What do I Annotate Next? An Empirical Study of Active Learning for Action Localization (Supplementary Material)

Fabian Caba Heilbron[1]*, Joon-Young Lee[2], Hailin Jin[2], and Bernard Ghanem[1]

[1] King Abdullah University of Science and Technology (KAUST), Saudi Arabia
[2] Adobe Research, San Jose, CA, USA
{fabian.caba,bernard.ghanem}@kaust.edu.sa
{jolee,hljin}@adobe.com
https://cabaf.github.io/what-to-annotate-next

Abstract. In this supplementary material, we complement our paper submission by providing additional information of the newly collected Kinetics-Localization dataset. We also provide additional insights into our proposed active learning framework and its performance in the wild.

Keywords: Video Understanding · Temporal Action Localization · Active Learning · Video Annotation

1 Insights on the Proposed Active Learning Framework

We provide additional insights into our proposed active learning framework. Here, experiments are conducted on Kinetics-Localization.

Which videos are selected? To better understand our active selection behavior, we plot the histogram of confidence scores of the selected videos at different learning steps in Figure 1. Interestingly, our active learning framework (using LAL [2] selection function) exhibits different selection behaviors depending on the localization model state. For instance, at early learning steps (*e.g.* 25% of training data), our learner tends to pick video instances with histograms presenting peak values at low prediction scores. Samples with such histogram can be interpreted as prototypical samples due to the majority of training videos contain a single (or very few) action instances. When the learner has observed 50% of the data, we observe that it switches its behavior and now looks similar to what uncertainty sampling does. Finally, at later stages, *i.e.* 75%, our selection function samples atypical videos, in this case, videos where multiple temporal segments have high confidence score.

*Work done during internship at Adobe Research

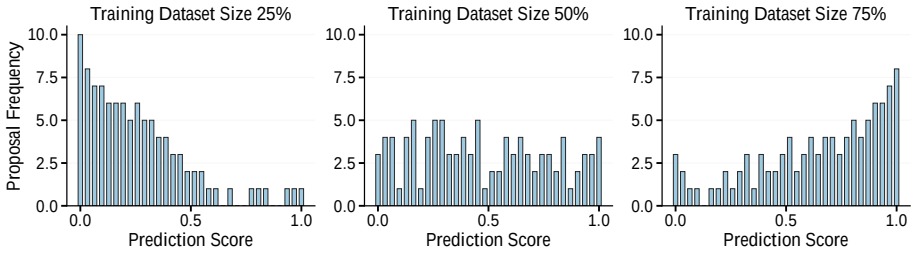


Fig. 1: **Video Selection Analysis:** We plot the histogram of the proposals' confidence scores from the selected video to be annotated. These confidence scores are obtained by applying our localization model to different temporal segments (or proposals). The proposed active learner shows different selection behaviors at different states of the localization model, *i.e.* at different training dataset sizes.

What type of error the learners fix? Our goal is to diagnose the type of errors our learner makes at different active learning stages. We decompose the false positive errors into two kinds: classification and localization errors. Classification errors happen when the predicted class is wrong. Localization Errors are composed of double detection predictions, and predictions with the right action class but with not enough tIoU to match a ground truth instance. Figure 2 shows the distribution of localization errors over multiple ratios of labeled dataset. We report the ratio of localization errors to total error. Interestingly, we notice that the localization error rate drops drastically after labeling few data. This results suggest that our method is selecting samples that help to improve the attention module.

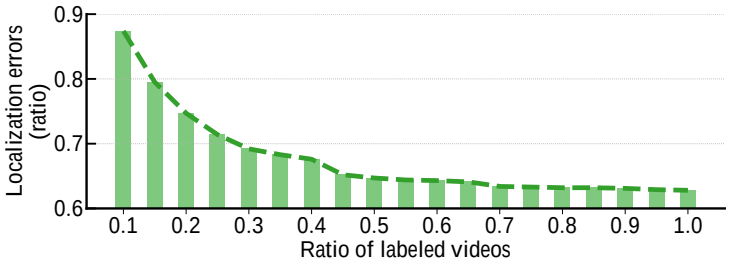
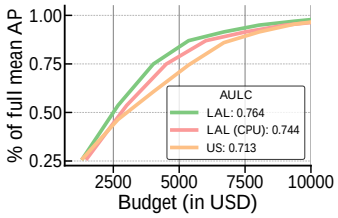


Fig. 2: **Localization Errors.** We report the ratio of localization errors at different ratios of labeled videos. Interestingly, the localization rate drops quickly with the adding of labeled samples. For example, at 25% of labeled data, the localization ratio drops by 15%. Abounding localization errors motivate our design choice of improving the attention module, which inherently improves the system's localization component.

