This WACV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Harnessing Object Grounding for Time-Sensitive Video Understanding

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

Abstract

We propose to improve the time-sensitive video understanding (TSV) capability of video large language models (Video-LLMs) with grounded objects (GO). We hypothesize that TSV tasks can benefit from GO within frames, which is supported by our preliminary experiments on LITA, a state-of-the-art Video-LLM for reasoning temporal localization. While augmenting prompts with textual descriptions of these object annotations improves the performance of LITA, it also introduces extra token length and susceptibility to the noise in object-level information. To address this, we propose **GO-Tokenizer**, *a lightweight add-on module for Video-LLMs leveraging off-the-shelf object detectors to encode compact object information on the fly. Experimental results demonstrate that pretraining with GO-Tokenizer outperforms the vanilla Video-LLM and its counterpart, utilizing textual descriptions of objects in the prompt. The gain generalizes across different models, datasets, and video understanding tasks, such as reasoning temporal localization and dense captioning.*

1. Introduction

Extending the ability of Large-Language-Models (LLMs) to the video modality is an emerging field that seeks to equip Video-LLMs [5, 22, 27, 44] with the ability to understand long and complex video, and respond to user queries for various downstream tasks, such as video summarization [32, 33] and video captioning [18, 48]. To leverage the knowledge embedded in pre-trained LLMs, most Video-LLMs learn video representations that share a latent space with text, allowing features from both modalities to be jointly processed during training and inference.

While Video-LLMs inherit the knowledge and reasoning capabilities from pretrained LLMs, the limitation of LLMs in modeling long sequences remains. This is particularly worrisome since a video usually contains several frames, and the spatial resolution of each frame could be large. Given the sequence length limitation, there is a clear trade-off between preserving temporal information (more frames) and spatial details (higher resolution). For example, to manage a reasonable input length, prior work [3, 22, 25, 44] compressed each frame into a single video token. While this might be sufficient for downstream tasks that only require global content like action classification [8, 17], it is inadequate for tasks that require fine-grained details, such as video dense captioning [48], highlights detection [20], temporal action localization [17], and video temporal grounding [10]. These types of tasks are time-sensitive video understanding (TSV) tasks, as they require detailed object-level information, including classes, attributes, and spatial relationships, to be preserved across frames.

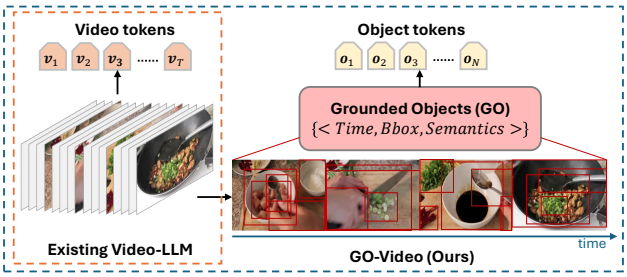


Figure 1. Video tokens in existing Video-LLMs [14] are highly compressed and compromise on spatial information. We harness GO information in sparsely sampled video frames as a supplement to improve the time-sensitive video understanding (TSV) tasks.

In this work, we investigate the task of TSV and hypothesize that the performance of TSV can be improved when the object-level information is preserved. Since recent Video-LLMs [5, 22, 27, 31, 44] do not harness such object-level information either during training or inference, we first conduct a preliminary study on this problem with an existing Video-LLM, namely LITA [14]. More specifically, we convert the object annotations in the ActivityNet-Captions [18] dataset into a textual description encompassing the object classes, locations, and the timestamps where the objects appear. These object descriptions across frames are then passed to the Video-LLM to perform downstream tasks. Initial results using this approach demonstrate an improvement in the model's performance, suggesting the need to integrate object information into existing Video-LLMs.

However, this gain comes with the cost of having an excessive amount of extra tokens. For example, when introducing the object description that covers the object class, bounding box, and timestamp of the objects, the model consumes around 1,400 additional tokens during inference, compared to vanilla LITA. Furthermore, the preliminary study also shows that the gain diminishes significantly when the text description contains inaccurate object classes or misaligned bounding boxes. This finding motivated us to investigate an effective and efficient method for utilizing object-level information in temporal understanding tasks.

To address this, we propose **GO-Tokenizer**, a lightweight module for existing Video-LLMs that encodes the object semantics, spatial positions, and temporal cues of the **grounded objects (GO)** in video frames into compact object tokens, as illustrated in Figure 1. For each object in a frame, we introduce ROI patch pooling to average the patch features within the given object bounding box, where the patch features combines a semantic-rich visual feature map from a pretrained encoder [4]) and location-aware positional embeddings. This pooling mechanism inherently captures the semantic and spatial information. The resulting feature is projected into a latent space shared with text tokens and enriched with a time-aware positional encoding to capture when the object appears. Each object is thus efficiently represented as a single object token, which is passed to the LLM alongside text and video tokens. The GO-Tokenizer module is seamlessly integrated with existing Video-LLMs, referred to as GO-Video models, which is optimized end-to-end with low-rank adaptation [12] applied in the LLM component. During inference, where GO annotations are unavailable, we uniformly sample sparse frames from the video, leveraging the redundancy in adjacent frames. We employ an off-the-shelf object detector [4, 50] to extract GO information in these frames, and encode them with GO-Tokenizer into object tokens. This strategy readily ensures precise temporal grounding.

