# EvoGround: Self-Evolving Video Agents for Video Temporal Grounding

**Minjoon Jung**[1,2] **Byoung-Tak Zhang**[2] **Lorenzo Torresani**[1]

[1]Northeastern University [2]Seoul National University

{jung.minjo,l.torresani}@northeastern.edu {btzhang}@bi.snu.ac.kr

## Abstract

Video temporal grounding (VTG) takes an untrimmed video and a natural-language query as input and localizes the temporal moment that best matches the query. Existing methods rely on large, task-specific datasets requiring costly manual annotation. We introduce EvoGround, a framework of two coupled self-evolving agents, a *proposer* and a *solver*, that learn temporal grounding from raw videos without any task-specific annotations. The proposer generates query–moment pairs from raw videos, while the solver learns to ground them and feeds back signals that improve the proposer in return. Through this self-reinforcing reinforcement-learning loop, the two agents are initialized from the *same* backbone and mutually improve across iterations. Trained on 2.5K unlabeled videos, EvoGround matches or surpasses most fully supervised models across five VTG benchmarks, while emerging as a state-of-the-art fine-grained video captioner. We publicly release all resources at https://minjoong507.github.io/projects/EvoGround/.

## 1 Introduction

Accurate temporal grounding underpins a wide range of applications, including video retrieval [1], highlight detection [2], and grounded question answering [3], and is increasingly relied upon as a core module in complex video understanding pipelines [4, 5]. Given an untrimmed video and a natural language query, video temporal grounding (VTG) [6] aims to localize the temporal moment within the video that corresponds to the query. Despite remarkable progress, existing models typically rely on large-scale human-labeled data for training, fundamentally limiting their scalability and applicability.

Collecting labeled data for VTG is particularly burdensome. Annotators must manually identify temporal segments and provide corresponding natural language descriptions within untrimmed videos, a time-consuming process that inherently limits scalability. This often leads to a sparsity issue [7, 8], where only a fraction of video events are covered, amplifying the narrowness of the training distribution. Beyond scalability, obtaining high-quality annotations remains non-trivial: queries are often ambiguous or misaligned with the underlying visual content, causing models to learn spurious correlations or exploit shortcuts rather than being grounded in true visual evidence [9–11].

*Can a model reliably learn temporal information in videos without manual labels?* At first glance, this seems unlikely, as temporal grounding requires precise, fine-grained reasoning that one might assume can only emerge from explicit supervision. We challenge this assumption with EvoGround, a framework of two coupled self-evolving agents that learn to ground temporal moments through mutual interaction without any task-specific video annotation.

As shown in Figure 1, EvoGround consists of a *proposer* and a *solver*, evolved through an iterative two-stage process via reinforcement learning (RL), with both agents initialized from the same base model. Rather than relying on human annotators, the proposer takes on their role: given a raw video, it identifies temporal moments and generates corresponding natural language queries. The solver

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

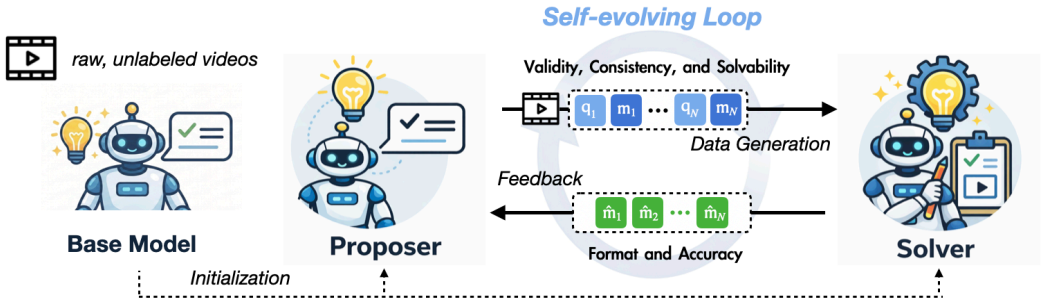


Figure 1: **EvoGround: a self-evolving loop, with unlabeled videos.** A proposer and a solver, both initialized from the same base model, co-evolve through reinforcement learning. The proposer generates query ($q$)–moment ($m$) pairs from a raw video; the solver grounds them and produces predictions ($\hat{m}$) that feed back as a learning signal. Dedicated reward designs guide each agent.

then attempts to ground these queries back to their moments, and its success or failure is fed back to the proposer as a learning signal.

