainability, due to the GPU memory limitation, this baseline can only span either (i) short temporal extent with dense video frame sampling or (ii) long temporal extent with sparse video frame sampling. We study both of these ST-E2E variants.

TALLFormer: Compared to the previous two baselines, we believe that our approach achieves the best trade-off between short-term and long-term modeling. In other words, the short-term feature extractor in our framework can be trained end-to-end on high spatial resolution videos. Furthermore, our long-term memory module enables the model to maintain strong long-term modeling capability for precise temporal boundary localization.

Analysis. From Table 2, we observe that long-term modeling is important, i.e., reducing the temporal support in ST-E2E leads to sub-optimal performance. With a 32GB GPU memory limit, ST-E2E with a maximum temporal support of 12 seconds achieves 4.1% lower average mAP than TALLFormer with 32 second

temporal support. We also point out that the ST-E2E variant that spans 32 seconds using sparsely sampled frames (i.e., 8 vs. 15 fps) also produces 2.9% worse performance than TALLFormer. We observe similar trends for the models trained under the 12GB GPU memory constraint. Additionally, our results indicate that the the end-to-end training of a short-term feature extractor is also important as LT-Frozen baseline achieves 6.5% lower accuracy than TALLFormer. We observe this trend in both 12GB and 32GB GPU memory settings.

Overall, we can conclude that TALLFormer achieves the best accuracy-memory trade-offs under both 12GB and 32GB GPU memory constraints. Specifically, TALLFormer outperforms the LT-Frozen and ST-E2E baselines by a large margin especially with tighter GPU memory constraints (i.e. 12GB).

4.4 Comparison to the State-of-the-Art

Next, we compare TALLFormer to the state-of-the-art methods as shown in Tab. 3. The upper part of Tab. 3 includes methods that operate on pre-extracted action recognition features. The middle section of Tab. 3 includes recently proposed end-to-end trainable methods, AFSD and DaoTAD, that operate on small spatial video resolutions (i.e., 96×96 and 112×112 respectively) to fit into GPU memory. Lastly, in the bottom part of the table, we include our TALLFormer, which can be trained end-to-end on long 224×224 videos.

We experiment with three variants of our method. First, we introduce a variant, named TALLFormer-12, which is cheap enough to fit in a 12GB memory GPU, with two backbones I3D and Swin-B for fair comparison with prior methods. Additionally, for the first variant (using I3D backbone), we use the clip sampling rate $r = 0.4$ on THUMOS14 and $r = 1/3$ on AcitivityNet-1.3, whereas for the latter variant (using Swin-B backbone) the clip sampling rate is set to $r = 0.15$ for THUMOS14 and $r = 1/8$ for ActivityNet-1.3. Lastly, our best performing variant is TALLFormer-32 with Swin-B as its backbone, which uses a clip sampling-rate of $r = 0.4$ for THUMOS and $r = 0.375$ for ActivityNet. We set these sampling rates so that our model would fit in the available GPU memory.

THUMOS14. The results in Tab. 3 (the left part of the table), indicate several interesting trends. First, we notice that despite using a small spatial resolution, the end-to-end trainable methods such as AFSD and DaoTAD, outperform methods that operate on pre-extracted action recognition features by a large margin. Second, our results indicate that the memory-constrained TALLFormer-12 with an I3D backbone outperforms a strong AFSD baseline by a substantial margin according to all evaluation metrics. Moreover, when increasing the GPU memory constraints, TALLFormer-12 achieves 7.2% higher accuracy on average than AFSD. We note that the GPU consumption for TALLFormer is 29GB, which is still within the capacity of the mainstream Tesla V100 GPUs. We also point out that even when using the same amount of GPU memory as prior methods, our method still largely outperforms previous SOTAs, i.e., TALLFormer-12 with I3D backbone and VSwin-B backbone outperforms AFSD by 1.9% and 7.0%.