To demonstrate the efficacy of the GO-Tokenizer, we integrate it into two leading Video-LLMs, TimeChat [31] and LITA [14], resulting models referred to as GO-TimeChat and GO-LITA respectively, where the former is evaluated on the dense video captioning task using YouCook-2 [48] dataset and the latter is evaluated on the task of reasoning temporal localization using the ActivityNet-RTL [14] dataset. In both tasks, models equipped with the GO-Tokenizer outperform (1) their vanilla version and (2) their vanilla version trained and inferred with object text descriptions. Additionally, we conduct ablation studies considering different configurations during inference, including the use of different object detectors and the number of objects per video frame. Under all configurations, the proposed GO-Tokenizer outperforms the baselines using a more robust and compact representation to encode object information.

Overall, this work investigates how to effectively and efficiently utilize GO information for time-sensitive video understanding, making the following three contributions. First, we demonstrate that, without modifying the underlying architecture, existing Video-LLMs can be improved by incorporating object information at varying levels of granularity, such as class labels and spatial grounding, for training and testing at the text prompt level. Second, we propose to utilize the object-level visual features from existing object detectors and train learnable grounded object (GO) tokens to efficiently encode object information. Finally, we show that the proposed GO-Tokenizer consistently outperforms the baselines on the TSV tasks when applied to two different Video-LLMs, LITA and TimeChat.

2. Related Work

Large Language Models for Video Understanding. The advent of large language models (LLM) pre-trained on extensive datasets notably increases the capability of handling various tasks using prompts without requiring fine-tuning. Several studies have been made about the utilization of LLMs in the computer vision field, such as LLaVA [23], GPT-4V[43], and Visual-ChatGPT [37]. LLMs integrated with video understanding opened up completely new ways of interacting with videos. These models are equipped with the power of complex multimodal reasoning, interacting with textual prompts in ways more natural for humans. In recent years, there has been a surge of models for such Video-LLM tasks. The survey paper [34] is an excellent reference to the Video-LLM field as a whole. Typically, most of these models utilize open-source LLMs like LLaMA[35] or Vicuna[6] as the backbone. Recent Video-LLM models significantly outperform previous methods on several video understanding tasks, ranging from video classification to video question-answering, with the aim of near-human level performance for video interpretation capability.

Time-Sensitive Video Understanding Tasks. Video understanding encompasses a range of tasks, each with distinct applications and benchmarks. These tasks can be broadly grouped into three categories. The first involves holistic understanding, including video classification [1], action recognition [19], multi-modal retrieval [9], video captioning [39], and video-to-textual summarization [13], etc. The second category, containing time-sensitive video understanding (TSV) tasks, focuses on temporal understanding and includes tasks such as video highlights [20], condensing long video to short duration [29], temporal action or event localization [28], moment retrieval [26], dense video captioning [49]. Finally, the third category centers on spatial-temporal understanding and is associated with object-level tracking [36], re-identification [42], segmenting video objects [41], etc. This work concentrates on the

second category, which is TSV. While objects provide valuable cues for interpreting scene semantics, we observe that state-of-the-art Video-LLM models [5, 15, 21, 25, 27, 31, 44] developed for TSV do not explicitly leverage such information. To assess the utility of grounded object information for this task, we augment existing Video-LLMs with textual descriptions of detected objects, and further propose a token-efficient representation strategy for these objects, leading to improved performance on TSV tasks.

Large Multimodal Models with Object Information. The core idea in Multimodal Large Language Models (MLLMs) lies in vision tokenization, which transforms visual signals into feature representations apt for pre-trained LLMs. Recent approaches for MLLMs introduced several ways to leverage explicit entity-level information for image-based reasoning tasks. RegionGPT [11] enables users to specify a region of interest, and such placeholders are subsequently replaced with semantic region-level embeddings that are later consumed by the language model. VCoder [16] explores off-the-shelf object perception modalities such as segmentation or depth maps as additional control signals and projects this information into the LLM input space. Another recent work, MG-LLaVA [46], introduced concept-level image-text alignment via semantically equivalent visual tokens. While these methods focus on visual tokenization strategies for image-level reasoning, they design tokenization schemes tailored to their specific tasks with static images rather than videos, where temporal dynamics pose critical challenges. Although GO-Tokenizer is also designed to be integrated into a pretrained LLM, it is tailored for video understanding, especially for TSV tasks.

3. Preliminaries

Video-LLM architecture. LLMs have been widely adapted to different modalities due to their established efficacy in sequence modeling and remarkable performance in predicting text tokens. In the visual modality, a popular approach is to represent an image with a series of visual patch tokens [7], and these visual tokens are projected to a latent space that is shared with the text tokens [23]. The LLM then ingests the combined text tokens and visual tokens and generates text tokens for various downstream tasks. Similarly, video large language models (Video-LLMs) extract video tokens with a video encoder (e.g., Video-Qformer [31] or ViT [7]) from the given video $\mathbf{V} \in \mathbb{R}^{T\times H\times W\times 3}$, where T denotes the number of frames, and H, W are the spatial dimensions of the video frame. In summary, Video-LLMs consume two types of input tokens, video tokens $\mathbf{v}_{1\ldots K}$ and text tokens $\mathbf{t}_{1\ldots L}$, with K and L denoting the token length of the respective modality.

