End-to-End Temporal Action Detection with 1B Parameters Across 1000 Frames [Supplementary Material]

Shuming Liu[1] Chen-Lin Zhang[2] Chen Zhao[1*] Bernard Ghanem[1]
[1]King Abdullah University of Science and Technology (KAUST) [2]4Paradigm Inc

In this appendix, we provide more details of our method and present more experiment results. Specifically, we give a detailed illustration of AdaTAD† in Section A. Then, we present the implementation details between different datasets in Section B. Next, we provide our results on the Ego4D dataset in Section C. After this, we show additional experiments and further analysis in Section D. Then, the error analysis is conducted in Section E, and qualitative visualization is demonstrated in Section F. Finally, we discuss the limitations of our work in Section G.

A. Further illustration of AdaTAD†

To reduce the memory usage and further scale up the model and data, AdaTAD† proposes an alternative placement for the temporal-informative adapters (TIAs). The entire pipeline of AdaTAD† is shown in Fig. 1.

Concretely speaking, we retain the TIA architecture, but eliminate the last residual connection in AdaTAD, as illustrated in Equation 1. Therefore, given layer i 's output $\boldsymbol{x_i}$, the corresponding TIA's output $\boldsymbol{x'_i}$ will be 0 at the start of the training, since the weights and biases of $\boldsymbol{W}_{\text{up}}$ are initialized to 0.

$$\begin{aligned} \boldsymbol{x'_i} &= \boldsymbol{\sigma}(\boldsymbol{W}^{\top}_{\text{down}} \cdot \boldsymbol{x_i}), \\ \boldsymbol{x'_i} &= \boldsymbol{W}^{\top}_{mid} \cdot \text{DWConv}_k(\boldsymbol{x'_i}) + \boldsymbol{x'_i}, \\ \boldsymbol{x'_i} &= \alpha \cdot \boldsymbol{W}^{\top}_{\text{up}} \cdot \boldsymbol{x'_i}. \end{aligned} \tag{1}$$

What's more, in the above equation, $\boldsymbol{x'_i}$ is not added into $\boldsymbol{x_i}$, which is different from AdaTAD. This means that the adapter's outputs do not contribute to the middle activations of the original backbone. Instead, it serves as a *residual* and is directly added to the backbone's final output $\boldsymbol{x_N}$, where N is the total number of layers in the backbone. Consequently, the output of the video encoder becomes $\boldsymbol{y}$, as depicted in Equation 2. At the beginning of the fine-tuning, $\boldsymbol{y}$ is initialized as $\boldsymbol{x_N}$, and is gradually augmented with the aforementioned *residuals* to adapt the fine-tuning, which is driven by the updated TIAs.

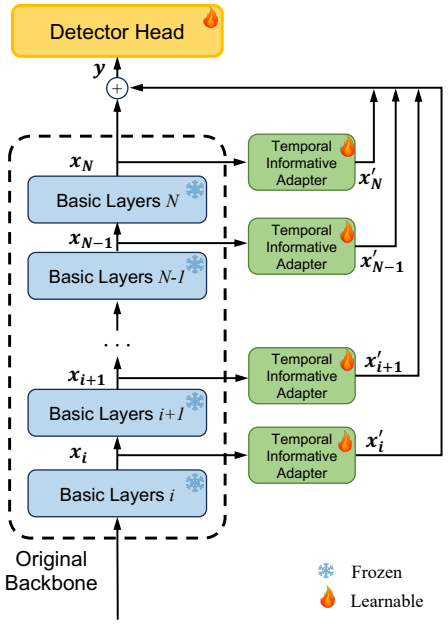


Figure 1. **Detailed architecture of AdaTAD†**. The output of each adapter will be added to the final output of the original backbone.

$$\boldsymbol{y} = \boldsymbol{x_N} + \sum_{i=1}^{N} \boldsymbol{x'_i}. \tag{2}$$

In our design, the key to memory reduction lies in **stopping the gradient backpropagation for the original backbone**. Since the TIA's output directly goes to the final output, gradients during backpropagation do not trace back to the original backbone but are confined to the shallow and lightweight adapters. This architecture can be viewed as the *Side Network* [20] or *Ladder Network* [24], meaning that it creates a light network apart from the original heavy backbone. Compared to the traditional PEFT approach or our AdaTAD, this design requires only a few activations from the backbone, and it can further refine these activations to adapt the downstream task during transfer learning.