ActivityNet. First, we point out that all the previous methods achieve strong TAL results on ActivityNet while relying on an external action recognition

Table 3: Comparison to the state-of-the-art on THUMOS14 and ActivityNet-v1.3. **FE** and **E2E** denote the feature extractor backbone and whether a method is end-to-end trainable respectively. The feature extractor backbones include TS [39], I3D [5], P3D [37] and Swin-B [32] (denoted as SW). **Flow** and **Ext. Cls.** denote whether each method uses optical flow as input and whether an external action recognition classifier is needed respectively. Note that AFSD relies on an external classifier on ActivityNet-1.3.

Method	FE	E2E	Flow	Ext. Cls.	mAP								Mem (GB)
					THUMOS14				ActivityNet-1.3				
					0.3	0.5	0.7	Avg.	0.5	0.75	0.95	Avg.	
BSN[27]	TS	✗	✓	✓	53.5	36.9	20.0	36.8	46.5	30.0	8.0	30.0	-
BMN[25]	TS	✗	✓	✓	56.0	38.8	20.5	38.5	50.1	34.8	8.3	33.9	-
BC-GNN[2]	TS	✗	✓	✓	57.1	40.4	23.1	40.2	50.6	34.8	**9.4**	34.3	-
BU-TAL[62]	I3D	✗	✓	✓	53.9	45.4	28.5	43.3	43.5	33.9	9.2	30.1	-
GTAN[33]	P3D	✗	✓	✓	57.8	38.8	-	-	52.6	34.1	8.9	34.3	-
G-TAD[53]	TS	✗	✓	✓	54.5	40.2	23.4	39.3	50.4	34.6	9.0	34.1	-
TAL [6]	I3D	✗	✓	✗	53.2	42.8	20.8	39.8	38.2	18.3	1.3	20.2	-
RTD-Action [41]	TS	✗	✓	✓	68.3	51.9	23.7	49.0	47.2	30.7	8.6	30.8	-
VSGN [60]	TS	✗	✓	✓	66.7	52.4	30.4	50.2	52.4	36.0	8.4	35.1	-
AFSD[23]	I3D	✓	✓	-	67.3	55.5	31.1	52.0	52.4	35.3	6.5	34.4	12
DaoTAD[45]	I3D	✓	✗	✗	62.8	53.8	30.1	50.0	-	-	-	-	11
DaoTAD[45]	SW	✓	✗	✗	72.7	59.8	33.3	56.3	-	-	-	-	30
TALLFormer-12	I3D	✓	✗	✗	68.4	57.6	30.8	53.9	41.3	27.3	6.3	27.2	12
TALLFormer-12	SW	✓	✗	✗	**76.1**	63.1	34.2	59.0	51.4	34.0	7.6	33.7	12
TALLFormer-32	SW	✓	✗	✗	76.0	**63.2**	**34.5**	**59.2**	**54.1**	**36.2**	7.9	**35.6**	29

classifier [52], which ensembles the predictions from ResNet-200 and Inception-V2 models operating on RGB and optical flow inputs. Instead, we simplify this pipeline by predicting action boundaries and categories using a single model. Not only is our proposed framework simpler and more efficient, but it also outperforms all previous approaches by 1.2% using RGB inputs alone. One interesting observation is that TALLFormer with I3D backbone is 6.5% lower than TALLFormer with a Swin-B backbone. Our analysis of this result reveals that TALLFormer-12 variant with an I3D backbone achieves a low video-level action recognition accuracy (78.2%) while the accuracy of TALLFormer-12 with Swin-B backbone is 90.1%. We also note that the accuracy of an external action recognition classifier [52] used by AFSD is 88.9%. This empirical finding also explains why all previous methods require an external action recognition classifier.

HACS. We train TALLFormer with Swin-B as backbone using the same network structure and hyperparameters as in AcitivityNet-1.3. We report these results in Tab. 4. These results suggest that similar to our previously considered datasets, TALLFormer also achieves state-of-the-art results on the HACS-Segment dataset. Specifically, it outperforms the GTAD [53] and BMN [25,36] baselines by 9.1% and 0.7% average mAP respectively without an external classifier.

Inference Speed Discussion. Compared with previous methods, we use a larger backbone and higher spatial resolution. On the other hand, our proposed

Table 4: Our results (in mAP) on the HACS-Segment dataset.