Advancing Video-LLMs for TSV. Time-sensitive video understanding (TSV) capability of Video-LLMs is typically empowered by instruction tuning [23, 45] the model on large, annotated video datasets with temporal labels and event-related queries. This enables the model to understand and predict the temporal relationships between events, such as when they start, end, or overlap. For instance, in event localization, the instruction could be *"When does [event] happen in the video? Answer the question only using start and end timestamps,"* with the model providing the corresponding timestamps. Similarly, for dense video captioning, the instruction might be *"Detect and report the start and end timestamps of activity events in the video, along with descriptions."*, with the model generating a series of narrations with their timestamps. Since the output is natural language, the model is usually trained using next-token prediction as its optimization objective.

4. Motivation: Can object grounding in Video-LLMs help TSV?

Due to computational limitations and the complexity of modeling long sequences, the existing LLMs can only process a finite number of tokens, and such a constraint leads to the trade-off between spatial and temporal resolutions [38, 40] for Video-LLMs. For example, to model the long-sequence video (i.e., large T), the size of each video frame is reduced (i.e., small H, W), which leads to fewer video tokens per frame. Prior works [21, 24, 44] even represent each frame with only one video token (i.e., $K = T$). Since the compression in spatial dimension indicates the loss of fine-grained spatial details, this hinders the Video-LLMs from excelling in time-sensitive downstream tasks that require fine-grained information in each frame, such as reasoning temporal localization [14] and dense video captioning [48]. Recently, LITA [14] introduced the *Slow-Fast* tokens for this trade-off, with *Slow* and *Fast* tokens that favor the spatial and time resolution, respectively.

In this paper, we explore an orthogonal direction, augmenting the highly condensed video tokens on a more abstract level. We hypothesize that leveraging the sparse but informative **grounded objects (GO)** in video frames could improve the performance of TSV tasks. We aim to answer two questions: (1) Can providing GO information in Video-LLMs enhance TSV? (2) What levels of GO information are needed? Object grounding in the wild, however, may contain some noise in the class label and bounding box location space. To disentangle this factor from our hypothesis, we construct a subset of ActivityNet-RTL [14] evaluation set where object entity annotations (e.g., object class, bounding box location, etc.) are available from [47], namely **ActivityNet-RTL-GO**. The dataset sparsely labels an average of 10.43 objects per video and 2.41 objects per frame, with object labels present in 4.45 frames on aver-

GO at Inference	mIOU	P@0.5
n/a	23.04	18.78
Text *(Class)*	23.65	18.08
Text *(Class+Time)*	26.47	20.47
Text *(Class+Time+Bbox)*	27.63	23.71

Table 1. Using different levels of ground truth GO information as context to evaluate LITA-13B [14] on ActivityNet-RTL-GO.

Flip Ratio	mIOU	P@0.5
10%	27.07	23.25
20%	25.89	22.63
50%	24.63	20.13

Table 2. Flipping a percentage of object class labels in each video into randomly sampled classes.

Shift	mIOU	P@0.5
$0.01 \times (H, W)$	27.37	21.13
$0.02 \times (H, W)$	26.42	19.01
$0.05 \times (H, W)$	24.71	17.14

Table 3. Shifting bounding box locations by varying amounts with respect to the image height (H) and width (W) in pixels.

age. In this section, we conduct empirical studies on these questions using LITA [14], a state-of-the-art Video-LLM for TSV, and evaluate its temporal localization performance on the ActivityNet-RTL-GO dataset, with different levels of GO information provided at inference time.

4.1. Augmenting Video-LLMs with GO information

Object grounding in videos involves several dimensions, including the object category *(Class)*, the timestamp where the objects are visible *(Time)*, and the spatial locations in the frames *(Bbox)*. To incorporate GO into Video-LLMs, we encode this information into text descriptions as additional context prepended to the text instructions during inference. We provide GO information in various granularities to Video-LLMs as distinct variants to examine which levels are essential for temporal understanding, i.e.,

- **Class:** *"<Obj> Objects in this video are: man, window, ...</Obj>"*.
- **Class+Time:** *"<Obj> Each object is provided with its timestamp and class label in the format of <time, class label>. Here are the objects: <91.2 second, man>, ...</Obj>"*.
- **Class+Time+Bbox:** *"<Obj> Each object bounding box is provided with its timestamp and class label in the format of <time, (x1,y1,x2,y2), class label>. Here are the objects: <91.2 second, (0.0001, 0.1715, 0.0806, 0.3784), man>, ...</Obj>"*.

Table 1 presents the temporal localization performance of LITA with these variants, with the first row showing the baseline performance without any GO information provided. The mIOU calculates the average intersection-over-union (IOU) between the predicted and ground truth time segments, whereas P@0.5 measures precision at a 0.5 IOU threshold. When introducing GO information, the results demonstrate a noticeable improvement over the baseline, with greater gains observed as more fine-grained GO information is provided. This supports the hypothesis that augmenting video tokens with GO can improve the TSV capability of Video-LLMs. However, assuming perfect object grounding at test time is unrealistic.

4.2. Analysis of Object Grounding Perturbations

To evaluate the robustness of incorporating GO information as text instructions, we simulate the perturbations in object grounding by introducing noise to ground truth class labels and bounding box locations. Perturbed GO information is then applied to the *Class+Time+Bbox* approach from the previous section. Note that time annotation is always precise during inference, as object grounding is obtained by running object detection on the sampled video frames. We elaborate the perturbations considered below.