*Corresponding author.

Table 1. **End-to-End setting in different TAD datasets.**

Config	ActivityNet-1.3 [11]	THUMOS14 [13]	EPIC-Kitchens 100 [7]	Ego4D-MQ [10]
Backbone Setting				
Video Preprocessing	Resize	Sliding Window	Sliding Window	Padding
Frame Stride	-	4	2	2
Frame Number	192×4	768	768×8	900×8
Frame Resolution	160×160			
Data Augmentation	RandomResizedCrop + Flip + ImgAug + ColorJitter			
Feature Post Processing	Spatial Average Pooling + Resize			
Feature Resize Length	192	768	768	900
Detector Setting				
Warmup Epoch	5	5	5	5
Total Epoch	15	60	35	15
Batch Size	16	2	2	2

B. Implementation Details

Training Details. Unless otherwise specified, we prefer AdaTAD for its high performance. By default, mixed precision training [16] and activation checkpointing [4] are adopted following previous work [6]. We also utilize flash attention [8] to accelerate the computation and save the memory in the VideoMAE-family models. Our method is evaluated on four datasets, with the hyper-parameters detailed in Table 1.

Data Preprocessing. Regarding the data preprocessing, we provide the following instructions. For ActivityNet, due to the variable action duration, we resize videos into fixed lengths with 768 frames. After the video encoder, average pooling is adopted along the spatial dimension, and the feature's temporal length is resized to 192. For THUMOS14 and EPIC-Kitchens, given their longer video lengths, we apply the random truncation to a fixed-length window during training and use sliding window during testing. For THUMOS14, the window length is 768 frames with a stride of 4. For EPIC-Kitchens, it's 768×8 frames with a stride of 2, and additionally, the feature after the video encoder is temporally resized to 768. On the Ego4D-MQ dataset, since all the videos are under 8 minutes, we sample 900×8 frames with a stride of 2, and resize the feature to length 900.

Video Model. All the action recognition models used in our paper, such as SlowFast [9], VideoSwin [15], and VideoMAE [21], are pretrained on Kinetics-400 [14], except the VideoMAEv2-giant [22] that is hybrid pretrained and fine-tuned on Kinetics-710 [2]. Meanwhile, for the ViT-based model, to avoid excessive temporal attention for the untrimmed video, we chunk the video into shorter clips before processing it in the ViT block. For example, given the input video with 768 frames, we reshape the 768 frames as 16×48, dividing it into 48 snippets with 16 frames each. Then VideoMAE processes these snippets independently, but before sending them into our proposed adapter, we reassemble the snippets into the original 768 frames to achieve cross-snippet information exchange. This approach, focusing temporal attention only on 16 frames, is specific to ViT-based models. CNN-based models like SlowFast and local attention-based models like VideoSwin do not require this reshaping.

Memory Usage on EPIC-Kitchens and Ego4D-MQ. On both datasets, we use the VideoMAE-Large as the backbone, but differently pretrained on respective datasets. On EPIC-Kitchens, the memory usage is 48.5GB per video for $768 \times 8 = 6144$ frames. On Ego4D-MQ, the memory usage is 60.0GB per video for $900 \times 8 = 7200$ frames.

C. Results on Ego4D-Moment Queries

We present our results on the Ego4D-MQ dataset in Table 2. The backbone is VideoMAE-Large, which is fine-tuned by InternVideo on Ego4D classification task [3]. Note that previous methods like VSGN [26] and ActionFormer [25] use the same backbone, yet they extract high-resolution, densely sampled offline features. Our work first establishes a stronger baseline by adopting an improved version of ActionFormer on this dataset [19], achieving 27.07% mAP. Furthermore, utilizing end-to-end training with our proposed temporal-informative adapter elevates the performance to 28.08% mAP.

More importantly, when we apply the full fine-tuning on Ego4d-MQ, no performance gain was observed (27.01% mAP). This implies that the pretrained model is sufficiently robust, and the downstream data is too limited for effective transfer learning. Nonetheless, our AdaTAD manages to yield a performance increase of +1.01%. This outcome further demonstrates the efficacy of our adapter-based transfer learning.

Table 2. **Results on the *validation* set of Ego4D-Moment Queries v2.0.** We report *m*AP at different tIoU thresholds. InternVideo [3] denotes the backbone is VideoMAE-L [21], which is pretrained and fine-tuned on Ego4D-Moment Queries.

