Revisiting Few-shot Activity Detection with Class Similarity Control

Huijuan Xu[1] Ximeng Sun[2] Eric Tzeng[1] Abir Das[3] Kate Saenko[2] Trevor Darrell[1]

[1]University of California, Berkeley [2]Boston University [3]IIT Kharagpur

[1]{huijuan, etzeng, trevor}@eecs.berkeley.edu, [2]{sunxm, saenko}@bu.edu, [3]abir@cse.iitkgp.ac.in

Abstract

Many activities of interest are rare events, with only a few labeled examples available. Therefore, models for temporal activity detection which are able to learn from a few examples are desirable. In this paper, we present a conceptually simple and general yet novel framework for few-shot temporal activity detection based on proposal regression which detects the start and end time of the few-shot input activities in an untrimmed video. Our model is end-to-end trainable, takes into account the frame rate differences between few-shot activities and untrimmed test videos, and can benefit from additional few-shot examples. At test time, each proposal is assigned the label of the few-shot activity class corresponding to the maximum similarity score. We experiment on three large scale benchmarks for temporal activity detection (ActivityNet1.2, ActivityNet1.3 and THUMOS14 datasets) in the few-shot setting. By controlling the novel classes' similarity with the pre-trained classification classes in creating the different base and novel data splits, we show how much our model performance is affected by this factor and give useful suggestions for future work.

1. Introduction

Few-shot activity detection is performed as example-based activity detection where one stream encodes the few-shot activity videos along with a main stream that encodes the untrimmed video and extracts candidate proposals likely to contain activities. A final label is put to the candidate proposals depending on the similarity between the query few-shot videos and the proposed candidates, instead of classifying over a predefined label space. Yang *et. al.* [7] address few-shot activity detection via example-based action detection based on the Matching Network.It utilizes correlations between the sliding window feature encoding and few-shot example video features, to localize actions of previously unseen classes. However, their model uses the "sliding window" approach to propose likely activities (proposals), and thus suffers from the high computational cost and inflexible activity boundaries.

To generate more accurate and efficient proposals for few-shot detection, in this paper we propose an approach based on direct coordinate regression for temporal localization, called Few-shot Proposal-based Activity Detection model (*F-PAD*). We encode both few-shot examples and the untrimmed video using 3D convolutional (C3D) features [5] followed by a temporal proposal network extracting action proposals from the untrimmed video. We produce similarity scores by comparing each few-shot example with the proposed candidates in the encoded feature space and assign each proposal an activity label same as that of the maximally similar few-shot video. To minimize the feature discrepancy arising from these frame rate differences, we design a distributional adaptation loss during training.

Traditionally, an action/object detection model is warm-started from a pre-trained classification backbone. However, in few-shot action / object detection task, this practice can result in potential overlap between the novel classes for detection and the activity classes seen by the pre-trained classification backbone. Still, few-shot object detection [1, 4] uses pre-trained imagenet classification backbone riding on the justification that the labels used in classification and detection are different. As detection requires bounding box annotations in addition to the class labels, the argument is that detection still remains few shot even if the backbone is trained on a large-scale classification dataset. In this paper, we investigate the effect of the potential class overlap between training classes in the pre-trained activity classification model and the few-shot activity detection model. The analysis is done by creating different random splits of train (base classes) and test (novel classes) data where the overlap of novel classes with large-scale pretraining dataset is a control variable.

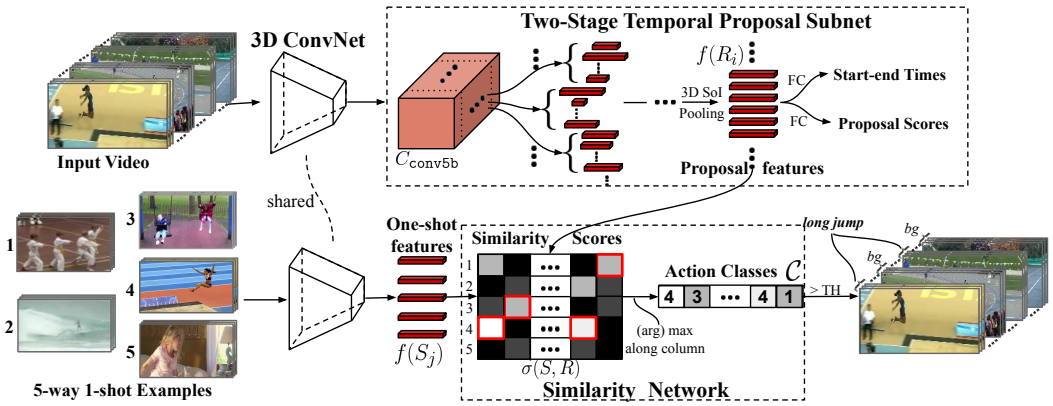

Figure 1. Detail diagram of our proposed F-PAD network for exampled based few-shot activity detection.