In the first stage, we encourage the proposer to provide valid, semantically consistent, and solvable query-moment pairs; in the second stage, the solver is trained on the data generated by the updated proposer. Through this self-reinforcing loop with dedicated reward designs for each agent, they challenge and sharpen each other: a better proposer generates more meaningful and solvable queries, which in turn train a better solver, whose improved feedback further raises the bar for the proposer. This virtuous cycle, driven entirely by reinforcement learning, improves over successive iterations before saturating.

Despite never being exposed to any single manual grounding annotation during the self-evolution phases, EvoGround matches or outperforms most fully-supervised models across five VTG benchmarks: Charades-STA [6], ActivityNet-Captions [12], TVGBench [13], ReXTime [14], and E.T.Bench [15]. Moreover, EvoGround demonstrates its strong fine-grained video captioning capability on TemporalBench [16], suggesting that the self-evolution loop fosters a broader and more grounded understanding of video content.

Our primary contributions are summarized as follows:

- We propose EvoGround, a framework of self-evolving agents that learns temporal information from raw videos without any task-specific annotation.
- EvoGround adopts a proposer–solver loop in which the two agents, initialized from the same backbone, iteratively challenge and improve each other.
- EvoGround achieves accuracy on par with, and in five VTG benchmarks surpassing, most fully supervised models, while demonstrating its fine-grained video captioning on TemporalBench.

## 2 Related Work

**Video Temporal Grounding (VTG).** VTG is a task that aims to localize temporal moments from given natural language sentences within untrimmed videos. VTG models typically encode multimodal features using a pre-trained encoder (*e.g.*, CLIP [17]) and adopt proposal-based [18], proposal-free [19, 20], and DETR-based methods [2, 21] for prediction. However, these models are inherently less generalizable and limited by the capacity of pre-trained encoders. Such limitations have motivated a shift toward more general and scalable approaches that employ large language models (LLMs).

**Reducing Annotation Cost in VTG.** Prior work reduces but does not remove annotation cost. They typically synthesize pseudo query–moment pairs through fixed pipelines, leveraging pre-trained encoders [22, 1], prompted LLMs [23], or post-hoc refinement of generated annotations [24]. Because the generator is never updated, labels can be corrected but the proposal distribution cannot improve. EvoGround optimizes the generator itself, so data quality improves across iterations (Section 5, Appendix Table 16).

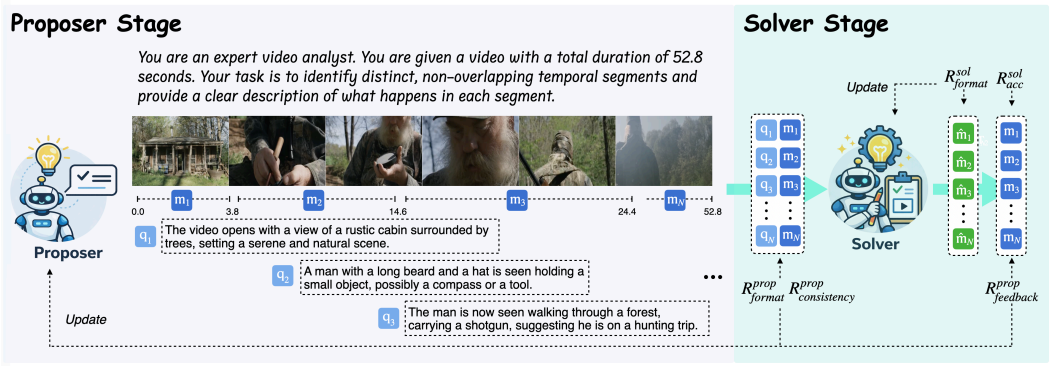


Figure 2: **Overview of EvoGround.** Both agents start from the same backbone. The proposer is updated via three rewards: $R^{\text{prop}}_{\text{format}}$ (*validity*), $R^{\text{prop}}_{\text{consistency}}$ (*consistency*, computed with SigLIP-2), and $R^{\text{prop}}_{\text{feedback}}$ (*solvability*, derived from the solver's tIoU). The solver is updated via $R^{\text{sol}}_{\text{format}}$ and $R^{\text{sol}}_{\text{acc}}$. Stages alternate: the proposer's pairs train the solver, the solver's predictions sharpen the proposer.

**Video Large Language Models (Video-LLMs).** Video-LLMs leverage the powerful capabilities of LLMs [25, 26] to handle a wide range of downstream tasks, from question answering [27–29] to fine-grained temporal understanding [30–32, 11, 5, 33]. Recently, inspired by the success of RL (*e.g.*, DeepSeek-R1 [34]), RL-based Video-LLMs [35, 13, 36, 37] have emerged and demonstrate superior spatio-temporal reasoning. However, existing models typically rely on labeled data and often require strong teacher models for guidance. For instance, VideoChat-R1.5 [37] leverages DeepSeek to ensure training data quality, and VideoChat-R1 [36] utilizes Qwen2.5-72B [38] to encourage alignment between model outputs and caption supervision. In contrast, EvoGround avoids such reliance and learns temporal information directly from raw videos through iterative self-reinforcing loops.

