Improve Baseline for Temporal Action Detection: HACS Challenge 2020 Solution of IVUL-KAUST team

Mengmeng Xu, Chen Zhao, Merey Ramazanova, David S. Rojas, Ali Thabet, and Bernard Ghanem

King Abdullah University of Science and Technology, Saudi Arabia

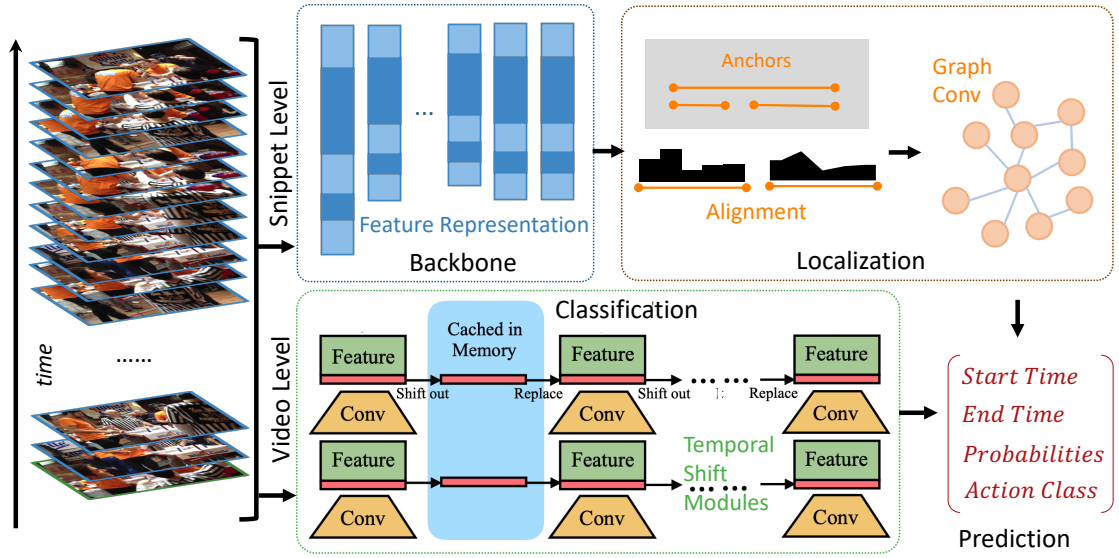

Figure 1: **Temporal Action Detection Framework.** We take two branches to detect actions from a video. In the action localization branch, we predict the action boundary with a confidence score. In the action classification branch, we classify the action from the entire video. Combining the output of both branches, we can have the action boundary, class, and confidence.

Abstract

In this report, we present our solution for the HACS Temporal Action Localization Challenge - Weakly-supervised Learning Track. Temporal action localization is an important yet challenging task in video analysis. Motivated by previous works on this task, we designed a three-stage temporal action detection framework. First, a backbone module helps to enrich and align video features to each proposal. Second, we build a proposal graph to predict action boundaries. Third, we train a video classification network. The final output is obtained by merging the proposal score with the video scores. With the proposed solution, our team achieved 23.54% *average mAP on the HACS Challenge 2020, and can achieved* 28.90% *average mAP with proper hyper-parameter settings.*

1. Proposed solution

Our proposed solution is a three-stage temporal action detection framework. It takes a video sequence as input and predicts multiple scored candidate actions. First, we leverage a backbone module to enhance and align video features to each proposal. Second, we create a proposal graph to predict action boundaries. Then we apply a video classification network to predict the action classes for the proposals. The whole framework is shown in Fig. 1.

1.1. Backbone Module

Our backbone module learns video representations and is comprised of the following sequential modules.

First, the input sequence is passed through four convolutional layers. These layers reduce the sequence temporal resolution (for the computational efficiency in the subsequent layers). Then a layer of multiple dilated convolutions with different dilation ratios aggregate temporal information from different scales. Afterwards, a linear interpolation layer followed by a convolutional layers upscales the temporal resolution back.

1.2. Proposal Localization

Based on SGAlign [7], we extract segments of features from the video representations as proposals. The proposals in the same video are highly correlated and utilizing this property can facilitate proposal recognition [4, 8]. Considering that instead of densely generating proposals, we sample and obtain sparse proposals, so grid operations such as 2D convolutions no longer apply. We use GCNs to model proposal-proposal correlations the proposal classifier of our framework.

In our GCN-based network, we represent each proposal as a node, and proposals-proposal correlations as edges. Notably, the edges here are constructed based on the temporal correlations, which are formulated as the IoU between two proposals.

1.3. Video Classification

The video-level class prediction has shown to be effective context to the temporal action detection problem.[5] We leverage TSM[3] for video classification. TSM proposes a generic and effective Temporal Shift Module that enjoys both high efficiency and high performance. Specifically, it can achieve the performance of 3D CNN but maintain 2D CNN's complexity. TSM shifts part of the channels along the temporal dimension; thus facilitate information exchanged among neighboring frames. Our video classification model architecture follows Temporal Segment Network[6], but the residual connections are equipped with TSM.

1.4. Training and Inference

Training. We use the localization loss L_{loc}, snippet classification loss L_{cls}, and an $\mathcal{L}_2$-norm regularization loss $\mathcal{L}_r$ for training the entire network, formulated as $\mathcal{L} = \mathcal{L}_{loc} + \mathcal{L}_{cls} + \lambda\mathcal{L}_s$ where λ denotes the weight decay. The loss L_{loc} is used to determine the confidence scores of proposals, while the loss L_{cls} classify the video snippet as start, end, or actions. In addition, in order to reduce the computation for training the GCN, we adopt the SAGE sampling strategy suggested in [8].