Class flipping. This operation randomly selects $x\%$ of the objects per video, and replaces their class labels with randomly chosen labels from the dataset, simulating classification errors in object detection. We report the average performance for three random seeds.

Bounding box shift. The object localization error is simulated by shifting ground truth bounding box positions by a percentage of the image's height and width, allowing control over the noise in object detection.

Table 2 and Table 3 show that providing GO information simply as text is not robust to the perturbations in object grounding, as performance degrades at higher noise levels. Another side effect of this approach is the inefficiency in the token length for representing an object. For instance, *Class+Time+Bbox* requires roughly 42 additional tokens per object, with the total overhead increasing proportionally to the number of objects. This raises the question of what is the most effective and generalizable manner to represent the GO information, so that it can generalize across visual domains while keeping the input sequence to Video-LLMs within a manageable length. Inspired by the preliminary experiments, we introduce the notion of object tokens $\mathbf{o}_{1...N}$, representing the GO information for N objects at a more abstract level. This is a new type of input token in Video-LLMs to supplement the video and text tokens. The details of the GO tokenization are presented in the next section.

5. Grounded Object Tokenization

To encode GO information with a more efficient and generalizable representation to Video-LLMs, we propose **GO-Tokenizer** as a lightweight add-on module to abstract the GO information into object tokens. We refer to the Video-LLM architecture supplemented with GO-Tokenizer as **GO-Video**, as illustrated in the left of Figure 2.

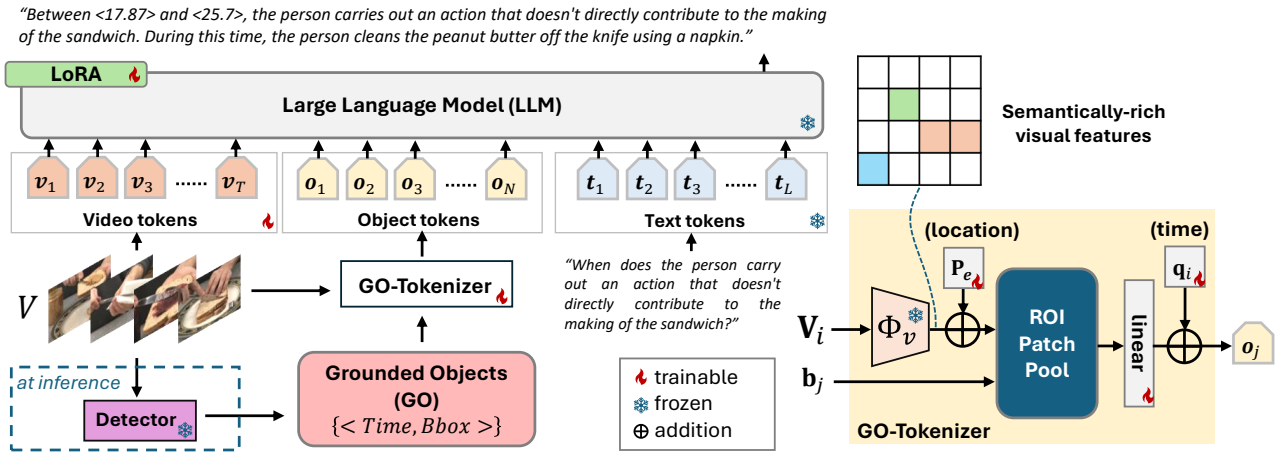


Figure 2. (Left) Model architecture of GO-Video. The LLM input space is augmented with the object tokens extracted by the GO-Tokenizer. GO information is extracted with off-the-shelf object detectors during inference. (Right) GO-Tokenizer extracts the object semantics, locations and time information of each object into a single object token.

5.1. Architecture

Given an image $\mathbf{V}_i \in \mathbb{R}^{H\times W\times 3}$ sampled at i-th video frame and a bounding box coordinate $\mathbf{b}_j \in \mathbb{R}^4$ associated with an object within the image, GO-Tokenizer extracts an object token $\mathbf{o}_j \in \mathbb{R}^{d_t}$, which contains the object semantics, bounding box locations and time information in a single token. More specifically, given an image $\mathbf{V}_i \in \mathbb{R}^{H\times W\times 3}$, we leverage the YOLO-World backbone [4] as a pre-trained visual encoder Φ_v, to extract the frame-level feature map. YOLO-World is selected because it is a lightweight CNN-based architecture and it is trained with region-based image-text alignment using the text encoder from CLIP [30]. Note that other feature extractors like the ViT in vanilla CLIP are also feasible, while it does not have image-text alignment at the region level but the image level [30], which tends to lead to inferior performance (as shown in the appendix). To encode the location information, we add a patch positional embedding $\mathbf{P}_e$ to the frame-level feature map extracted by Φ_v, i.e.

$$\mathbf{F}_i = \Phi_v(\mathbf{V}_i) + \mathbf{P}_e \tag{1}$$

where $\Phi_v(\mathbf{V}_i), \mathbf{P}_e \in \mathbb{R}^{N_p\times N_p\times d_v}$, N_p denote the number of patches, and d_v the dimension of the visual feature. Note that we keep Φ_v frozen in both training and inference.