**Self-Evolving Agents.** Self-evolving frameworks [39–41] have improved LLMs with minimal or no training data. Recent works [42–44] have expanded this mechanism from text to vision-language. We adopt this loop, and extend it to video temporal understanding. Unlike text-domain self-play [40], video requires rewards that verify each task against the input and grade it by overlap rather than correctness. Both agents share the same backbone, with no stronger video grounder or teacher in the loop. Any gain reflects the procedure itself rather than distillation.

## 3 Method

As shown in Figure 2, we introduce EvoGround, consisting of two agents: a *proposer* and a *solver*. The proposer identifies candidate temporal events from raw videos and generates corresponding query-moment pairs, while the solver learns to ground temporal moments using the generated data. Both agents are initialized from the same backbone [45], and evolve solely through the self-reinforcing loop without any labeled data. Appendix Figure 6 provides full prompts for each agent.

### 3.1 Proposer

To guide the proposer toward generating high-quality data, we define three key criteria: *validity* (format reward), *consistency* (consistency reward), and *solvability* (feedback reward). (1) Validity: Temporal moments must be defined by numeric timestamps within a valid range (*i.e.*, non-negative and within the video duration) and avoid extremely short intervals or trivial spans that cover the entire video. (2) Consistency: A query–moment pair should be semantically coherent, *i.e.*, the query should accurately describe the visual content of the moment and be discriminative with respect to other candidates, independent of the solver. (3) Solvability: The generated pairs must be learnable by the solver, *i.e.*, the solver should be able to localize the moment from the query. A pair may be semantically consistent yet unsolvable if its temporal boundaries are too ambiguous or imprecise. We design the corresponding reward functions based on these criteria.

For each video, we prompt the proposer to generate a set of temporal moments and corresponding queries. Let the proposer generate $N$ query-moment pairs $\{(q_n, m_n)\}_{n=1}^{N}$, where $q_n$ denotes the $n$-th query and $m_n = (s_n, e_n)$ represents the corresponding temporal moment with start time

$s_n$ and end time $e_n$ in seconds. For each query $q_n$, the solver produces a moment prediction $\hat{m}_n = (\hat{s}_n, \hat{e}_n)$. Based on these generated pairs and predictions, we denote the $i$-th generated sample as $\mathbf{s}_i = \{q_n, m_n, \hat{m}_n\}_{n=1}^{N}$, which serves as the input to all reward functions defined below.

**Format Reward.** The format reward verifies whether the proposer generates valid query–moment pairs. To facilitate reliable parsing and evaluation, we prompt the model to provide multiple moments and queries, each enclosed within predefined tags, `<time>...</time>` and `<description>...</description>`, respectively. The format reward is defined as:

$$R^{\text{prop}}_{\text{format}}(\mathbf{s}_i) = \begin{cases} \frac{1}{N}\sum_{n=1}^{N} \mathbb{I}\big((s_n, e_n) \in \mathcal{V}\big), & \text{if the proposer follows the template,} \\ 0, & \text{otherwise,} \end{cases} \tag{1}$$

where $\mathbb{I}(\cdot)$ is the indicator function and $\mathcal{V}$ denotes the set of valid temporal moments satisfying: (1) $s_n \geqslant 0$ and $e_n \leqslant T_{\text{video}}$ and (2) $e_n - s_n > \tau_{\text{length}}$, where $T_{\text{video}}$ is the video duration and $\tau_{\text{length}}$ specifies the minimum segment length. In later iterations, we further condition the format reward on the solver's feedback to ensure that generated query–moment pairs are not only structurally valid but also solvable. We measure the feedback using timestamp-aware IoU (tIoU) [13]:

$$\text{tIoU}(m_n, \hat{m}_n) = \text{IoU}(m_n, \hat{m}_n) \cdot \left(1 - \frac{|s_n - \hat{s}_n|}{T_{\text{video}}}\right) \cdot \left(1 - \frac{|e_n - \hat{e}_n|}{T_{\text{video}}}\right), \tag{2}$$

which extends standard IoU [9] by incorporating penalties for deviations in start and end timestamps relative to the video duration, leading to improved boundary sensitivity. Here, IoU is defined as:

$$\text{IoU}(m_n, \hat{m}_n) = \frac{|[s_n, e_n] \cap [\hat{s}_n, \hat{e}_n]|}{|[s_n, e_n] \cup [\hat{s}_n, \hat{e}_n]|}. \tag{3}$$