Inference. At inference time, we only run the localization head to generate the score for each proposal. We construct predicted actions $\Phi = \left\{\phi_j = (\hat{t}_{s,j}, \hat{t}_{e,j}, \hat{c}_j, p_j)\right\}_{j=1}^{J}$, where $(\hat{t}_{s,j}, \hat{t}_{e,j})$ refer to the predicted action boundaries, $\hat{c}_j$ is the predicted action class, and p_j is the fused confidence score of this prediction, computed as $p_j = p_{cls}^{\alpha} \cdot p_{reg}^{1-\alpha}$. In our experiments, we search for the optimal α in each setup. After that, we run soft non-maximum suppression (NMS) and remove low-scored predictions.

2. Experiments

2.1. Action Classification

We finetune the TSM model on the HACS Segment dataset. The TSM architecture is adopted from temporal segment network, which takes ResNet50 as the backbone, but shift the channel on the residual connections to involve temporal information. The model is pretrained on kinetics-600 and takes as input 8 video frames and predict action class of the video.

The finetuning takes 25 epochs. The initial learning rate is 0.001, then we reduced it by 10 on the 10th and 20th epochs. We use the official training set to train the model and save the model that reaches the best precision. In both training and validation, we only use video-level annotations – the action class of each video. Tab 1 shows model performance on the validation set of HACS dataset.

Table 1: **TSM model performance on the validation set of HACS dataset.**

Dataset	Class Accuracy	Prec@1	Prec@5
Kinetics-400	74.14%	74.12%	91.21%
HACS Segment	85.44%	85.47%	98.01%

2.2. Action Localization

Video Feature We compare video feature extracted from different models. (1) TSN feature: We extract the video frames from HACS Segment dataset. Then we feed the frames in the TSN model pretrained on Kinetics and save the output from the last fully connect layer. (2) TSM feature: similar to TSN feature, we extract video feature from our finetuned TSM model from Sec. 2.1. Since the model is finetuned on the dataset, we expect it to be more representative. (3) I3D feature: It is provided by HACS organizers. The feature is from the global pooling layer of a I3D model pretrained on Kinetics-400.

BSN experiment We compared TSN and TSM features on BSN. We use the public BSN code to produce the action proposals for HACS validation set, and evaluate them by the average recall at top 1, top 100, and the averaged recall over

the curve (AUC), shown in Tab. 2. We also assign the proposal class by the predicted video class, and evalute the detection result by Average mAP over IoUs in [0.5:0.05:0.95], shown in the last column of Tab. 2. Comparing the two rows, the finetuned model doesn't always produce better video features for the proposal general and action detection task. In our experiment, the pretrained model gives more general video features and produces better model performance.

Table 2: **BSN model performance on the validation set of HACS dataset.**

Feature	AR@1	AR@100	AUC	mAP
TSN	12.24	54.20	41.58	13.96
TSM	11.64	47.43	36.26	12.29

2.3. Experiment on our proposed solution.

we use the publicly available features extracted using an I3D-50 [2] model pre-trained on Kinetics-400 [2] and temporally rescale them into 400 snippets.

We implement and test our framework using PyTorch 1.1, Python 3.7, and CUDA 10.0. In training, the learning rates is 2E−3 on HACS-v1.1 for the first 7 epochs, and are reduced by 10 for the following 8 epochs. In inference, we leverage the global video context and take the video classification scores from action recognition model and [3], and multiply them by the confidence score for evaluation.

Tab. 3 compares our baseline method with representative temporal action detectors. We report mAP at different tIoU thresholds, as well as average mAP.

Our method reaches 28.90% average mAP on the test set, surpassing S-2d-TAN[9], the winner of HACS Challenge 2019, with a large margin.

Table 3: **Action detection results on HACS-v1.1**, measured by mAP (%) at different tIoU thresholds and the average mAP. '-' means the results are not provided in the papers.

Method	Validation				Test
	0.5	0.75	0.95	Average	Average
SSN [10]	28.82	18.80	5.32	18.97	16.10
BMN [1]	-	-	-	-	22.10
S-2D-TAN [9]	-	-	-	-	23.49
ours*	40.26	26.96	**8.08**	27.01	23.54
ours	**43.33**	**29.65**	6.23	**29.24**	**28.90**

* submission to the leader board.

References

[1] Report of temporal action proposal. http://hacs.csail.mit.edu/challenge/challenge19_report_runnerup.pdf. Accessed: 2020-06-5.

[2] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[3] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.

[4] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3888–3897, 2019.

[5] Merey Ramazanova, Chen Zhao, Mengmeng Xu, Humam Alwassel, Sara Rojas Martinez, Fabian Caba, and Bernard Ghanem. Logistic regression is still alive and effective: The 3rd youtube 8m challenge solution of the ivul-kaust team.

[6] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[7] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. *arXiv preprint arXiv:1911.11462*, 2019.

[8] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7093–7102, 2019.

[9] Songyang Zhang, Houwen Peng, Le Yang, Jianlong Fu, and Jiebo Luo. Learning sparse 2d temporal adjacent networks for temporal action localization, 2019.

[10] Hang Zhao, Zhicheng Yan, Lorenzo Torresani, and Antonio Torralba. Hacs: Human action clips and segments dataset for recognition and temporal localization. *arXiv preprint arXiv:1712.09374*, 2019.