To acquire semantically-rich object features, we use the given object bounding box $\mathbf{b}_j$ to crop the ROI (region of interest) in the frame-level feature map $\mathbf{F}_i$. This is implemented with a *ROI-Patch-Pool*$(\cdot)$ that extracts the object ROI feature $\mathbf{h}_j$ by averaging over all the patches within the patches covered by the bounding box, as illustrated in Figure 3. The *ROI-Patch-Pool* operation can be formulated as

$$\mathbf{h}_j = \textit{ROI-Patch-Pool}(\mathbf{F}_i, \mathbf{b}_j). \tag{2}$$

Since we are performing a pooling operation in the feature map, the architecture becomes more robust to minor errors in bounding box predictions. Finally, to project the ROI feature to the same latent space of the text tokens, a linear projection layer $\mathbf{W}_o \in \mathbb{R}^{d_v\times d_t}$ is trained and a time-aware positional embedding $\mathbf{q}_i$ is added, i.e.,

$$\mathbf{o}_j = \mathbf{W}_o^T \mathbf{h}_j + \mathbf{q}_i, \tag{3}$$

where $\mathbf{q}_i \in \mathbb{R}^{d_t}$ represents the i-th frame and $\mathbf{o}_j$ is the object token that represents an object in the i-th frame. The overall architecture of GO-Tokenizer is summarized in the right of Figure 2.

5.2. Training and Inference

To demonstrate the generality of the GO-Tokenizer, we integrate it to two state-of-the-art Video-LLMs tuned for TSV tasks, LITA [14] and TimeChat [31], resulting in GO-LITA and GO-TimeChat, respectively. While they employ different model architectures to extract video tokens (e.g., ViT and Q-Former), GO-Tokenizer remains a versatile module applicable to different Video-LLM models.

For each GO-Video model (i.e., GO-LITA/GO-TimeChat), the model extracts video tokens and text tokens following the original setting, while extracting object tokens with GO-Tokenizer. The LLM consumes the concatenation of three types of input tokens

$$\mathbf{v}_{1\ldots K} \oplus \mathbf{o}_{1\ldots N} \oplus \mathbf{t}_{1\ldots L} \tag{4}$$

where $\oplus$ denotes sequence concatenation. We optimize the Video-LLM with GO-Tokenizer (with trainable parameters: $\mathbf{W}_o$, $\mathbf{q}$, $\mathbf{P}_e$) in an end-to-end manner with the language modeling loss in their original settings. Low-rank adaptation [12] is applied to the LLM component, keeping the

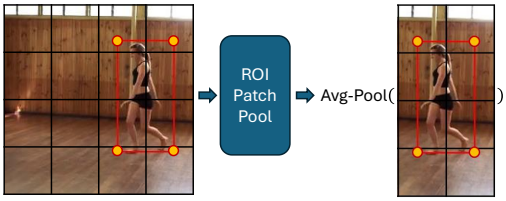


Figure 3. **ROI-Patch-Pool.** All the patches covered by the bounding box are average-pooled.

majority of its parameters fixed. Detailed settings for each model are elaborated in the experiment section.

During inference, we extract GO information on-the-fly by applying an off-the-shelf object detector [4, 50] on sparse frames uniformly sampled from the video. To reduce noisy predictions, we set a confidence threshold δ and only keep the top-k object predictions above that threshold. Thus, for a video with F sampled frames, the number of object tokens N is bounded by $F \times k$. In this paper, we set $F = 8$ and $k = 5$ if not specifically noted. The extracted GO information is then encoded by the GO-Tokenizer into object tokens.

6. Experiments

6.1. Experimental Setup

GO-LITA. We obtain object annotations from ActivityNet-Entities [47] for both ActivityNet-RTL [14] and ActivityNet-Captions [18]. We finetune the LITA-13B model jointly with the former dataset on reasoning temporal localization (RTL) and the latter on dense video captioning and event localization tasks. The ActivityNet-RTL [14] dataset comprises 10,009/160 training/evaluation videos with 33,557/229 RTL question-answer pairs generated by GPT-4. The RTL task demands advanced reasoning of the video content to determine the event's timing and offer an explanation. We evaluate the model on the full ActivityNet-RTL [14] dataset and provide the results on ActivityNet-RTL-GO in the appendix. We report the mIOU and P@0.5 metrics for temporal localization. For the explanation evaluation, since we don't have access to the GPT-4 API, we evaluate the relative score (Rel. Score) metric [14] with Llama-3-8B-Instruct [2].

GO-TimeChat. Similar to the GO-LITA setting, we enrich ActivityNet-Captions [18] with bounding box annotations [47] and finetune the vanilla TimeChat model with the dense video captioning task. We evaluate our approach on the YouCook-2 dataset [48], which comprises approximately 1,790 raw cooking video clips. The model is tasked with identifying around eight essential cooking steps and generating descriptions that accurately reflect the visual content. We use only the validation portion of the dataset for the zero-shot evaluation and report performance using CIDEr, F1 score, METEOR, and SODAc metrics.

Model	GO Training	mIOU	P@0.5	Rel. Score
LITA	✗	28.28	26.40	54.7
LITA + Text (*Class+Time+Bbox*)	✗	28.07	25.83	54.3
Finetuned on ActivityNet-Captions & ActivityNet-RTL				
LITA	✗	28.72	24.53	54.7
LITA + Text (*Class+Time+Bbox*)	✗	28.82	25.10	55.6
GO-LITA	✓	**31.52**	**28.07**	**59.0**

Table 4. Evaluating LITA-13B on the ActivityNet-RTL dataset with/without GO-Tokenizer. YOLO-World [4] is adopted at inference time to extract the GO information.