2. Example based Temporal Activity Detection

A detailed diagram of our F-PAD network is in Figure 1. The model consists of three components: a 3D ConvNet feature extractor [5], a temporal proposal network, and a similarity network. Specifically, the 3D ConvNet feature extractor encodes both the untrimmed input video and the few-shot trimmed example videos; the proposal subnet predicts temporal segments of variable lengths that contain potential activities; the similarity network classifies these proposals into one of the few-shot activity categories.

For the few-shot branch in the bottom of Figure 1, we take five trimmed videos belonging to five classes, with one video from each class in the one-shot setting. We uniformly sample $L = 16$ frames from each few-shot video, and then get a fixed-dimensional feature vector $f(S_j)$ for each video j via the 3D ConvNet (see "One-shot features" in Figure 1). For the untrimmed input video in the upper branch, we use fully convolutional layers in C3D to encode a long sequence of frames (sampled at a fixed frame rate) resulting in the feature map C_{conv5b}. From the untrimmed video feature encoding C_{conv5b}, our proposal subnet proposes candidate segment proposals and extracts proposal features $f(R_i)$ for each proposal i. The two video encoding branches share the same 3D ConvNet weights. We further minimize the feature encoding discrepancy arising from the apparent frame rate differences between inputs to these two branches using L2 penalty and learn frame rate adaptive features. We inherit the temporal proposal module from the original R-C3D temporal activity detection model [6], and additionally, we adapt the second classification module in the original model to be class agnostic and form a two-stage temporal proposal subnet, based on the previous experience that refined temporal proposals from second stage are generally better than the initial proposals from first stage.

Finally, similarities are measured between proposal features $f(R_i)$ and few-shot video feature encoding $f(S_j)$ for few-shot classification. Each proposal i is assigned a few-shot class label $\mathcal{C}_i$, based on the Similarity Score matrix $\sigma(S, R)$ between the proposal features and the few-shot video features. In the one-shot setting, $\arg\max$ is directly applied on each column of the Similarity Score matrix $\sigma(S, R)$. In multi-shot setting, we first average the similarity scores of multiple examples belonging to the same class, and then assign $\mathcal{C}_i$ by applying $\arg\max$ among the few-shot class labels. Finally, each proposal is paired with a proposal score and a few-shot class with maximum similarity score, and the final detected activity is determined by the score thresholds.

The total training loss is as follows:

$$\mathcal{L}_{total} = \mathcal{L}_{p1} + \mathcal{L}_{p2} + \mathcal{L}_{fewShot\_cls} + \lambda\mathcal{L}_{adaptation} \quad (1)$$

Each of the proposal losses L_{p1} and L_{p2} consists of a smooth L1 regression loss for proposal regression and a binary classification loss for binary proposal classification. The few-shot classification loss $L_{fewShot\_cls}$ in the few-shot Similarity Network is a multi-class cross-entropy loss.

3. Experiments

We use the same few-shot problem setup as in the previous work [7], i.e. the test classes (novel classes) having no overlap with the training classes. We also use the same meta-learning setup as in [7]. Results are shown in terms of mean Average Precision - mAP@α where α denotes Intersection over Union (IoU) threshold, and average mAP at 10 evenly distributed IoU thresholds between 0.5 and 0.95, as is the common practice in the literature. However in the meta-testing stage, mAP@0.5 and average mAP are calculated in each iteration for one untrimmed test video, and the final results are reported by averaging over 1000 iterations. In this paper, we evaluate F-PAD on three large-scale activity detection datasets - ActivityNet1.2 and ActivityNet1.3 [2], and THUMOS'14 [3]. Next, we introduce the experimental setup on these three datasets.