Backbone	mAP(%)			
	0.5	0.75	0.95	Avg.
GTAD [53]	-	-	-	27.5
BMN [25,36]	52.5	**36.4**	10.4	35.8
TALLFormer	**55.0**	36.1	**11.8**	**36.5**

Table 5: Ablation studies on THUMOS14: **(a)** TALLFormer works well even for a small sampling rate r; **(b)** The temporal consistency module leads to 1.5% boost in the average mAP; **(c)** TALLFormer performs better with longer temporal support. For **(a)** the backbone is Swin-B with spatial resolution 224×224. For **(b)** and **(c)**, the backbone is Swin-T with spatial resolution 112×112.

(a) Analysis of the clip sampling rate r.

r	mAP(%)	Mem
0.15	59.0	12
0.3	59.4	22
0.4	59.2	29
0.6	**60.0**	45
1.0	-	OOM

(b) Importance of Temporal Consistency Module (TCM).

TCM	mAP(%)
✗	51.1
✓	**52.6**

(c) Temporal Support (TS) analysis.

TS (sec)	mAP(%)
8	41.5
16	49.7
24	51.5
32	52.8
40	**53.3**

framework is much simpler than the frameworks of many prior TAL methods. In particular, we use a single model with only RGB frames as input while most previous methods adopt a two-stream approach that requires an external action classifier. This is costly, because (i) optical flow extraction is slow and because (ii) training and inference of an external action classifier is also time-consuming.

Due to the complexity of the existing systems, and the lack of publicly available implementations, it is difficult to quantitatively measure the inference speed of many prior methods. However, we note that in general, TALLFormer provides a much simpler, elegant and more efficient framework to the TAL problem. For example, consider performing inference on AcitivyNet-1.3 using a RTX A6000 GPU. The overall inference speed of a recent state-of-the-art AFSD [23] is 11.74s/video, which includes (i) optical flow extraction, (ii) processing two modalities and (iii) performing video-level action classification using [52]. On the other hand, TALLFormer only costs 1.58s/video while outperforming AFSD by 1.2% mAP.

4.5 Ablation Study

Lastly, we study various design choices of our TALLFormer model. Specifically, we investigate (i) TAL performance as a function of our clip sampling rate r,

(ii) the importance of the temporal consistency module (TCM) and (iii) TAL performance as a function of temporal support. We present these results below.

Accuracy vs. Clip Sampling Rate. During training, our model samples a fraction of r total short-term clips from an untrimmed video input. We study the performance as a function of clip sampling rate r. From Tab. 1(a), we can observe that i) GPU memory usage is proportional to the sampling rate r; ii) standard end-to-end training ($r = 1$) causes out-of-memory (OOM) error; iii) TALLFormer performs quite well even with a very small sampling rate 0.15, i.e., the TAL accuracy in mAP drops by only 1.0% while reducing the GPU memory usage to only 12 GB (compared to $>$ 45GB using $r = 1.0$)

Importance of Temporal Consistency Module (TCM). As shown in Tab. 5(b), our proposed TCM increases the average mAP by 1.5%, which indicates its importance to our overall framework. More conceptually, these results suggest that encouraging long-range interactions between the memory features, and the online-processed features can alleviate the feature distribution inconsistency issue, which is also suggested by the visualized features before and after TCM in Appendix. B.1.

Analysis of Temporal Support. We evaluate TALLFormer when using different temporal support (measured in seconds). Based on the results in Tab 5(c), we observe that longer temporal supports leads to consistently higher mAP.

5 Discussion

We present TALLFormer, a long-memory Transformer for temporal action localization. Our method is simple, flexible, and it can be efficiently trained on long high-resolution videos for TAL. Furthermore, we demonstrate that TALLFormer significantly outperforms previous TAL approaches on the THUMOS14 and ActivityNet-1.3 benchmarks.

Some readers might wonder whether optimizing the GPU memory usage for long-video processing is a valuable contribution since modern GPUs can accommodate larger and larger GPU memory requirements. Furthermore, there exist many prior memory saving techniques such as Gradient Checkpoint [7] and Mixed Precision [34]. Despite the advances in GPU hardware, and new developments in memory saving techniques, we believe that TALLFormer is still a valuable contribution to the research community. With the new developments in GPU hardware, the demands for higher resolution video analysis and larger models also grow. Thus, such demands pose new GPU memory-related challenges, especially for long-term video understanding tasks such as temporal action localization. We also note that TALLFormer can be easily combined with the existing memory-saving techniques, which we demonstrated in our experiments. Our future work involves extending our framework to various multimodal settings that involve processing both visual inputs and language.