Method	Feature	E2E	0.1	0.2	0.3	0.4	0.5	Avg.
VSGN [26]	EgoVLP	✗	16.63	-	11.45	-	6.57	11.39
VSGN [26]	InternVideo	✗	-	-	-	-	-	19.35
ActionFormer [25]	EgoVLP	✗	26.84	-	20.57	-	14.54	20.60
ActionFormer [25]	InternVideo	✗	-	-	-	-	-	23.29
ASL [17]	EgoVLP	✗	29.45	-	23.03	-	16.08	22.83
AdaTAD	InternVideo	✗	32.40	29.50	26.98	24.43	21.88	27.07
AdaTAD	**InternVideo**	✓	**33.53**	**30.71**	**28.04**	**25.51**	**22.59**	**28.08**

D. Additional Experiments

In this section, we provide additional experiments to study the effectiveness of our proposed method. These experiments are omitted from the main paper due to lack of space.

D.1. More Results of AdaTAD†

In our paper, we propose two distinct adapter placement designs: AdaTAD and AdaTAD†. The former directly inserts the adapters into the original backbone, while the latter positions the adapter outside the backbone. This external placement in AdaTAD† stops gradient backpropagation to the original backbone, negating the need to save massive intermediate activations and thus reducing memory usage. However, previous research [20] suggests that such a design might comprise the performance, as the lighter adapters may limit the representation capacity during transfer learning. Therefore, to verify this hypothesis and explore the advantages of AdaTAD†, we conduct experiments summarized in Table 3, leading to two key conclusions:

1. **With identical input data, AdaTAD† underperforms to AdaTAD.** For example, when using VideoMAE-Base with 768 frames and a resolution of 160^2, the performance of AdaTAD† drops from 71.5% to 70.2%. This verifies that the transfer learning ability of AdaTAD is better than AdaTAD†.
2. **Under a similar memory budget, AdaTAD† can achieve comparable performance than AdaTAD by scaling up the input data.** Owing to its lower memory usage, AdaTAD† allows for larger input data, thereby improving performance. For instance, AdaTAD† with inputs of 768, 224^2 can marginally outperform AdaTAD with inputs of 768, 160^2 under the same memory budget.

These experiments underscore the significance of data scaling. Particularly with larger models like VideoMAEv2-giant, AdaTAD has reached a limit of scaling up the input data. In such scenarios, only AdaTAD† can manage a similar memory budget while enhancing input data for superior performance. In conclusion, AdaTAD† is tailored for higher detection performance in situations where the backbone model is very large and cannot accommodate more frames or higher image resolution. **In scenarios where these are viable, AdaTAD remains the recommended choice for optimal performance.** Therefore, except in the case of giant models, we default to using AdaTAD.

Table 3. **The advantage of AdaTAD† lies in scaling up the data with low memory usage.** When using the same input, the performance of AdaTAD† is inferior to AdaTAD. However, under a similar memory budget, AdaTAD† can achieve comparable or better performance thanks to data scaling. We report the memory usage and average mAP on the THUMOS14 dataset.

Model	Method	Input	Mem.	mAP
VideoMAE-S	AdaTAD	768, 160^2	2.5G	68.8
	AdaTAD†	768, 160^2	1.8G	68.0
	AdaTAD†	768, 224^2	**2.7G**	**68.9**
VideoMAE-B	AdaTAD	768, 160^2	4.9G	71.5
	AdaTAD†	768, 160^2	4.0G	70.2
	AdaTAD†	768, 224^2	**4.9G**	**71.9**
VideoMAE-L	AdaTAD	768, 160^2	11.0G	73.5
	AdaTAD†	768, 160^2	8.1G	73.1
	AdaTAD†	768, 224^2	**10.8G**	**73.7**

D.2. More Results with Swin and SlowFast

To validate the efficacy of our adapter tuning approach, we expand our study to include a broader range of backbone models. This extended analysis, detailed in Table 4, encompasses not only the window-based transformer model, *i.e.*, VideoSwin [15], but also the 3D CNN model, *i.e.*, SlowFast [9]. The findings are in line with those reported in the main paper, indicating that adapter tuning can yield better detection performance compared to full fine-tuning. Notably, the performance gains are even more pronounced with VideoSwin and SlowFast than with VideoMAE. Specifically, our method enhanced detection performance from 55.1% to 63.7% with VideoSwin-B, and from 62.3% to 66.1% with SlowFast-R50.

