An Empirical Study of End-to-End Temporal Action Detection Supplementary Material

Xiaolong Liu[1] Song Bai[2] Xiang Bai[1*]
[1]Huazhong University of Science and Technology [2]ByteDance Inc.
{liuxl, xbai}@hust.edu.cn, songbai.site@gmail.com

A. Implementation Details

Initialization of Video Encoders. All the video encoders are initialized with the pretrianed models on Kinetics-400 [2]. This is similar to the common practice of ImageNet [3] pre-training in image understanding.

Combination with External Video Labels on ActivityNet. As mentioned in the main body of the paper, most videos on ActivityNet only contain one action class. Therefore, most previous works [1, 9, 10, 15, 17, 22–24, 26, 28], **including some end-to-end methods** [7, 11, 16], decompose temporal action detection (TAD) into class-agnostic temporal localization and video-level action classification. Following these works, we use the video-level action classification results of [27], a winning solution in ActivityNet Challenge 2017. To be concrete, we assign the top two video-level classes predicted by [27] to all class-agnostic detections, forming class-aware detections. The confidence score of the original class-agnostic detection and the classification score by [27] are fused by multiplication.

Implementation Details of AFSD. When implementing AFSD [7], we follow the details in their official code. They use a smaller batch size (1 *vs*. 4), a smaller learning rate (10^{-5} *vs*. 10^{-4}), and a longer training schedule (16 epochs *vs*. 12 epochs). We tried the setting defined in this paper but observed a performance drop of around 1.5% average mAP. Therefore, we stick to the original settings.

B. Additional Results

The Effect of Batch Size. Tab. A1 studies the effect of batch size for training. We change the learning rate following the linear scaling rule [5] when changing the batch size. We observe that increasing batch size from 4 (the default setting) to 16 results in similar performance. Therefore, we may use a larger batch size to improve GPU utilization and speed up training. For example, we can finish training of TadTR [14] with TSM ResNet-18 [8] encoder on ActivityNet using two NVIDIA RTX 3090 GPUs in **41 minutes**.

*Corresponding author

Table A1. The effect of batch size, measured by average mAP on ActivityNet. Encoder: **TSM ResNet-18**. Detector: **TadTR**. Only cropping and horizontal flipping augmentations are used. All models are trained using a single GPU (except * uses two GPUs).

Batch Size	4	8	16
mAP	33.40	33.43	33.25
Training Time	96min	85min	41min*

Even so, we stick to a relatively small batch size so that our experiments can be reproduced more easily.

Generality of the Effectiveness of End-to-End Training. In the main body of this paper, we validate the effectiveness of end-to-end training on TadTR with TSM [8] encoders. Here we also conduct the validation on more video encoders (SlowFast [4] and I3D [2]) and detection heads (AFSD [7] and G-TAD [23]). The results are summarized in Tab. A2. All results are obtained with the default setting. We see that end-to-end learning consistently improves detection performance. The performance gain ranges between 9.79% and 10.54% on THUMOS14 [6]. On ActivityNet, the performance gain is at least 1.77%. The results show that the effectiveness of end-to-end learning is general. Note that the results of G-TAD with TSM ResNet-18 are lower than those in BSP [21] and LowFi [22]. The reason might be the low resolution of videos.

It is interesting that the performance difference between TadTR and G-TAD increases when we switch from head-only learning to end-to-end learning. It indicates that traditional head-only learning might not be appropriate for benchmarking different approaches, as head-only learning restricts the performance of an approach.

Besides THUMOS14 and ActivityNet, we also evaluate the effect of end-to-end learning on HACS Segments [25]. As can be observed in Tab. A3, end-to-end training results in a performance gain of 6.42% in terms of average mAP. This again demonstrates the benefit of E2E learning and its generality.

Encoder Head	I3D TadTR	SF R50 TadTR	TSM R50 AFSD
Head-only	32.66	44.55	30.52
E2E	**45.06**	**54.17**	**40.31**
Gain	+10.54	+9.62	+9.79

(a) **THUMOS14**

Encoder Head	SF R50 TadTR	 G-TAD	TSM R18 G-TAD
Head-only	32.61	32.53	31.12
E2E	**35.10**	**34.36**	**32.89**
Gain	+2.49	+1.83	+1.77

(b) **ActivityNet**

Table A2. End-to-end learning is effective for different video encoders and detection heads. SF: SlowFast. R18/50: ResNet-18/50. Performance measured by average mAP.

Paradigm	0.5	0.75	0.95	Avg. (Gain)
Head-only	30.69	18.94	5.26	19.28
E2E	**40.32**	**24.97**	**7.71**	**25.70** (+6.42)

Table A3. The effect of end-to-end learning on HACS Segments. Encoder: TSM ResNet-50. Head: TadTR.