In the ActivityNet1.2 dataset, the trimmed videos in the few-shot branch come from the ground truth annotations inside each untrimmed video. We initialize the 3D ConvNet with C3D weights trained on Sports-1M released by

the authors in [5] for all the three datasets. This classification backbone has similar classification accuracy with the video classification backbone used in the previous work [7] which is roughly 80% on UCF-101 dataset. Regarding the base and novel data splits, the 100 activity classes are randomly split into 80 classes (ActivityNet1.2-train-80) for training and 20 classes (ActivityNet1.2-test-20) for test, and we do this random split three times resulting in "randomSplit1", "randomSplit2", "randomSplit3". These splits are used to test and evaluate the performance when there can be possible overlap between pretraining classes and the novel classes. Next, to decouple any overlap, we manually check the 100 activity classes and pick the classes those do not exist in the Sports-1M dataset. We found 63 such activity classes. Now, we randomly sample 20 classes (ActivityNet1.2-test-20) for test from these 63 classes, and the rest of 80 classes (ActivityNet1.2-train-80) are used for training. We perform this controlled random split three times resulting in "controlled_randomSplit1", "controlled_randomSplit2", "controlled_randomSplit3". We report results on these six different base and novel splits.

Regarding the base (ActivityNet1.3-train-180) and novel (ActivityNet1.3-test-20) data splits on ActivityNet1.3, we follow the same strategy as followed in ActivityNet1.2. Note that we keep ActivityNet1.2-test-20 and ActivityNet1.3-test-20 the same. In the THUMOS'14 dataset, the trimmed videos in the few-shot branch come from UCF101. Regarding the base and novel data splits, we first follow previous work [7] to split the 20 classes into 6 classes (Thumos-val-6) for training and 14 classes (Thumos-test-14) for testing, and do the random split three times without class similarity control resulting in "randomSplit1", "randomSplit2", "randomSplit3". Then we checked the class overlap between 20 classes in THUMOS'14 and the 487 classes in Sports-1M. Out of the 20 classes, 9 do not exist in Sports-1M. Thus we could create only one controlled split with 11 classes (Thumos-val-11) for training and 9 classes (Thumos-test-9) not existing in Sports-1M for testing, resulting in "controlled_randomSplit1".

3.1. One-shot Results on Different Base/Novel Splits

We run our one-shot model on the six different base and novel splits of each data set. The mean and standard deviation of the mAP@0.5 results are shown in Figure 2. The yellow histogram shows the mean and standard deviation of three different runs on "randomSplit1". From the yellow histogram, we can see that though there is certain randomness of sampling one shot example in each iteration, on the same data split, the mAP@.5 results of three different runs are quite stable with small standard deviation. The green histogram shows the mean and standard deviation of three runs on three different random base and novel splits without control on class overlap, namely "randomSplit1", "randomSplit2" and "randomSplit3". The cyan histogram shows the mean and standard deviation of three runs on three random base and novel splits with no overlap between novel classes and classes in Sports-1M. The corresponding data-splits are "controlled_randomSplit1", "controlled_randomSplit2", "controlled_randomSplit3". Comparing the green histogram with the cyan histogram, we observe that the mean mAP@0.5 decreases as novel classes become completely disjoint from pretraining classes. The standard deviations on both sets of splits (green histogram and cyan histogram) are relatively large. The reason might be that the three random splits might have very different demography between the novel and the base classes. For example, one split could have all the novel classes belong to aquatic sports while none is present in the base class while another split may have the presence of aquatic sports (belonging to disjoint classes) present in both base and novel classes. We argue that splitting randomly and showing the standard deviation on the performance would help the community to understand the robustness of the methods better. Note that for Thumos'14, there is only one controlled split without class overlap, thus no standard deviation is shown for it, and also the number of base and novel classes in this split is different on this dataset and thus results can't be directly compared.

3.2. One-shot and Five-shot Result Comparison

In Table 1, we choose two base and novel splits "randomSplit1" and "controlled_randomSplit1" from each dataset and report the one-shot and five-shot results of our F-PAD in terms of mAP@0.5 and average mAP. Though we list the results from existing work [7], based on our observation in previous section that the standard deviation is high between different base and novel splits, these results are not directly comparable to us since neither the split nor the code is public. In general, the performance of our five-shot model (F-PAD@5) is better than the one-shot model (F-PAD@1) significantly which shows the benefit of our end-to-end model in a scenario where a few more example is available for training. Besides, we find that, on ActivityNet1.2 and ActivityNet1.3, the relative improvement is less prominent for the controlled split (last two rows) compared to the uncontrolled split (4^{th} and 3^{rd} last rows) when going from 1-shot to 5-shot scenario. For "randomSplit1" and "controlled_randomSplit1" on Thumos14 dataset, the number of base and novel classes in these two splits are different, thus results can't be directly compared. "randomSplit1" has 6 classes for training and 14 classes for testing following [7], while "controlled_randomSplit1" has 11 classes for training and 9 classes for testing since we can only find 9 unpolluted classes for novel classes. However, we can observe that there is a significant drop in performance in "controlled_randomSplit1" compared to "randomSplit1". This might be because in "randomSplit1" certain novel classes