Table 4. **Compared to full fine-tuning, our adapter tuning can achieve better performance with less memory.** Param. is the number of tunable parameters in the backbone. $*$ means out of memory on A100-80GB, and we report the estimated number. We conduct the following experiments on THUMOS14 dataset.

Model	Setting	E2E	Param.	Mem.	mAP
VideoSwin-B	Feature	✗	0	-	55.1
	Snippet Full FT	✓	87.6M	213G*	-
	Frame Full FT	✓	87.6M	16.4G	60.4
	AdaTAD	✓	3.9M	16.1G	**63.7**
SlowFast-R50	Feature	✗	0	-	62.3
	Snippet Full FT	✓	33.6M	36.9G	66.1
	Frame Full FT	✓	33.6M	3.9G	64.3
	AdaTAD	✓	11.4M	4.3G	**66.0**

D.3. Study of Different Kernel Size in TIA

In our temporal-informative adapter (TIA), we employ depth-wise convolution along the temporal dimension to capture context from adjacent frames. This design utilizes a 3D depth-wise convolution layer with kernel size (t, h, w). By default, the kernel is set to $(3, 1, 1)$. To assess the influence of various kernel sizes in TIA, we conduct a study summarized in Table 5.

First, we expand the kernel size to $(3, 3, 3)$, and note a decrease in performance. This suggests that the spatial context has been effectively handled by the original backbone, and additional spatial processing could potentially disrupt the pretrained knowledge. Subsequently, reducing the kernel size to $(1, 1, 1)$ results in inferior performance compared to ours, likely due to insufficient temporal information aggregation. Moreover, we gradually increase the temporal kernel size from 3 to 7, 13, and 21, observing a consistent downward trend in performance. This indicates that aggregating a longer-range temporal context does not necessarily benefit the backbone. Overall, our default TIA configuration demonstrates the best performance.

Table 5. **Ablation of different kernel size in depth-wise convlution in AdaTAD.** The order of the kernel size follows t, h, w. VideoMAE-B is used as the backbone on THUMOS dataset.

DW Kernel	0.3	0.5	0.7	mAP
(3,1,1)	87.04	75.33	49.22	**71.56**
(1,1,1)	85.97	74.61	49.12	71.06
(3,3,3)	85.46	73.74	49.13	70.63
(7,1,1)	86.17	74.62	48.93	71.02
(13,1,1)	85.68	72.80	47.98	69.96
(21,1,1)	84.75	72.53	46.74	69.03

D.4. Ablation of Adapter Design on ActivityNet-1.3

In the main paper, we present a comparison of various adapter architectural designs on THUMOS14. Extending our analysis, we conduct a similar ablation study on ActivityNet-1.3, with results detailed in Table 6. The findings from this study align with the conclusions drawn in the main paper. Notably, since the size of ActivityNet is much larger than THUMOS14, the competition in performance metrics is more intense, resulting in smaller gains. Despite this, our AdaTAD still stands out among other adapter architectural designs. Compared to using frozen features, AdaTAD enhances performance from 36.64% to 38.39%, affirming its efficacy in a more challenging dataset.

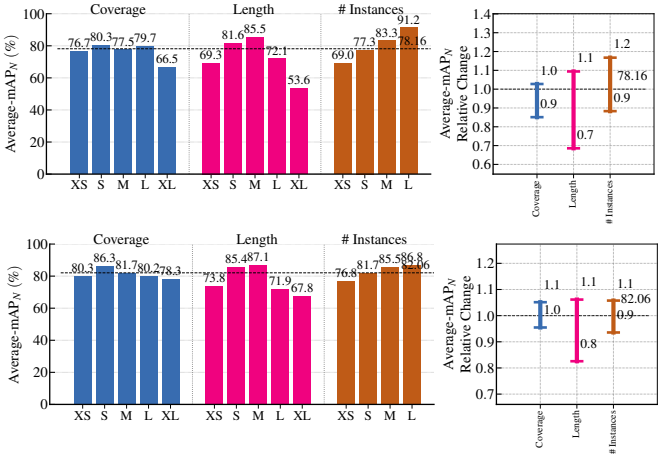
Moreover, we also combine our proposed TIA modules with full fine-tuning strategy, but observe a decreased mAP. This is due to the learning rate conflicts between the pretrained backbone and the newly added adapter. The former prefers a smaller learning rate since the model is pretrained on large datasets, while the latter prefers a larger learning rate since it is newly added and randomly initialized.