References

1. Bagchi, A., Mahmood, J., Fernandes, D., Sarvadevabhatla, R.K.: Hear me out: Fusional approaches for audio augmented temporal action localization. arXiv preprint arXiv:2106.14118 (2021)
2. Bai, Y., Wang, Y., Tong, Y., Yang, Y., Liu, Q., Liu, J.: Boundary content graph neural network for temporal action proposal generation. In: European Conference on Computer Vision. pp. 121–137. Springer (2020)
3. Bertasius, G., Wang, H., Torresani, L.: Is space-time attention all you need for video understanding. arXiv preprint arXiv:2102.05095 **2**(3), 4 (2021)
4. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of the ieee conference on computer vision and pattern recognition. pp. 961–970 (2015)
5. Carreira, J., Zisserman, A.: Quo vadis, action recognition? a new model and the kinetics dataset. In: proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6299–6308 (2017)
6. Chao, Y.W., Vijayanarasimhan, S., Seybold, B., Ross, D.A., Deng, J., Sukthankar, R.: Rethinking the faster r-cnn architecture for temporal action localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1130–1139 (2018)
7. Chen, T., Xu, B., Zhang, C., Guestrin, C.: Training deep nets with sublinear memory cost. arXiv:1604.06174 (2016)
8. Cheng, F., Xu, M., Xiong, Y., Chen, H., Li, X., Li, W., Xia, W.: Stochastic backpropagation: A memory efficient strategy for training video models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8301–8310 (2022)
9. Choromanski, K., Likhosherstov, V., Dohan, D., Song, X., Gane, A., Sarlos, T., Hawkins, P., Davis, J., Mohiuddin, A., Kaiser, L., et al.: Rethinking attention with performers. arXiv preprint arXiv:2009.14794 (2020)
10. Fan, H., Xiong, B., Mangalam, K., Li, Y., Yan, Z., Malik, J., Feichtenhofer, C.: Multiscale vision transformers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6824–6835 (2021)
11. Feichtenhofer, C.: X3d: Expanding architectures for efficient video recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 203–213 (2020)
12. Feichtenhofer, C., Fan, H., Malik, J., He, K.: Slowfast networks for video recognition. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 6202–6211 (2019)
13. Gao, J., Shi, Z., Wang, G., Li, J., Yuan, Y., Ge, S., Zhou, X.: Accurate temporal action proposal generation with relation-aware pyramid network. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 10810–10817 (2020)
14. Gao, J., Chen, K., Nevatia, R.: Ctap: Complementary temporal action proposal generation. In: Proceedings of the European conference on computer vision (ECCV). pp. 68–83 (2018)
15. Goyal, R., Ebrahimi Kahou, S., Michalski, V., Materzynska, J., Westphal, S., Kim, H., Haenel, V., Fruend, I., Yianilos, P., Mueller-Freitag, M., et al.: The" something something" video database for learning and evaluating visual common sense. In: Proceedings of the IEEE international conference on computer vision. pp. 5842–5850 (2017)

16. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9729–9738 (2020)
17. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
18. Hendrycks, D., Gimpel, K.: Gaussian error linear units (gelus). arXiv preprint arXiv:1606.08415 (2016)
19. Huang, G., Sun, Y., Liu, Z., Sedra, D., Weinberger, K.Q.: Deep networks with stochastic depth. In: European conference on computer vision. pp. 646–661. Springer (2016)
20. Idrees, H., Zamir, A.R., Jiang, Y.G., Gorban, A., Laptev, I., Sukthankar, R., Shah, M.: The thumos challenge on action recognition for videos "in the wild". Computer Vision and Image Understanding **155**, 1–23 (2017)
21. Jiang, B., Wang, M., Gan, W., Wu, W., Yan, J.: Stm: Spatiotemporal and motion encoding for action recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2000–2009 (2019)
22. Kwon, H., Kim, M., Kwak, S., Cho, M.: Motionsqueeze: Neural motion feature learning for video understanding. In: European Conference on Computer Vision. pp. 345–362. Springer (2020)
23. Lin, C., Xu, C., Luo, D., Wang, Y., Tai, Y., Wang, C., Li, J., Huang, F., Fu, Y.: Learning salient boundary feature for anchor-free temporal action localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3320–3329 (2021)
24. Lin, J., Gan, C., Han, S.: Tsm: Temporal shift module for efficient video understanding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7083–7093 (2019)
25. Lin, T., Liu, X., Li, X., Ding, E., Wen, S.: Bmn: Boundary-matching network for temporal action proposal generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3889–3898 (2019)
26. Lin, T., Zhao, X., Shou, Z.: Single shot temporal action detection. In: Proceedings of the 25th ACM international conference on Multimedia. pp. 988–996 (2017)
27. Lin, T., Zhao, X., Su, H., Wang, C., Yang, M.: Bsn: Boundary sensitive network for temporal action proposal generation. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)
28. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
29. Liu, Q., Wang, Z.: Progressive boundary refinement network for temporal action detection. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 11612–11619 (2020)
30. Liu, Y., Ma, L., Zhang, Y., Liu, W., Chang, S.F.: Multi-granularity generator for temporal action proposal. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3604–3613 (2019)
31. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10012–10022 (2021)
32. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv preprint arXiv:2106.13230 (2021)

33. Long, F., Yao, T., Qiu, Z., Tian, X., Luo, J., Mei, T.: Gaussian temporal awareness networks for action localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 344–353 (2019)
34. Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., et al.: Mixed precision training. arXiv preprint arXiv:1710.03740 (2017)
35. Patrick, M., Campbell, D., Asano, Y., Misra, I., Metze, F., Feichtenhofer, C., Vedaldi, A., Henriques, J.F.: Keeping your eye on the ball: Trajectory attention in video transformers. Advances in Neural Information Processing Systems **34** (2021)
36. Qing, Z., Su, H., Gan, W., Wang, D., Wu, W., Wang, X., Qiao, Y., Yan, J., Gao, C., Sang, N.: Temporal context aggregation network for temporal action proposal refinement. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 485–494 (2021)
37. Qiu, Z., Yao, T., Mei, T.: Learning spatio-temporal representation with pseudo-3d residual networks. In: proceedings of the IEEE International Conference on Computer Vision. pp. 5533–5541 (2017)
38. Shou, Z., Chan, J., Zareian, A., Miyazawa, K., Chang, S.F.: Cdc: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5734–5743 (2017)
39. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. Advances in neural information processing systems **27** (2014)
40. Su, H., Gan, W., Wu, W., Qiao, Y., Yan, J.: Bsn++: Complementary boundary regressor with scale-balanced relation modeling for temporal action proposal generation. arXiv preprint arXiv:2009.07641 (2020)
41. Tan, J., Tang, J., Wang, L., Wu, G.: Relaxed transformer decoders for direct action proposal generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13526–13535 (2021)
42. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M.: Learning spatiotemporal features with 3d convolutional networks. In: Proceedings of the IEEE international conference on computer vision. pp. 4489–4497 (2015)
43. Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., Paluri, M.: A closer look at spatiotemporal convolutions for action recognition. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 6450–6459 (2018)
44. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
45. Wang, C., Cai, H., Zou, Y., Xiong, Y.: Rgb stream is enough for temporal action detection. arXiv preprint arXiv:2107.04362 (2021)
46. Wang, L., Li, W., Li, W., Van Gool, L.: Appearance-and-relation networks for video classification. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1430–1439 (2018)
47. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Gool, L.V.: Temporal segment networks: Towards good practices for deep action recognition. In: European conference on computer vision. pp. 20–36. Springer (2016)
48. Wang, L., Xiong, Y., Wang, Z., Qiao, Y., Lin, D., Tang, X., Van Gool, L.: Temporal segment networks for action recognition in videos. IEEE transactions on pattern analysis and machine intelligence **41**(11), 2740–2755 (2018)
49. Wang, S., Li, B.Z., Khabsa, M., Fang, H., Ma, H.: Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768 (2020)