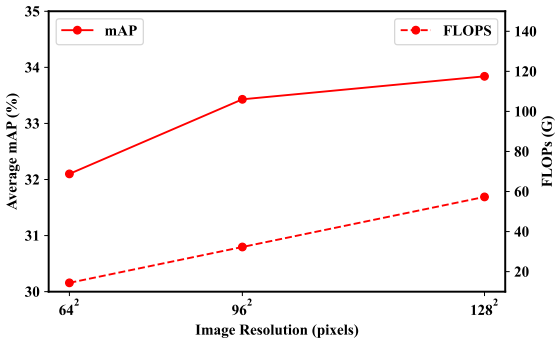

Figure A1. The effect of **spatial** resolution on ActivityNet. Encoder: **TSM ResNet-18**. Head: **TadTR**.

The Effects of Image and Temporal Resolution on ActivityNet. Fig. A1 and Fig. A2 illustrate the effects of image resolution and temporal resolution (number of input frames) on ActivityNet, respectively. Similar to THUMOS14, increasing image resolution steadily boosts detection performance and also increases computation cost. As the image resolution reaches 96^2, the performance gain of increasing image resolution decreases. Compared to image resolution, the detection performance is less sensitive to temporal resolution. The reason might be that the action instances on ActivityNet are relatively longer than those on THUMOS14.

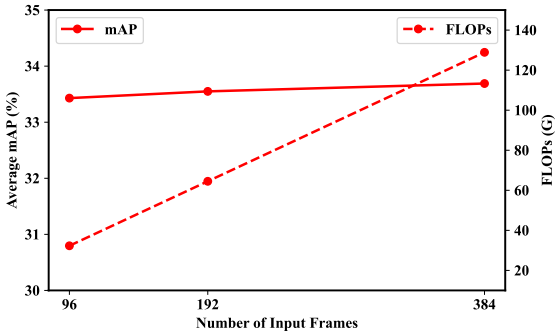

Figure A2. The effect of **temporal** resolution (number of input frames) on ActivityNet. Encoder: **TSM ResNet-18**. Head: **TadTR**.

Frame Rate	20FPS	10FPS	
Feature Fusion	-	-	✓
mAP	47.5	42.1	45.1

Table A4. The effect of multi-scale feature fusion. Encoder: I3D. Detector: TadTR.

Besides, the action classes on ActivityNet are more related to scenes than motion. This observation also supports our choice of using sparse frame sampling on ActivityNet.

The Effect of Feature Fusion. In the I3D encoder, we fuse the features from the fourth stage and fifth stage. We verify the effectiveness of this strategy in Tab. A4. It is observed that this strategy improves the performance by 3%. It helps to compensate for the temporal information loss due to a decrease of the frame rate from 20 FPS to 10 FPS.

The Effect of the Frame Sampling Manner. By default, the encoder takes all frames from an input clip and extract feature in a temporally fully convolutional manner in our study. An alternative way is to sample fixed-length snippets one by one in a sliding-window manner and extract features for each snippet. The length of snippets is defined when the video encoder is pre-trained for action recognition (*e.g.*, 8 for TSM). It is adopted by most works [9, 12, 23] based on offline features. However, it actually increases the total computation cost as adjacent snippets overlap with each other. In end-to-end training, we can still use this manner, at the expense of efficiency. Due to a high memory usage, we are only able to conduct the experiment with TSM ResNet-18 on 4 GPUs. As can be observed in Tab. A5, the snippet-wise manner actually gives lower performance than the fully convolutional manner, probability due to a limited temporal receptive field.

	mAP	FLOPs
snippet-wise	34.25	171.8G
fully-convolutional	36.12	21.5G

Table A5. The effect of frame sampling manner. Encoder: TSM ResNet-18. Detector: TadTR. Dataset: THUMOS14. FLOPs are measured on clips of 25.6 seconds.

C. Computation Cost Analyses

Reasons for the Lower Computation Cost Than [13]. In Tab. 1 of the main document, we show that the detector built in this work is 126× faster (587ms *vs*. 74.1s) than the previous state-of-the-art non-end-to-end method [13]. The speed-up comes from three aspects. **Firstly**, we use a smaller image size and a lower frame rate. The setting of image size and frame rate in [13] is 224^2 and 30 FPS, while the setting in our case is 96^2 and 10 FPS. **Secondly**, we extract features in a fully convolutional manner instead of the conventional snippet-wise manner used in [13]. To be concrete, they use a sliding window strategy to sample snippets of 64 frames, which is the default input length of the I3D encoder, for feature extraction. The stride between two adjacent windows is 8 frames, 1/8 of the window length. Therefore, redundant computation is introduced. **Finally**, the encoder (SlowFast) and the head (TadTR) are more efficient than those in [13]. We note that these issues are not unique to [13]. They are prevalent in previous methods based on offline features and impede the application of TAD in real-world scenarios. We believe that end-to-end TAD can help eliminate these obstacles.

Computation Cost of Various Video Encoders. Except for the video encoders studied in the main paper, we analyze the computation cost of several video encoders used in previous end-to-end methods [16, 20] and the pre-training method [1] in Tab. A6. As they have different settings in temporal pooling, we adjust the number of sampled frames to ensure that they output features of the same length. We observe that C3D [18] and R(2+1)D-18/34 [19] are much heavier than SlowFast, although the former two have shallower backbones. For example, C3D, the fastest among them, is around 5× slower than SlowFast. They are less appropriate for temporal action detection. Therefore we do not use them in our experiments.