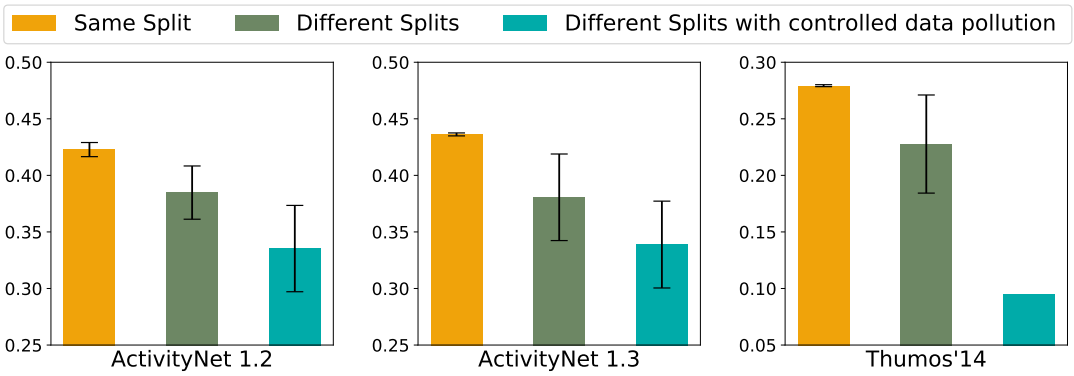

Figure 2. Mean and standard deviation of mAP@0.5 results on different base and novel class splits of ActivityNet1.2, ActivityNet1.3 and THUMOS'14 for our one-shot F-PAD model.

Table 1. Few-shot temporal activity detection results on ActivityNet1.2, ActivityNet1.3 and THUMOS'14 (in percentage). For ActivityNet1.2 and ActivityNet1.3, we report mAP at tIoU threshold $\alpha = 0.5$ and average mAP of $\alpha \in \{0.5, 0.95\}$. For THUMOS'14, we report mAP at tIoU threshold $\alpha = 0.5$. @1 means "one-shot" and @5 means "five-shot". Note that the results from paper [7] are not directly comparable to us since the base and novel split is unknown according to the analysis in Sec. 3.1

	base/novel split	ActivityNet 1.2		ActivityNet 1.3		THUMOS'14
		mAP@0.5	average mAP	mAP@0.5	average mAP	mAP@0.5
CDC@1 [7]	unknown	8.2	2.4	-	-	6.4
CDC@5 [7]	unknown	8.6	2.5	-	-	6.5
sliding window@1 [7]	unknown	22.3	9.8	-	-	13.6
sliding window@5 [7]	unknown	23.1	10.0	-	-	14.0
F-PAD@1	randomSplit1	41.5	28.5	43.4	29.1	24.8
F-PAD@5	randomSplit1	50.8	34.2	51.6	34.6	28.1
F-PAD@1	controlled_randomSplit1	31.7	19.4	31.4	20.8	9.5
F-PAD@5	controlled_randomSplit1	37.9	23.5	39.0	24.1	10.3

overlapping with the pretraining classes have high per-class AP and thus bringing the overall mean AP up.

4. Conclusion

In this paper, we propose a few-shot temporal activity detection model based on proposal regression, which is composed of a temporal proposal subnet and a few-shot classification branch based on similarity values. Our model is end-to-end trainable and can benefit from observing additional few-shot examples. Besides, we also observe that the few-shot activity detection performance on novel classes could be affected by the class similarity between novel classes and pretrained classification classes, as well as base classes. Suggesting and coming up with fairer evaluation strategy for few-shot activity detection taking the class similarity into consideration will be a future work for this.

References

[1] Xuanyi Dong, Liang Zheng, Fan Ma, Yi Yang, and Deyu Meng. Few-example object detection with model communication. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1641–1654, 2018. 1

[2] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. ActivityNet: A Large-Scale Video Benchmark for Human Activity Understanding. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015. 2

[3] Y-G 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. http://crcv.ucf.edu/THUMOS14/, 2014. 2

[4] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8420–8429, 2019. 1

[5] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 1, 2, 3

[6] Huijuan Xu, Abir Das, and Kate Saenko. R-c3d: region convolutional 3d network for temporal activity detection. In *IEEE Int. Conf. on Computer Vision (ICCV)*, pages 5794–5803, 2017. 2

[7] Hongtao Yang, Xuming He, and Fatih Porikli. One-shot action localization by learning sequence matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1450–1459, 2018. 1, 2, 3, 4