Table 6. **Ablation of different adapter architectural designs on ActivityNet-1.3.** VideoMAE-B is used as the backbone.

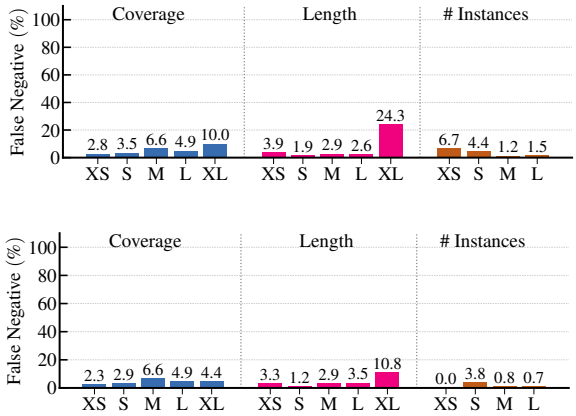
Setting	E2E	Param.	Mem.	mAP	gains
Snippet Feature	✗	-	-	36.64	
+ LongLoRA [5]	✓	28M	6.2G	37.69	+1.05
+ Full FT	✓	86M	5.6G	38.13	+1.49
+ Plain Adapter [12]	✓	3.6M	4.8G	38.21	+1.57
+ AdaTAD (w.o. residual)	✓	4.0M	4.9G	38.23	+1.59
+ **AdaTAD**	✓	**4.0M**	**4.9G**	**38.39**	**+1.75**
+ Full FT + TIA	✓	90M	5.8G	37.89	+1.25

D.5. AdaTAD with Different Detector Head

Our end-to-end framework with the introduced adapter is effective not only on ActionFormer [25], but also with other TAD heads. As shown in Table 7, we successfully employ the proposed method on GTAD [23] and TriDet [18]. When using VideoMAE-Large as the backbone, we observe significant improvements. For instance, compared to using densely extracted snippet features (16 frames per snippet with a resolution of 224^2), our approach elevates GTAD's performance from 50.8% to 55.5%, and TriDet's performance from 68.8% to 74.1%. Additionally, compared to ActionFormer, TriDet achieves better detection performance with offline features and end-to-end training, thanks to the proposed SGP layer and Trident-Head. Overall, our proposed end-to-end training paradigm is agnostic to different action detectors, and it can boost the detector's performance by a large margin.



(a) **Sensitive Analysis.** *Left*: normalized mAP at tIoU=0.5 under different video contents. *Right*: The relative normalized mAP change at tIoU=0.5 with respect to different characteristics of the ground truth instances.



(b) **False Negative Profiling.** The false negative rates are broken down into fine-grained metrics under different coverage, length, and the number of instances.

Figure 2. We adopt the VideoMAEv2-giant as the backbone and report more error analysis of our method on THUMOS14 using [1]. ***The first row*** denotes that we use snippet features to achieve an average mAP of 69.6%. ***The second row*** denotes that we use end-to-end training by AdaTAD† and achieve an average mAP of 75.4%.

Table 7. **Ablation of different detector heads.** VideoMAE-L is used as the backbone on THUMOS dataset.

Detector Head	Setting	0.3	0.5	0.7	mAP
GTAD [23]	Feature	65.8	53.6	31.3	50.8
	AdaTAD	**69.5**	**57.6**	**37.4**	**55.5**
ActionFormer [25]	Feature	82.9	70.8	42.7	66.5
	AdaTAD	**87.7**	**76.7**	**52.4**	**73.5**
TriDet [18]	Feature	84.0	73.4	45.1	68.8
	AdaTAD	**88.7**	**78.1**	**52.2**	**74.1**

E. Error Analysis

Apart from the false positive profiling provided in the main paper, we also present the sensitive analysis and false negative (FP) profiling in Fig. 2. Note that the first row utilizes offline snippet features, and the second row utilizes the end-to-end training by AdaTAD†. For the error analysis process and metrics, we refer the readers to [1] for more details.