Model	GO Training	CIDEr	SODAc	F1
Valley [25]	✗	0.0	0.1	1.5
Video-LLaMA [44]	✗	0.0	0.0	0.1
TimeChat [31]	✗	3.4	1.2	12.6
Finetuned on ActivityNet-Captions				
TimeChat	✗	3.4	1.1	11.3
TimeChat + Text (*Class+Time+Bbox*)	✗	2.4	0.8	10.6
GO-TimeChat	✓	**3.9**	**1.4**	**18.5**

Table 5. Zero-shot evaluation on YouCook-2 [48] dataset for dense video captioning. GO information is extracted with Detic [50] at inference. All the LLMs have approximately 7B parameter size.

Implementation. We build the codebase atop LITA [14] and TimeChat [31] for GO-LITA and GO-TimeChat, respectively. All experiments were trained with 4 NVIDIA-A100 GPUs. More details are reported in the appendix.

6.2. Comparisons to Vanilla Video-LLMs

Table 4 and Table 5 present the main results of utilizing GO-Tokenizer to encode object-level information, compared to the vanilla Video-LLMs and their text-based baselines. We evaluate two state-of-the-art Video-LLM models for the TSV task, LITA [14] and TimeChat [31], whose original models were trained on broader datasets (e.g., TimeIT [31]) alongside ActivityNet. For fair comparison, we fine-tune all models exclusively using ActivityNet, as it provides the necessary object entity annotations. The upper blocks of the tables report the performance of the published checkpoints for reference. Note that only GO-LITA and GO-TimeChat leverage the bounding box annotations during training. Compared to vanilla LITA or TimeChat, introducing GO-Tokenizer to these models significantly boosts the performance on all the metrics across datasets and models. Interestingly, vanilla TimeChat shows no improvement when GO information is provided as text (*Class+Time+Bbox*) during inference. This may be attributed to its zero-shot evaluation setting or limitations in context length, as encoding object-level details in text is token-inefficient. LITA, with twice the context length of TimeChat, is less affected by this constraint. In contrast, GO-Tokenizer offers a more scalable and generalizable approach to integrating object-level information to Video-LLMs.

Model	Go Training	Tokens/GO	CIDEr	SODAc	F1
TimeChat	✗	0	3.4	1.1	11.3
TimeChat + Text (*Class*)	✓	4	1.8	0.8	16.2
TimeChat + Text (*Class+Time*)	✓	12	1.8	0.9	16.8
TimeChat + Text (*Class+Time+Bbox*)	✓	42	1.6	0.8	13.0
GO-TimeChat	✓	1	**3.9**	**1.4**	**18.5**

Table 6. Comparisons to baselines that represent GO as text descriptions with TimeChat. Detic [50] is used to extract GO information at inference.

Model	Go Training	mIOU	P@0.5	Rel. Score
LITA	✗	28.72	24.53	54.7
LITA + Text (*Class+Time+Bbox*)	✓	30.05	27.97	55.5
GO-LITA	✓	**31.52**	**28.07**	**59.0**

Table 7. Comparisons to baselines that represent GO as text descriptions with LITA. YOLO-World [4] is used to extract GO information at inference.

Model	Object detector	Go Training	CIDEr	F1
TimeChat	n/a	✗	3.4	11.3
TimeChat + Text (*Class+Time+Bbox*)	Detic	✓	1.6	13.0
GO-TimeChat	Detic	✓	3.9	18.5
TimeChat + Text (*Class+Time+Bbox*)	YOLO-World	✓	1.7	14.3
GO-TimeChat	YOLO-World	✓	4.4	18.5

Table 8. Ablations on object detectors. Zero-shot results on YouCook-2. The performance of the GO-Tokenizer generalizes across the selection of object detectors.

6.3. Comparisons to Training with GO as Text

Since the results of both GO-LITA and GO-TimeChat have shown improvements on TSV tasks when finetuning Video-LLMs with ground truth object annotations, it is also feasible to incorporate GO information as text descriptions in both finetuning and testing phases, using the formulation described in section 4. We implement these variants as baselines, denoted as [Video-LLM] + Text (*Class*/*Class+Time*/*Class+Time+Bbox*). Note that all these baselines follow the same training/evaluation setting as GO-LITA and GO-TimeChat, respectively.

Table 6 compares the zero-shot performance when the models are trained on ActivityNet-Captions and evaluated on YouCook-2. While the temporal localization performance (F1) improves for some baselines, the caption quality (CIDEr, SODAc) decreases drastically, possibly because these models are not resilient to object detection noise, as shown in section 4, or the excessive tokens introduced by the GO text description exceeds the context length limitation. Note that the number of additional tokens allocated per GO in these baselines is much higher than in GO-Tokenizer, which increases proportionally with the object attributes considered (i.e. the token length of *Class+Time* is greater than that of *Class*). Given that the model's maximum sequence length remains fixed, this may prohibit users from issuing a more complex query. For example, sampling 16 frames per video and assuming only one object per frame would consume 42×16 tokens. In contrast, GO-Tokenizer abstracts the object information into a single token, offering the scalability as there are more objects in each frame.

