Harnessing Object Grounding for Time-Sensitive Video Understanding Supplementary Material

Tz-Ying Wu Sharath Nittur Sridhar Subarna Tripathi
Intel
{tz-ying.wu, sharath.nittur.sridhar, subarna.tripathi}@intel.com

The appendix is organized as follows. Section A summarizes the implementation details of GO-Video. Section B presents additional results with GO-LITA, and section C provides more ablations with GO-TimeChat.

A. Implementation Details

The code base is built atop the official code of TimeChat [8] and LITA [3] using PyTorch. We follow the hyperparameters used in their original papers if not specifically noted.

Architecture. For grounded object (GO) extraction, we uniformly sample F frames per video to run the object detection, with a maximum of k objects per frame having a minimum confidence score δ. Thus, the number of object tokens per video is bounded by $F \times k$. From our experiments, sampling with $F = 4$ and $k = 5$ can already achieve substantial improvement over the baseline without GO information. For example, GO-TimeChat and the vanilla TimeChat attain F1 scores of 16.1 and 11.3, respectively, on the YouCook-2 dataset. In this case, there are 20 object tokens at max per video, and this is less than the number of video tokens (i.e., 96). At inference time, two object detectors, Detic and YOLO-World, are considered. We adopt the Detic [10] model trained on combined LVIS, COCO, and ImageNet-21K datasets with Swin-B [6] architecture. For YOLO-World [1], we adopt the YOLO-Worldv2-L version based on the YOLOv8 architecture for both visual feature extraction and object detection. Note that the visual feature extraction only uses the backbone part of YOLO. We adopt LVIS [2] vocabulary for both detectors to prove the concept, which contains a wide range of classes (i.e., 1,203). However, since both detectors support open vocabulary, class names in the target domain can be directly applied to the detector if they are known.

Training. All the models in the paper are fine-tuned from the vanilla TimeChat-7B/LITA-13B model using 4 NVIDIA A100 GPUs. Newly introduced model parameters (for GO-Tokenizer and LoRA) are randomly initialized. We train the model in an end-to-end manner for three epochs. For TimeChat, the batch size is set to 8 and the initial learning rate is set to 3e-5. For LITA, the batch size is set to 4 with an initial learning rate of 1e-5. Due to GPU memory constraints, we adopt LoRA finetuning for all the LITA experiments and set the rank to 16.

Model	Inference w/ GO	mIOU
LITA	n/a	23.04
LITA	Text (*Class+Time+Bbox*) w/ GT	27.63
LITA	Text (*Class+Time+Bbox*) w/ YOLOW	27.91
GO-LITA	GO-Tokenizer w/ GT	27.80
GO-LITA	GO-Tokenizer w/ YOLOW	29.56

Table 1. Evaluating LITA performance on ActivityNet-RTL-GO with/without GO-Tokenizer.

Visual encoder Φ_v	# of params	CIDEr	SODAc	F1 Score
CLIP-ViT-B/16 [7]	71.6M	3.7	1.3	17.4
YOLO-World [1]	14.7M	**3.9**	**1.4**	**18.5**

Table 2. Ablations on the visual encoder of GO-Tokenizer.

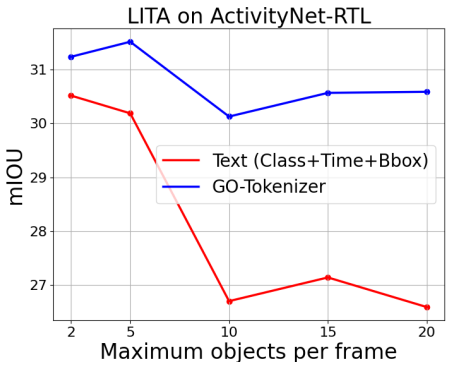

Figure 1. GO-LITA ablations on the number of objects per video frame (i.e. k) on the ActivityNet-RTL dataset with $F = 8$ using YOLOW during inference.

δ	# of objects/video	SODAc	F1 Score
0.1	39.9	1.3	18.1
0.3	39.5	**1.4**	**18.5**
0.5	34.3	**1.4**	18.4
0.7	18.3	1.3	14.7
0.9	2.7	0.8	6.3

Table 3. Ablations on the confidence threshold δ of the object detector. # of objects/video denotes the average number of objects that are actually detected and pass the confidence threshold per video.

Time embedding index	CIDEr	SODAc	F1 Score
0 for all frames	2.8	0.9	7.9
true timestamps	**3.9**	**1.4**	**18.5**

Table 4. Ablations on the time-aware embedding.