	Time (Hours)	Cost (USD)
Computational Costs		
Inference	0.208	0.039
Re-training	0.583	0.108
Total Cost	0.791	0.147
Annotation Expenses		
1 Turker	0.166	1.208



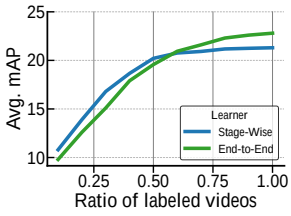


Fig. 3: **Left.** Computational and annotation costs summary per iteration step (per video). To estimate computational costs, we use the most recent AWS prices for a 4-CPU machine. We paid turkers the federal minimum hourly wage in the United States (7.25 USD). **Center.** Budget (in USD) against % of full mAP achieved by the learners. Interestingly, after adding computational expenses (LAL (CPU)), the proposed method remains an effective approach compared to Uncertainty Sampling (US). **Right.** Avg. mAP at different active learning steps (Kinetics-Localization). Notably, the Stage-Wise model provides a larger boost in performance at smaller ratios of labeled videos.

Computational and annotation cost analysis. We analyzed the effectiveness of LAL over uncertainty sampling (US). We compute the computational costs in USD for doing inference and re-training of the active learner (see Figure 3-Left). We also estimate the costs of hiring an Amazon Mechanical Turk worker (Tuker) to annotate a single video. Using that information, we re-plot Figure 3 in the paper, where the horizontal axis now represents the actual budget being spent, *i.e.* computational plus annotation expenses (see Figure 3-Center). We notice that the AULC achieved by LAL marginally decreases after taking into account the computational cost (LAL (CPU)). This result confirms that even after adding the computational cost overhead, the proposed Active Learner remains attractive for practical scenarios.

Stage-Wise against End-to-End. End-to-End models achieve better performance than staged approaches in many cases. However, this is evident when large-scale data is available to train the End-to-End models. To validate that argument, we present the performance of the Stage-Wise and End-to-End models in Figure 3-Right. On one hand, we observe that the staged model provides a more productive learning curve when small data is used for training. On the other hand, we see that the end-to-end performance is only superior after 55% of the dataset is available for training. Nevertheless, we argue that future works can develop end-to-end models that can gradually increase their capacity once data becomes available under an active learning setting.

Performance at different Kinetics-Localization sizes. Here, we aim to diagnose the performance of the proposed active learner when applied to collect Kinetics-Localization. We report the localization performance on Kinetics-Localization Validation Set at different percentages of the full training set size.

Table 1 summarizes the performance of the two localization models described in the main manuscript (Stage-Wise and End-to-End), using the annotated data gathered by our active learner. We notice significant boosts in performance when jumping from 10% to 25% dataset size, which suggests that our learnable selection is intelligently picking the right samples to annotate.

tIoU	Mean AP at dataset size									
threshold	10%	25%	50%	75%	100%	10%	25%	50%	75%	100%
	Stage-Wise					*End-to-End*				
0.05	30.4	53.7	63.3	70.5	72.1	34.5	57.6	68.8	70.2	72.8
0.25	25.8	39.8	46.2	51.8	54.5	26.1	38.7	44.9	51.6	55.0
0.5	16.2	27.3	35.1	42.8	45.1	15.9	25.9	32.8	44.7	49.6
0.75	15.4	19.7	23.2	24.0	24.5	13.1	16.8	23.0	24.8	26.1
0.95	0.9	2.2	2.7	3.0	3.1	0.4	1.8	2.9	3.7	4.4

Table 1: **Constructing Kinetics-Localization.** We report the mAP at different tIoU thresholds of the Stage-Wise and End-to-End [3] models. We note the localization performance on the test set increases once new data is added for training. Notably, the performance at higher tIoU thresholds improves significantly, which reaffirms the need for temporally annotated data to train models for temporal action localization.

