Supplementary Material: Rethinking the Faster R-CNN Architecture for Temporal Action Localization

Yu-Wei Chao[1*], Sudheendra Vijayanarasimhan[2], Bryan Seybold[2], David A. Ross[2], Jia Deng[1], Rahul Sukthankar[2]

[1]University of Michigan, Ann Arbor
{ywchao,jiadeng}@umich.edu

[2]Google Research
{svnaras,seybold,dross,sukthankar}@google.com

1. Training Strategy

As in Faster R-CNN [3], the training of proposal generation and action classification share a same form of multi-task loss, targeting both classification and regression:

$$\mathcal{L} = \sum_i \mathcal{L}_{cls}(p_i, p_i^*) + \lambda \sum_i [p_i^* \geq 1] \mathcal{L}_{reg}(t_i, t_i^*). \quad (1)$$

i is the index of an anchor or proposal in a mini-batch. For classification, p is the predicted probability of the proposal or actions, p^* is the ground-truth label, and $\mathcal{L}_{cls}$ is the cross-entropy loss. Note that $p^* \in \{0, 1\}$ for proposal generation, and $p^* \in \{0, \ldots, C\}$ for action classification, where C is the number of action classes of interest and 0 accounts for the background action class. For regression, t is the predicted offset relative to an anchor or proposal, t^* is the ground-truth offset, and $\mathcal{L}_{reg}$ is the smooth L1 loss defined in [2]. We parameterize the offsets $t = (t_c, t_l)$ and $t^* = (t_c^*, t_l^*)$ by:

$$\begin{aligned} t_c &= 10 \cdot (c - c_a)/c_i, \quad & t_l &= 5 \cdot \log(l/l_a), \\ t_c^* &= 10 \cdot (c^* - c_a)/c_i, \quad & t_l^* &= 5 \cdot \log(l^*/l_a), \end{aligned} \quad (2)$$

where c and l denote the segment's center coordinate and its length. c and c^* account for the predicted and ground-truth segments, while c_a accounts for the anchor and proposal segments, for proposal generation and action classification, respectively (similarly for l). The indicator function $[\cdot]$ is used to exclude the background anchors and proposals when the regression loss is computed. In all experiments, we set $\lambda = 1$ for both proposal generation and action classification, and jointly train both stages by weighing both losses equally.

For proposal generation, an anchor is assigned a positive label if it overlaps with a ground-truth segment with temporal Intersection-over-Union (tIoU) higher than 0.7. A negative label is assigned if the tIoU overlap is lower than 0.3 with all ground-truth segments. We also force each ground-truth segment to have at least one matched positive anchor.

*Work done in part during an internship at Google Research.

	InceptionV3 RGB (ImageNet pre-trained)
Zhao et al. [5]	18.3
Ours	**26.0**

Table 1: Action localization mAP (%) on THUMOS'14 using InceptionV3. The result of [5] is copied from [1].

For action classification, a proposal is assigned the action label of its most overlapped ground-truth segment, if the ground-truth segment has tIoU overlap over 0.5. Otherwise a background label (i.e. 0) is assigned.

Each mini-batch contains examples sampled from a single video. For proposal generation, we set the mini-batch size to 256 and the fraction of positives to 0.5. For action classification, we set the mini-batch size to 64 and the fraction of foreground actions to 0.25. We use the Adam optimizer with a learning rate of 0.0001.

2. Sample Results

We also include a set of videos showcasing our results on THUMOS'14 in this supplementary material package. Each video file shows a full test video along with a progress bar pinpointing the start and end times of ground-truth and top predicted action segments. The predicted segments are sorted by their confidence scores. We create the visualization interface based on the open source tool VATIC [4].

Our approach successfully localizes the actions in most cases. The failure cases include: (1) inaccurate boundaries (e.g. `02-0737.mp4`), (2) misclassified actions (e.g. `03-1527.mp4`, `04-0444.mp4`), (3) false positives due to indistinguishable body motions (e.g. `05-0046.mp4`, `11-1168.mp4`), and (4) false negatives due to small objects and occlusion (e.g. `07-1135.mp4`).

3. Benchmarks using InceptionV3

Besides I3D features, we also evaluate our method with features extracted from an InceptionV3 model pre-trained

Step	Running Time (ms)	
Optical Flow	239	per frame
I3D Features	825	per 16 frame input
Proposal + Classification	9	per 3000 frames

Table 2: Running time (ms) of each step during test time.

on ImageNet. This provides an apples-to-apples comparison with the result of Zhao et al. [5] reported in [1]. Tab. 1 shows the action localization mAP on THUMOS'14. Our approach outperforms Zhao et al. [5] by 7.7% in mAP, validating the effectiveness of our proposed architecture.

4. Computational Cost

Tab. 2 shows a running time breakdown during the test time for the following three steps: (1) optical flow extraction, (2) I3D feature extraction, and (3) proposal and classification. All these running time experiments are performed on CPUs, so further speedup is possible with GPU devices. The computational bottleneck is on the optical flow extraction (i.e. 239 ms per frame).

References

[1] https://github.com/yjxiong/action-detection. 1, 2

[2] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 1

[3] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*. 2015. 1

[4] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *IJCV*, 101(1):184–204, Jan 2013. 1

[5] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, D. Lin, and X. Tang. Temporal action detection with structured segment networks. In *ICCV*, 2017. 1, 2