50. Wang, X., Gao, C., Zhang, S., Sang, N.: Multi-level temporal pyramid network for action detection. In: Chinese Conference on Pattern Recognition and Computer Vision (PRCV). pp. 41–54. Springer (2020)
51. Wu, C.Y., Feichtenhofer, C., Fan, H., He, K., Krahenbuhl, P., Girshick, R.: Long-term feature banks for detailed video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 284–293 (2019)
52. Xiong, Y., Wang, L., Wang, Z., Zhang, B., Song, H., Li, W., Lin, D., Qiao, Y., Van Gool, L., Tang, X.: Cuhk & ethz & siat submission to activitynet challenge 2016. arXiv preprint arXiv:1608.00797 (2016)
53. Xu, M., Zhao, C., Rojas, D.S., Thabet, A., Ghanem, B.: G-tad: Sub-graph localization for temporal action detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10156–10165 (2020)
54. Xu, M., Xiong, Y., Chen, H., Li, X., Xia, W., Tu, Z., Soatto, S.: Long short-term transformer for online action detection. Advances in Neural Information Processing Systems **34**, 1086–1099 (2021)
55. You, C., Han, L., Feng, A., Zhao, R., Tang, H., Fan, W.: Megan: Memory enhanced graph attention network for space-time video super-resolution. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1401–1411 (2022)
56. You, C., Zhao, R., Liu, F., Chinchali, S., Topcu, U., Staib, L., Duncan, J.S.: Class-aware generative adversarial transformers for medical image segmentation. arXiv preprint arXiv:2201.10737 (2022)
57. Zeng, R., Huang, W., Tan, M., Rong, Y., Zhao, P., Huang, J., Gan, C.: Graph convolutional networks for temporal action localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7094–7103 (2019)
58. Zhang, C., Gupta, A., Zisserman, A.: Temporal query networks for fine-grained video understanding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4486–4496 (2021)
59. Zhang, D., Dai, X., Wang, X., Wang, Y.F.: S3d: single shot multi-span detector via fully 3d convolutional networks. arXiv preprint arXiv:1807.08069 (2018)
60. Zhao, C., Thabet, A.K., Ghanem, B.: Video self-stitching graph network for temporal action localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13658–13667 (2021)
61. Zhao, H., Torralba, A., Torresani, L., Yan, Z.: Hacs: Human action clips and segments dataset for recognition and temporal localization. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8668–8678 (2019)
62. Zhao, P., Xie, L., Ju, C., Zhang, Y., Wang, Y., Tian, Q.: Bottom-up temporal action localization with mutual regularization. In: European Conference on Computer Vision. pp. 539–555. Springer (2020)
63. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Tang, X., Lin, D.: Temporal action detection with structured segment networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2914–2923 (2017)

A Implementation Details

Here, we provide more details related to the (i) long memory module and (ii) temporal boundary localization module of our TallFormer model.

Algorithm 1 Pseudocode of short-term feature extraction and feature sampling from long-term memory.

```
# encoder: short-term Transformer encoder.
# clips: video clips (N_c x L_c x H x W x 3).
# long_memory: the pre-extracted features for this video (N_c x L_f x
    C_f).
# r: sampling rate (float).

# sample clips processed by encoder
sampled_idx = uniform_sample(N_c, r)
remaining_idx = [idx for idx in range(N_c) if idx not in sampled_idx]
sampled_clips = clips[sampled_idx]

# Short-term Transformer Encoder
sampled_features = encoder.forward(sampled_clips) # shape: [N_s, L_f,
    C_f]

# Long-term Memory Module
mem_features = long_memory[remaining_idx]. # shape: [N_c-N_s, L_f,
    C_f]
long_memory[sampled_idx] = sampled_features.detach()

# Temporal Consistent Module
## gather features
features = zeros(N_c,*sampled_features.shape[1:])
features[sampled_idx] = sampled_features
features[remaining_idx] = mem_features
features = features.reshape(N_c*L_f, C_f) #shape: [N_c*L_f, C_f]
## refine features
for i in range(L):
    features = TransformerLayer(features) #shape: [N_c*L_f, C_f]
```

A.1 Long Memory Module

The implementation details of the proposed Long Memory Module (LMM) and Temporal Consistency Module (TCM) are as shown in Alg. 1. The inputs are first participate into N_c clips. Among these clips, we sample N_s clips to be processed by the Short-term Transformer Encoder and the remaining $N_c - N_s$ clips by LMM. The clip features extracted by the encoder is also used to update the LMM. All the clips features are fed to the TCM to generate more consistent features. The output features of TCM are the input to the temporal boundary localization module.