2 Kinetics-Localization Annotation Details

Despite the arduous effort needed to build Kinetics [1], the dataset is not designed for the temporal localization task. As shown in the main paper, action detectors trained using the original dataset exhibit a poor localization performance, precisely at higher tIoU thresholds. This fact motivates us to build a new dataset, Kinetics-Localization, which contains precise temporal localization of actions. We fully annotate a portion of the Kinetics validation subset with temporally localized actions. Specifically, we annotate 3750 videos from 75 action categories. Additionally, we employ our active learner (described in the main paper) to gather temporal annotations of the Kinetics' training set. Using our method, we actively select the videos to be annotated by the turkers. We effectively compile a temporal action localization dataset comprising $15K$ videos from 75 different action categories, resulting in more than $30K$ temporal annotations.

To compile the Kinetics-Localization dataset, we develop a new video annotation system to localize actions in time. Our semi-automatic approach takes as input a video and a target action. Thus, it produces time intervals where the target action appears in the video. Purposely, we divide the annotation into two components: a *Localization Module*, which provides a pool of temporal intervals

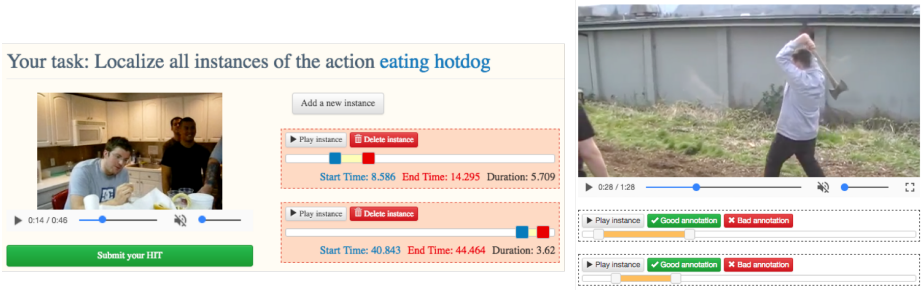


Fig. 4: **User Interfaces snapshots.** (left) User Interface to temporally localize actions of interest in video sequences. (right) User interface to verify the temporal annotations provided by the localization module.

that are likely to contain the action, and a *Verification Module*, which whether or not the segments truly contain the intended entity. Below, we describe each module in detail.

Localization Module. Given a video and a target action, this module's goal is to generate a set of temporal annotations where such target action occurs. The resulting temporal annotations are assumed to be of high recall, *i.e.* at least one of the produced segments contain the intended action. In practice, we rely on Amazon Mechanical Turk workers (Turkers) to review and to provide the video annotations. For completing each task, the turkers received \$0.3, and we submitted more than 20K tasks. We provide an online user interface that allows turkers to quickly scan the video and define starting and ending times of multiple actions (see Figure 4 (left)).

Verification Module. Once a set of temporal candidate segments is available for a video, the verification module inspects and selects segments that correctly match the target action. We also employ Turkers to conduct this task. Given that this task is simple and fast to complete, *i.e.* users only click whether or not the proposed segment is good, we assign each Turker 10 videos and pay \$0.1 for completing this task. Figure 4 (right) shows the user interface to verify the temporal annotations.

References

1. Kay, W., Carreira, J., Simonyan, K., Zhang, B., Hillier, C., Vijayanarasimhan, S., Viola, F., Green, T., Back, T., Natsev, P., et al.: The kinetics human action video dataset. arXiv preprint arXiv:1705.06950 (2017)
2. Konyushkova, K., Sznitman, R., Fua, P.: Learning active learning from data. In: Advances in Neural Information Processing Systems. pp. 4226–4236 (2017)
3. Zhao, Y., Xiong, Y., Wang, L., Wu, Z., Lin, D., Tang, X.: Temporal action detection with structured segment networks. In: ICCV (2017)