From Fig. 2(a), we can find that the performance across all metrics is improved by end-to-end training. Especially, the mAP of long actions (XL) is visibly increased. This phenomenon is more evident in false negative profiling. In Fig. 2(b), end-to-end training significantly reduces the FP rate from 24.3% to 10.8%, leading to better detection accuracy. Moreover, even for small actions, our method also alleviates false negative detection. For instance, it amazingly reduces the FP rate from 6.7% to 0% under the XS #instances.

F. Visualization

Further, we present the qualitative visualization of our prediction on THUMOS14 dataset. In Fig. 3, we plot the ground truth actions of each video (drawn in red and above the black line), and also the top-20 predicted proposals (drawn in colors and under the black line). The color of the proposal represents the maximum IoU of this proposal to the ground truth actions. Therefore, a proposal with a deeper color means it overlaps more with the ground truth, indicating this is a high-quality proposal. From the figure, we can observe that our method can yield accurate candidate actions and also provide reasonable proposal ranking.

G. Limitations and Future Work

One limitation of our method is how to further scale up the input data, since some datasets require extremely long video input. For instance, on Ego4D-MQ dataset, we utilize the VideoMAE-Large with 7,200 frames, costing 60GB per video. Although this is already an amazing data size for end-to-end training, however, it's meaningful to further scale up the frame resolution or model size to achieve better performance with reduced memory usage. On the other hand, loading, decoding, and processing such long videos take much longer time than pre-extracted features, which may cause difficulty in network training.

Interesting future directions include end-to-end training with multi-modality tasks, *e.g.*, end-to-end video grounding and end-to-end moment retrieval, pretraining for action localization, and open vocabulary end-to-end temporal action detection.