Training Time. Using the settings described in the implementation details, training SlowFast with TadTR head takes around 4 GPU hours on THUMOS14. On ActivityNet, the training time is around 11 and 1.5 GPU hours using SlowFast and TSM ResNet-18, respectively.

References

[1] Humam Alwassel, Silvio Giancola, and Bernard Ghanem. Tsp: Temporally-sensitive pretraining of video encoders for localization tasks. In *ICCV Workshops*, 2021. 1, 3

[2] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, pages 4724–4733, 2017. 1, 4

[3] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee, 2009. 1

[4] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *ICCV*, pages 6202–6211, 2019. 1, 4

[5] Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017. 1

[6] YG Jiang, Jingen Liu, A Roshan Zamir, G Toderici, I Laptev, Mubarak Shah, and Rahul Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014. 1

[7] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning salient boundary feature for anchor-free temporal action localization. In *CVPR*, pages 3320–3329, 2021. 1

[8] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *ICCV*, pages 7083–7093, 2019. 1, 4

[9] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *ICCV*, pages 3889–3898, 2019. 1, 2

[10] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *ECCV*, September 2018. 1

[11] Qinying Liu and Zilei Wang. Progressive boundary refinement network for temporal action detection. In *AAAI*, volume 34, pages 11612–11619, 2020. 1

[12] Xiaolong Liu, Yao Hu, Song Bai, Fei Ding, Xiang Bai, and Philip HS Torr. Multi-shot temporal event localization: a benchmark. In *CVPR*, pages 12596–12606, 2021. 2

[13] Xiaolong Liu, Yao Hu, Song Bai, Fei Ding, Xiang Bai, and Philip H. S. Torr. Multi-shot temporal event localization: A benchmark. In *CVPR*, pages 12596–12606, June 2021. 3

[14] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *arXiv preprint arXiv:2106.10271*, 2021. 1

[15] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *CVPR*, pages 344–353, 2019. 1

[16] Zheng Shou, Jonathan Chan, Alireza Zareian, Kazuyuki Miyazawa, and Shih-Fu Chang. Cdc: Convolutional-deconvolutional networks for precise temporal action localization in untrimmed videos. In *ICCV*, pages 1417–1426, 2017. 1, 3

[17] Deepak Sridhar, Niamul Quader, Srikanth Muralidharan, Yaoxin Li, Peng Dai, and Juwei Lu. Class semantics-based

Encoder	Input Length	Output Length	FLOPs (G)	Latency (ms)
TSM R18 [8]	64	64	21.5	**21.2**
TSM R50 [8]	64	64	48.8	37.0
I3D* [2]	256	64	83.2	62.1
SlowFast R50* [4]	256	64	41.4	51.2
C3D (VGG-11) [18]	512	64	906	277
R(2+1)D-18 [19]	512	64	960	382
R(2+1)D-34 [19]	512	64	1803	638

Table A6. Comparison of the computation cost of various video encoders, measured on video clips of 25.6 seconds. image resolution: 96^2. *Modified by us to make a temporal output stride of 4.

attention for action detection. In *ICCV*, pages 13739–13748, 2021. 1

[18] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, pages 4489–4497, 2015. 3, 4

[19] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *CVPR*, pages 6450–6459, 2018. 3, 4

[20] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: region convolutional 3d network for temporal activity detection. In *ICCV*, pages 5794–5803, 2017. 3

[21] Mengmeng Xu, Juan-Manuel Pérez-Rúa, Victor Escorcia, Brais Martinez, Xiatian Zhu, Li Zhang, Bernard Ghanem, and Tao Xiang. Boundary-sensitive pre-training for temporal localization in videos. In *ICCV*, pages 7220–7230, 2021. 1

[22] Mengmeng Xu, Juan-Manuel Perez-Rua, Xiatian Zhu, Bernard Ghanem, and Brais Martinez. Low-fidelity video encoder optimization for temporal action localization. In *NeurIPS*, 2021. 1

[23] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-TAD: Sub-graph localization for temporal action detection. In *CVPR*, pages 10156–10165, 2020. 1, 2

[24] Chen Zhao, Ali K Thabet, and Bernard Ghanem. Video self-stitching graph network for temporal action localization. In *ICCV*, pages 13658–13667, 2021. 1

[25] Hang Zhao, Antonio Torralba, Lorenzo Torresani, and Zhicheng Yan. HACS: human action clips and segments dataset for recognition and temporal localization. In *ICCV*, pages 8667–8677, 2019. 1

[26] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *ECCV*, 2020. 1

[27] Yue Zhao, Bowen Zhang, Zhirong Wu, Shuo Yang, Lei Zhou, Sijie Yan, Limin Wang, Yuanjun Xiong, Wang Yali, Dahua Lin, Yu Qiao, and Xiaoou Tang. CUHK & ETHZ & SIAT submission to ActivityNet challenge 2017. *arXiv preprint arXiv:1710.08011*, pages 20–24, 2017. 1

[28] Zixin Zhu, Wei Tang, Le Wang, Nanning Zheng, and Gang Hua. Enriching local and global contexts for temporal action localization. In *ICCV*, pages 13516–13525, 2021. 1