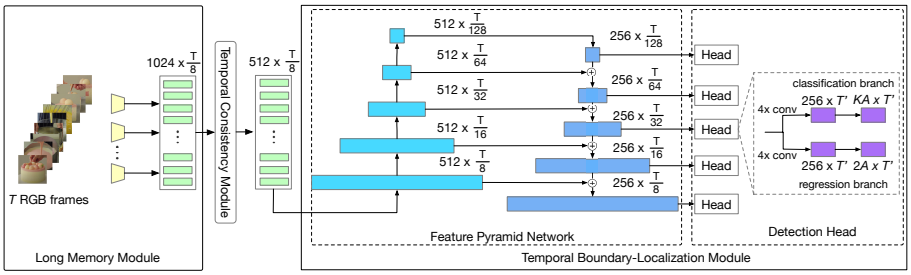

Fig. 3: Network structure for THUMOS14.

A.2 Temporal Boundary-Localization Module

Given the refined features $f_r \in \mathbb{R}^{C_c \times L}$, the Temporal Boundary-Localization Module (TBLM) aims to produce the action boundaries and categories for each action instance. We use different TBLMs for THUMOS14 [20] and ActivityNet [4].

THUMOS14. The detailed architecture is shown in Fig. 3. The TBLM is composed of a Feature-Pyramid Network (FPN) and a Detection Head. The Detection Head is taken from DaoTAD [45]. In the FPN, the features are downsampled (bottom to up) using 1D kernel-3, stride-2 convolutions and are upsampled (up to bottom) by linear interpolation along the temporal dimension. We use Focal loss [28] for the sigmoid-activated classification branch and DIoU loss [**?**] for the regression branch. The weights are 1 for both losses. We refer readers to [45] for more details.

ActivityNet-1.3. The detailed architecture is shown in Fig. 4. We use the same Long Memory Module, Temporal Consistency Module, Feature Pyramid Network as in THUMOS14. We adopt the Detection Head design from AFSD [23]. Additionally, after the Temporal Consistency Module, we also add a video-level classifier composed of a global average pooling layer, dropout layer with drop-rate 0.5 and a linear layer with a dimensionality equal to the number of action classes. AFSD Detection Head is a two-stage detector. First, it uses a Basic Prediction Module to predict the coarse action boundaries and action-agnostic classes (background or not). Then a Saliency-based Refinement Module is used to refine the predicted boundaries and action-agnostic classes. Finally, we assign each predicted action proposal with the action category predicted by the video-level classifier. We use Cross-entropy loss for video-level classifier, Focal loss [28] for classification branches in the detection head, tIoU loss [23] for the regression of Basic Prediction Module and L1 loss for the boundary refinement in the Saliency-based Prediction Module. The weights are 1 for all the losses. We refer the readers to [23] for more specific details related to the Detection Head.

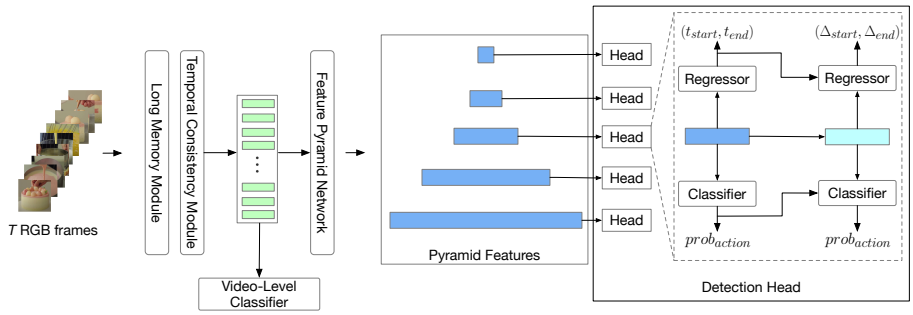

Fig. 4: Network structure for ActivityNet-1.3. The same Long Memory Module, Temporal Consistency Module and Feature Pyramid Network are used as in THUMOS14.

