Supplementary Material for 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

Our supplementary material consists of:

1. Implementation Details.
2. Additional Quantitative Results.

1 Implementation Details

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

1.1 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.

1.2 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 [2] and ActivityNet [1].

THUMOS14. The detailed architecture is shown in Fig. 1. The TBLM is composed of a Feature-Pyramid Network (FPN) and a Detection Head. The Detection Head is taken from DaoTAD [6]. 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 [4] for the sigmoid-activated classification branch and DIoU loss [7] for the regression branch. The weights are 1 for both losses. We refer readers to [6] for more details.

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]
```

ActivityNet-1.3. The detailed architecture is shown in Fig. 2. We use the same Long Memory Module, Temporal Consistency Module, Feature Pyramid Network as in THUMOS14. We adopt the Detection Head design from AFSD [3]. 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 [4] for classification branches in the detection head, tIoU loss [3] for the

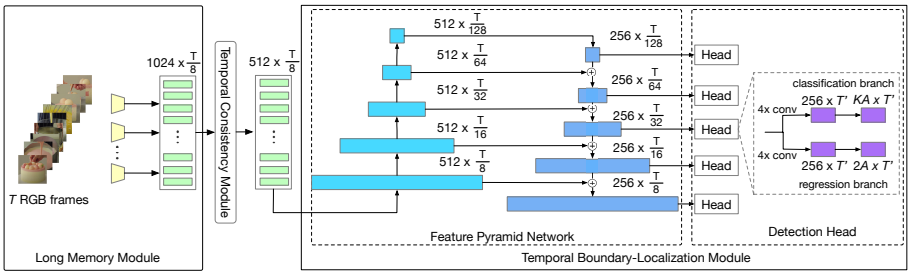

Fig. 1: Network structure for THUMOS14.

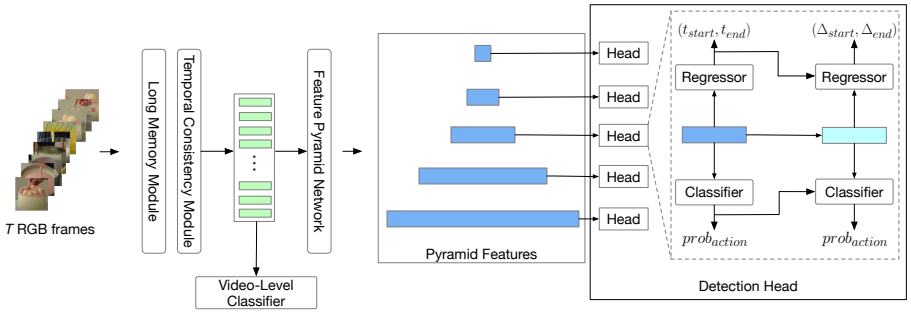

Fig. 2: Network structure for ActivityNet-1.3. The same Long Memory Module, Temporal Consistency Module and Feature Pyramid Network are used as in THUMOS14.

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 [3] for more specific details related to the Detection Head.

2 Additional Results

2.1 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. 3. 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.

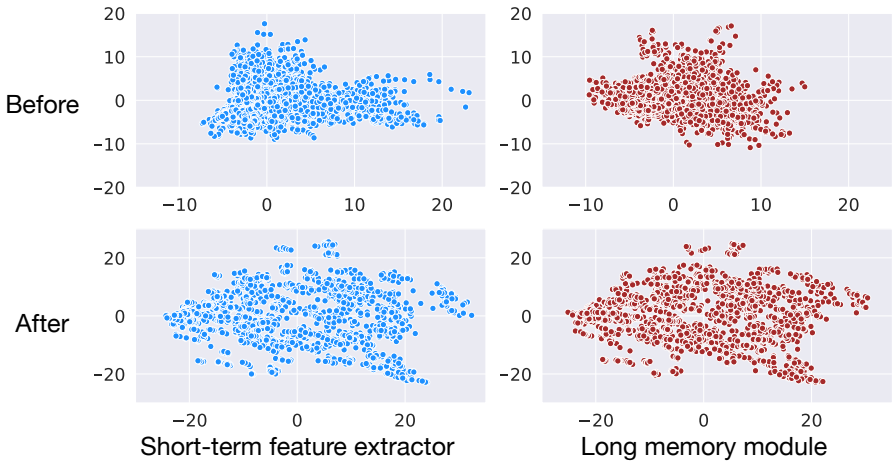

Fig. 3: Network structure for THUMOS14.

Table 1: Ablating different short-term transformer encoders within our TALLFormer framework on THUMOS14 [2].

Transformer Encoder	mAP(%)					
	0.3	0.4	0.5	0.6	0.7	Avg.
Swin-T [5]	72.7	69.0	60.8	48.3	34.3	57.0
Swin-S [5]	74.9	70.3	62.1	48.9	34.3	58.1
Swin-B [5]	76.0	71.5	63.2	50.9	34.5	59.2

2.2 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 Swin Transformers [5], i.e. Swin-tiny, Swin-small and Swin-base. As shown in Tab. 1, TALLFormer achieves pretty high average mAPs on all the backbones.

2.3 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 2, we observe that longer temporal supports leads to consistently higher average mAP.

References

1. Caba Heilbron, F., Escorcia, V., Ghanem, B., Carlos Niebles, J.: Activitynet: A large-scale video benchmark for human activity understanding. In: Proceedings of

Table 2: Temporal Support (TS) ablation on THUMOS14 [2]. The model is TALLFormer [6] 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**

the ieee conference on computer vision and pattern recognition. pp. 961–970 (2015) 1

2. 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) 1, 4, 5
3. 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) 2, 3
4. 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) 1, 2
5. Liu, Z., Ning, J., Cao, Y., Wei, Y., Zhang, Z., Lin, S., Hu, H.: Video swin transformer. arXiv preprint arXiv:2106.13230 (2021) 4
6. Wang, C., Cai, H., Zou, Y., Xiong, Y.: Rgb stream is enough for temporal action detection. arXiv preprint arXiv:2107.04362 (2021) 1, 5
7. Zheng, Z., Wang, P., Liu, W., Li, J., Ye, R., Ren, D.: Distance-iou loss: Faster and better learning for bounding box regression. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 12993–13000 (2020) 1