Table 7 presents the comparison of GO-LITA to the most informative baseline (*Class+Time+Bbox*). Unlike TimeChat, the temporal localization metrics (mIOU and P@0.5) show improvement with both the text-based baseline and the GO-Tokenizer. GO-LITA further enhances the improvement over the baseline on the Rel. score metric.

6.4. Ablation Studies

Object detector. Two different object detectors are considered in this paper, including Detic [50] and YOLO-World [4]. As shown in Table 8, the proposed GO-Tokenizer consistently improves over the text prompt approaches (baselines in section 6.3) and the vanilla Video-LLM, regardless of what object detector is being used.

Number of frames. To investigate the impact of varying the number of frames containing object information (i.e., F), we perform ablation experiments with GO-TimeChat on YouCook-2 [48], as shown in Figure 5. For these experiments, we keep the number of objects per frame (k) fixed to one. The trends in Figure 5 clearly indicate that increasing the value of F leads to improvements across all the metrics for dense video captioning. In addition, thanks to the compact representation of GO-Tokenizer, scaling to a larger number of frames can be easily achieved.

Number of objects. We conduct additional ablation experiments on YouCook-2 [48] with GO-TimeChat to understand the impact of varying the number of objects per frame (k) while keeping the frame count (F) constant at 8. As illustrated in Figure 6, we observe that increasing the object count per frame initially improves the accuracy metrics. However, beyond a certain threshold, the metrics either plateau or begin to decline. The best performance is achieved when there are 4 to 5 objects in the frame. We hypothesize that this drop in accuracy is due to a large number of input prompt tokens, which makes it more challenging for the model to process effectively. Additionally, with more objects, the confidence of the object detector also decreases, which could negatively impact the scores as well.

6.5. Qualitative Result

Figure 4 presents a visualization of zero-shot dense captioning results on YouCook-2 associated with different ways of utilizing GO information. The example shows an instructional video for cooking stir-fried chicken. As shown in the second row, the class names detected by the YOLO-World detector [4] are not always correct (for example, the crawfish is incorrect). This can be due to the vocabulary change



Figure 4. Qualitative results on YouCook-2. The first row shows the frames sampled from the video, the second row shows the objects detected by YOLO-World, and the last two rows show the predicted dense captions from the baseline model with text-based object tokens and with the tokens from GO-Tokenizer, respectively.

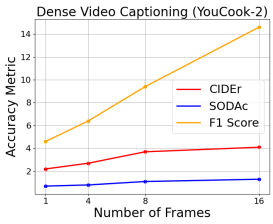


Figure 5. Ablation on the number of frames per video (i.e. F). Results of GO-TimeChat on YouCook-2 with 1 object per frame using Detic at inference.

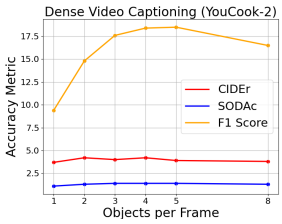


Figure 6. Ablation on the number of objects per video frame (i.e. k). Results of GO-TimeChat on YouCook-2 with $F = 8$ using Detic at inference.

between the object detector and the target video dataset, which can introduce a lot of noise to the LLM input. However, GO-Tokenizer does not rely on the decoded class labels and can tolerate minor bounding box drift by its design. Therefore, it generates captions with better quality compared to the baseline models with text prompts.

7. Conclusions

This work focuses on time-sensitive video understanding (TSV), which requires discerning not only the type of event but also its temporal context, making fine-grained scene comprehension essential for accurate interpretation. We hypothesize that incorporating grounded object (GO) information can enhance TSV tasks and validate this by injecting textual descriptions of objects during inference. However, adding such descriptions for every object across all frames is inefficient and susceptible to prediction noise. To overcome these limitations, we propose GO-Tokenizer, which encodes GO information at a more abstract level. The resulting Video-LLM, integrated with GO-Tokenizer, is referred to as GO-Video and is fine-tuned on the target dataset using low-rank adaptation. Experiments demonstrate that GO-Tokenizer can be applied to various state-of-the-art Video-LLMs (e.g., LITA and TimeChat), with GO-Video outperforming vanilla Video-LLMs on ActivityNet-RTL and YouCook-2. Extensive ablation studies further reveal that GO-Tokenizer offers a more token-efficient and generalizable approach to encoding object information.

References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 2

[2] AI@Meta. Llama 3 model card, 2024. 6

[3] Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. Videollm-online: Online video large language model for streaming video. In *CVPR*, 2024. 1

[4] 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. 2, 5, 6, 7

[5] Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, and Lidong Bing. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms, 2024. 1, 3

[6] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E Gonzalez, et al. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. *See https://vicuna. lmsys. org (accessed 14 April 2023)*, 2(3):6, 2023. 2

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021. 3

[8] Bernard Ghanem Fabian Caba Heilbron, Victor Escorcia and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015. 1

[9] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 214–229. Springer, 2020. 2

[10] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query, 2017. 1

[11] Qiushan Guo, Shalini De Mello, Hongxu Yin, Wonmin Byeon, Ka Chun Cheung, Yizhou Yu, Ping Luo, and Sifei Liu. Regiongpt: Towards region understanding vision language model. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024. 3

[12] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 2, 5

[13] Hang Hua, Yunlong Tang, Chenliang Xu, and Jiebo Luo. V2xum-llm: Cross-modal video summarization with temporal prompt instruction tuning. *ArXiv*, abs/2404.12353, 2024. 2