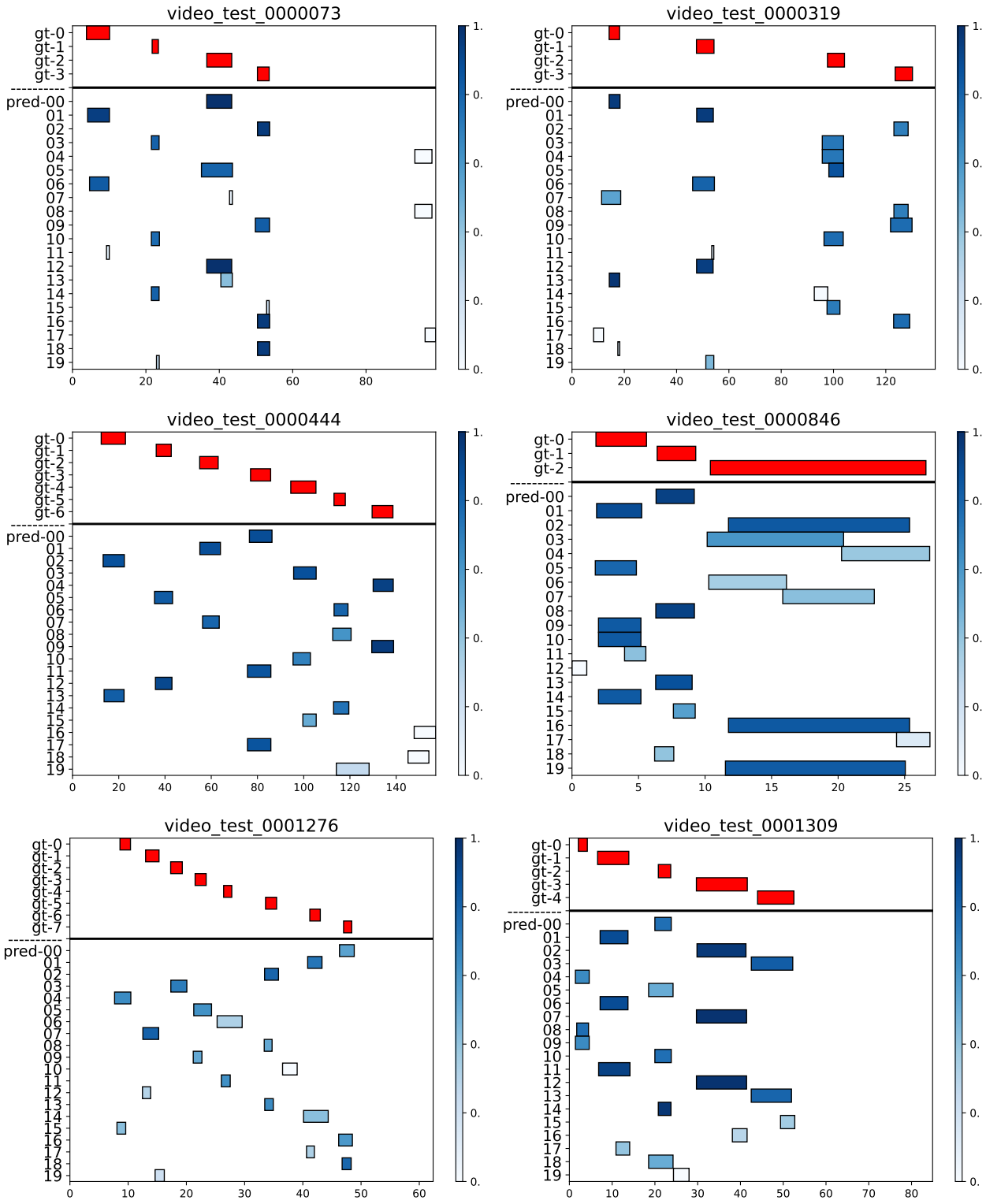


Figure 3. **Qualitative results of our method with VideoMAEv2-giant on THUMOS14.** The color of the proposal represents the maximum IoU of this proposal to ground truth actions. We plot the ground truth actions of each video (drawn in red and above the black line), and top-20 predicted proposals (drawn in colors and under the black line).

References

[1] Humam Alwassel, Fabian Caba Heilbron, Victor Escorcia, and Bernard Ghanem. Diagnosing error in temporal action detectors. In *ECCV*, 2018. 5

[2] Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019. 2

[3] Guo Chen, Sen Xing, Zhe Chen, Yi Wang, Kunchang Li, Yizhuo Li, Yi Liu, Jiahao Wang, Yin-Dong Zheng, Bingkun Huang, et al. Internvideo-ego4d: A pack of champion solutions to ego4d challenges. *arXiv preprint arXiv:2211.09529*, 2022. 2, 3

[4] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv preprint arXiv:1604.06174*, 2016. 2

[5] Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023. 4

[6] Feng Cheng and Gedas Bertasius. Tallformer: Temporal action localization with long-memory transformer. In *ECCV*, 2022. 2

[7] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, 2018. 2

[8] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *NeurIPS*, 2022. 2

[9] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast networks for video recognition. In *ICCV*, 2019. 2, 3

[10] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, and et al. Liu, Xingyu. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, 2022. 2

[11] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A large-scale video benchmark for human activity understanding. In *CVPR*, 2015. 2

[12] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *ICML*, 2019. 4

[13] YG Jiang, J Liu, A Roshan Zamir, G Toderici, I Laptev, M Shah, and R Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014. 2

[14] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2

[15] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022. 2, 3

[16] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *ICLR*, 2017. 2

[17] Jiayi Shao, Xiaohan Wang, Ruijie Quan, Junjun Zheng, Jiang Yang, and Yi Yang. Action sensitivity learning for temporal action localization. In *ICCV*, 2023. 3

[18] Dingfeng Shi, Yujie Zhong, Qiong Cao, Lin Ma, Jia Li, and Dacheng Tao. Tridet: Temporal action detection with relative boundary modeling. In *CVPR*, 2023. 4, 5

[19] Lin Sui, Fangzhou Mu, and Yin Li. Nms threshold matters for ego4d moment queries–2nd place solution to the ego4d moment queries challenge 2023. *arXiv preprint arXiv:2307.02025*, 2023. 2

[20] Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. In *NeurIPS*, 2022. 1, 3

[21] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *NeurIPS*, 2022. 2, 3

[22] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *ICCV*, 2023. 2

[23] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-TAD: Sub-graph localization for temporal action detection. In *CVPR*, 2020. 4, 5

[24] Dongshuo Yin, Xueting Han, Bin Li, Hao Feng, and Jing Bai. Parameter-efficient is not sufficient: Exploring parameter, memory, and time efficient adapter tuning for dense predictions. *arXiv preprint arXiv:2306.09729*, 2023. 1

[25] Chenlin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *ECCV*, 2022. 2, 3, 4, 5

[26] Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In *ICCV*, 2021. 2, 3