B Additional Results

B.1 Importance of Temporal Consistency Module

In addition to the quantitative results in the main paper, we visualize the features before and after Temporal Consistency Module (TCM) as in Fig. we extracted four sets of features: features from short-term feature extractor (1) before, and (2) after the TCM, and features from the Long Memory Module (3) before, and (4) after the TCM. We then applied PCA and plotted the first two principal components as shown Fig. 5. We observe that the features from the short-term feature extractor and long-term memory are more similar after the TCM than they were before the TCM. This suggests that TCM effectively reduces the inconsistency between features from the short-term feature extractor and long memory module.

Table 6: Ablating different short-term transformer encoders within our TALLFormer framework on THUMOS14 [20].

Transformer Encoder	mAP(%)					
	0.3	0.4	0.5	0.6	0.7	Avg.
Swin-T [32]	72.7	69.0	60.8	48.3	34.3	57.0
Swin-S [32]	74.9	70.3	62.1	48.9	34.3	58.1
Swin-B [32]	76.0	71.5	63.2	50.9	34.5	59.2

B.2 Ablating Different Short-term Transformer Encoders

The flexibility of our TALLFormer model allows us to use any short-term transformer encoder as our clip-level backbone. To demonstrate TALLFormer's generalization with different backbones, we experiment with different variations of

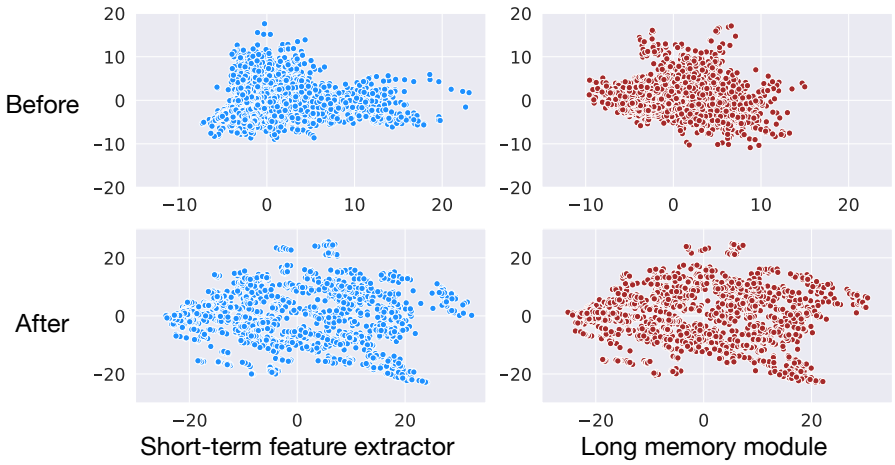

Fig. 5: Network structure for THUMOS14.

Swin Transformers [32], i.e. Swin-tiny, Swin-small and Swin-base. As shown in Tab. 6, TALLFormer achieves pretty high average mAPs on all the backbones.

B.3 Ablating Temporal Support

Due to long actions (e.g., 30 seconds in length), our model needs to span long temporal extent. Thus, here, we evaluate TALLFormerwhen using different temporal support (measured in seconds). Based on the results in Tab 7, we observe that longer temporal supports leads to consistently higher average mAP.

Table 7: Temporal Support (TS) ablation on THUMOS14 [20]. The model is TALLFormer [45] with Swin-T as backbone and spatial resolution 112×112. We observe that longer temporal supports leads to higher average mAP.

TS (sec)	mAP(%)					
	0.3	0.4	0.5	0.6	0.7	Avg.
8	59.8	54.1	45.4	31.3	17.0	41.5
16	65.0	60.3	52.5	42.6	28.3	49.7
24	65.7	62.0	53.8	45.0	31.0	51.5
32	66.9	63.2	**56.7**	**46.0**	31.1	52.8
40	**68.0**	**63.7**	56.2	**46.0**	**32.6**	**53.3**