[14] 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, 2, 3, 4, 5, 6

[15] 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. 3

[16] Jitesh Jain, Jianwei Yang, and Humphrey Shi. VCoder: Versatile Vision Encoders for Multimodal Large Language Models. In *CVPR*, 2024. 3

[17] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset, 2017. 1

[18] 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. 1, 6

[19] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. Hmdb: a large video database for human motion recognition. In *2011 International conference on computer vision*, pages 2556–2563. IEEE, 2011. 2

[20] Jie Lei, Tamara L. Berg, and Mohit Bansal. Qvhighlights: Detecting moments and highlights in videos via natural language queries. In *NeurIPS*, 2021. 1, 2

[21] KunChang Li, Yinan He, Yi Wang, Yizhuo Li, Wenhai Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. Videochat: Chat-centric video understanding, 2024. 3

[22] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection, 2024. 1

[23] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023. 2, 3

[24] Ruipu Luo, Ziwang Zhao, Min Yang, Junwei Dong, Da Li, Pengcheng Lu, Tao Wang, Linmei Hu, Minghui Qiu, and Zhongyu Wei. Valley: Video assistant with large language model enhanced ability, 2023. 3

[25] Ruipu Luo, Ziwang Zhao, Min Yang, Junwei Dong, Minghui Qiu, Pengcheng Lu, Tao Wang, and Zhongyu Wei. Valley: Video assistant with large language model enhanced ability, 2023. 1, 3, 6

[26] Kaijing Ma, Xianghao Zang, Zerun Feng, Han Fang, Chao Ban, Yuhan Wei, Zhongjiang He, Yongxiang Li, and Hao Sun. Llavilo: Boosting video moment retrieval via adapter-based multimodal modeling. In *2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 2790–2795, 2023. 2

[27] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models, 2024. 1, 3

[28] Long Qian, Juncheng Li, Yu Wu, Yaobo Ye, Hao Fei, Tat-Seng Chua, Yueting Zhuang, and Siliang Tang. Momentor: Advancing video large language model with fine-grained

temporal reasoning. *ArXiv*, abs/2402.11435, 2024. 2

[29] Zhaofan Qiu, Ting Yao, Yan Shu, Chong-Wah Ngo, and Tao Mei. Condensing a sequence to one informative frame for video recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16311–16320, 2021. 2

[30] 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. 5

[31] 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, 2, 3, 5, 6

[32] Aidean Sharghi, Jacob S. Laurel, and Boqing Gong. Query-focused video summarization: Dataset, evaluation, and a memory network based approach, 2017. 1

[33] Yale Song, Jordi Vallmitjana, Amanda Stent, and Alejandro Jaimes. Tvsum: Summarizing web videos using titles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1

[34] Yunlong Tang, Jing Bi, Siting Xu, Luchuan Song, Susan Liang, Teng Wang, Daoan Zhang, Jie An, Jingyang Lin, Rongyi Zhu, Ali Vosoughi, Chao Huang, Zeliang Zhang, Pinxin Liu, Mingqian Feng, Feng Zheng, Jianguo Zhang, Ping Luo, Jiebo Luo, and Chenliang Xu. Video understanding with large language models: A survey, 2024. 2

[35] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023. 2

[36] Hang Wang, Yanjie Wang, Yongjie Ye, Yuxiang Nie, and Can Huang. Elysium: Exploring object-level perception in videos via mllm. In *European Conference on Computer Vision*, 2024. 2

[37] Chenfei Wu, Sheng-Kai Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *ArXiv*, abs/2303.04671, 2023. 2

[38] Wenhao Wu. Freeva: Offline mllm as training-free video assistant, 2024. 3

[39] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5288–5296, 2016. 2

[40] Mingze Xu, Mingfei Gao, Zhe Gan, Hong-You Chen, Zhengfeng Lai, Haiming Gang, Kai Kang, and Afshin Dehghan. Slowfast-llava: A strong training-free baseline for video large language models, 2024. 3

[41] Cilin Yan, Haochen Wang, Shilin Yan, Xiaolong Jiang, Yao Hu, Guoliang Kang, Weidi Xie, and Efstratios Gavves. Visa: Reasoning video object segmentation via large language models. *arXiv preprint arXiv:2407.11325*, 2024. 2

[42] Shan Yang and Yongfei Zhang. Mllmreid: Multimodal large language model-based person re-identification. *ArXiv*, abs/2401.13201, 2024. 2

[43] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v(ision), 2023. 2

[44] Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023. 1, 3, 6

[45] Yuanhan Zhang, Bo Li, haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. Llava-next: A strong zero-shot video understanding model, April 2024. 3

[46] Xiangyu Zhao, Xiangtai Li, Haodong Duan, Haian Huang, Yining Li, Kai Chen, and Hua Yang. Towards semantic equivalence of tokenization in multimodal llm. In *ICLR*, 2025. 3

[47] Luowei Zhou, Yannis Kalantidis, Xinlei Chen, Jason J Corso, and Marcus Rohrbach. Grounded video description. In *CVPR*, 2019. 3, 6

[48] 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. 1, 2, 3, 6, 7

[49] Xingyi Zhou, Anurag Arnab, Shyamal Buch, Shen Yan, Austin Myers, Xuehan Xiong, Arsha Nagrani, and Cordelia Schmid. Streaming dense video captioning. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18243–18252, 2024. 2

[50] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022. 2, 6, 7