B. Additional Experiments with GO-LITA

Performance on ActivityNet-RTL-GO. Table 1 presents the RTL performance of LITA on the ActivityNet-RTL-GO dataset with/without GO information. We also compare GO-Tokenizer with Text (*Class+Time+Bbox*) using ground truth (GT) and YOLO-World detected (YOLOW) GO information. Since the objects are sparsely labeled in GT, both methods with YOLOW achieve better mIOU, while GO-LITA is consistently better.

Object count ablations on GO-LITA. We ablate the inference time object counts per frame (k) in Figure 1, while keeping the frame count (F) constant at 8. We evaluate on the ActivityNet-RTL dataset and show the variation in mIOU scores. Up to the top 20 objects were selected based on decreasing confidence score. GO-LITA consistently outperforms Text (*Class+Time+Bbox*), especially at higher object counts.

C. Additional Ablations with GO-TimeChat

This section presents additional ablations on GO-TimeChat components with zero-shot results reported on YouCook-2 [9]. Note that we adopt Detic [10] as the object detector with $F = 8$ and $k = 5$ in all experiments here.

Visual encoder of GO-Tokenizer. As discussed in section 5.1, GO-Tokenizer adopts the backbone of YOLO-World [1] as the visual encoder Φ_v, since it has a region-level image-text alignment, while the vanilla CLIP (CLIP-ViT-B/16) [7] only has an image-level one. Table 2 presents the comparison between the two visual features, showing that YOLO-World [1] achieves superior performance even with a lower number of parameters.

Model	GO Training	mAP	Hit@1
TimeChat	✗	10.9	15.2
GO-TimeChat w/ Detic	✓	12.5	17.2
GO-TimeChat w/ YOLOW	✓	**12.9**	**18.1**

Table 5. Evaluating performance of GO-TimeChat on QV Highlights [5] dataset with GO information extracted from Detic [10] or YOLOW-WORLD [1] during inference.

Model	Data for fine-tuning	CIDEr	SODAc	F1
TimeChat	ActivityNet-Captions	3.4	1.1	11.3
GO-TimeChat		4.4	1.4	18.5
Fine-tuned TimeChat	YouCook-2	10.1	3.1	18.4

Table 6. Comparison between the zero-shot and fine-tune performance evaluated on YouCook-2.

Object detection confidence threshold. We ablate the object detection confidence threshold δ in Table 3. A lower threshold implies a larger potential of getting false positive predictions, while a higher threshold may lead to more missing objects. However, since the number of objects is bounded by k per video frame (as described in section 5.2), the TSV performance is not highly sensitive to δ with a considerable range of confidence score (0.1 - 0.5) as shown in Table 3. We adopt $\delta = 0.3$ as it generates the best results.

Does time information matter? To verify the effectiveness of time-aware embedding $\mathbf{q}_i$, we did a sanity check by inferring the model with the time embedding of the first frame for all the frames, i.e. $\mathbf{q}_i = \mathbf{q}_0 \forall i$. Results in Table 4 present such an experiment and validate the importance of encrypting time information in the object tokens.

Generalization to video highlight detection. We further evaluate GO-TimeChat on the video highlight detection task using the QVHighlights [5] dataset shown in Table 5. This task requires a more detailed and fine-grained comprehension of videos at each frame, aiming to identify the time and associated saliency scores for highlight frames. We report the mAP and Hit@1 scores for this task. All models, including the baseline, were finetuned on ActivityNet-Captions [4]. Results show that the gain of integrating GO training to TimeChat generalizes to the video highlight detection task.

Fine-tune performance. In Table 6, we present the gap between the zero-shot (i.e., TimeChat and GO-TimeChat) and the fine-tuned performance on the YouCook-2 benchmark, where `Finetuned TimeChat` denotes the TimeChat model finetuned with the YouCook-2 dataset. The proposed model (second row) largely improves over the baseline model without GO information (first row) in all metrics, and reaches the fine-tune performance (last row) on

the F1 score, which is a metric for capturing the temporal localization performance.

References

[1] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 2

[2] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019. 1

[3] De-An Huang, Shijia Liao, Subhashree Radhakrishnan, Hongxu Yin, Pavlo Molchanov, Zhiding Yu, and Jan Kautz. LITA: Language instructed temporal-localization assistant. In *ECCV*, 2024. 1

[4] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. Dense-captioning events in videos. In *International Conference on Computer Vision (ICCV)*, 2017. 2

[5] Jie Lei, Tamara L. Berg, and Mohit Bansal. Qvhighlights: Detecting moments and highlights in videos via natural language queries. In *NeurIPS*, 2021. 2

[6] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 1

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, A. Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 1, 2

[8] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu Hou. Timechat: A time-sensitive multimodal large language model for long video understanding. In *CVPR*, 2024. 1

[9] Luowei Zhou, Chenliang Xu, and Jason J Corso. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence*, pages 7590–7598, 2018. 2

[10] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022. 1, 2