Based on this, we define the *conditioned format reward* as:

$$\mathbb{I}\big((s_n, e_n) \in \mathcal{V} \ \wedge \ \text{tIoU}(m_n, \hat{m}_n) \geqslant \delta\big), \tag{4}$$

where $\delta \in [0, 1]$ is a threshold that conditions the validity of query–moment pairs on the current solver's accuracy. This design ensures that the proposer generates pairs that are both structurally valid and learnable at each training step. The format reward in Equation 1 corresponds to the case of $\delta = 0$.

**Consistency Reward.** The consistency reward measures two complementary properties of each query-moment pair: *intra-consistency*, which captures how coherently the query aligns with the frames within its corresponding moment, and *inter-consistency*, which captures how discriminatively the query matches its own moment relative to other candidates. We use SigLIP-2 [46], an image–text encoder carrying no temporal grounding supervision, to compute frame-level similarities $S_{n,t} = \cos(q_n, f_t)$ by uniformly sampling $|m_n|$ frames from each moment at a rate proportional to its duration, and obtain their average $\mu_n$. The intra-consistency score penalizes high standard deviation in $S_{n,t}$, rewarding pairs where the query aligns uniformly with frames in $m_n$:

$$\text{Intra}_n = \exp(-\gamma \cdot \sigma_n), \quad \sigma_n = \sqrt{\frac{1}{|m_n|}\sum_{t=1}^{|m_n|}(S_{n,t} - \mu_n)^2}, \tag{5}$$

where $\gamma$ controls sensitivity to alignment fluctuations. However, a pair with uniformly low similarity would still score highly under $\text{Intra}_n$ alone. The inter-consistency score addresses this by measuring how discriminatively the query matches its own moment relative to others, computed as a softmax over cross-moment similarities with temperature $\tau$:

$$\text{Inter}_n = \frac{\exp(\mu_n/\tau)}{\sum_{j=1}^{N}\exp(\mu_{n,j}/\tau)}, \tag{6}$$

where $\mu_{n,j}$ denotes the mean similarity between query $q_n$ and moment $m_j$. The final consistency reward is their product, requiring both temporal coherence within the moment and semantic discriminability across moments to be simultaneously satisfied:

$$R^{\text{prop}}_{\text{consistency}}(\mathbf{s}_i) = \frac{1}{N}\sum_{n=1}^{N} \text{Intra}_n \cdot \text{Inter}_n. \tag{7}$$

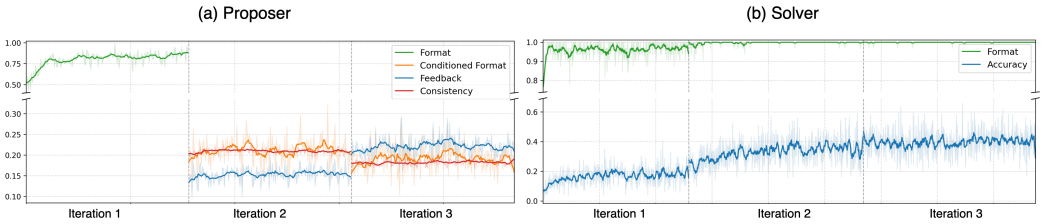


Figure 3: **Reward dynamics across iterations.** We visualize the evolution of the proposer and solver in (a) and (b), respectively. As the proposer evolves over iterations, the solver correspondingly demonstrates progressively higher accuracy.

In Appendix A.3, we provide more details on the intra-consistency formulation and the computation of $|m_n|$. We further explore variants of inter-consistency under different formulations (*e.g.*, hard ranking-based) and find that the softmax formulation consistently outperforms the alternatives.

**Feedback Reward.** The feedback reward confirms whether the pair is solvable and is measured by the solver's accuracy:

$$R^{\text{prop}}_{\text{feedback}}(\mathbf{s}_i) = \frac{1}{N}\sum_{n=1}^{N} \text{tIoU}(m_n, \hat{m}_n), \tag{8}$$

where $\hat{m}_n$ denotes the moment predictions from query $q_n$.

### 3.2 Solver

From the query-moment pairs generated by the proposer, each pair is processed independently. We consider a single query–moment pair $(q, m)$. Given the query $q$, the solver generates multiple rollout samples $\{o_k\}_{k=1}^{K}$, from which we extract the corresponding predicted moments $\{\hat{m}_k\}_{k=1}^{K}$.

**Format Reward.** To facilitate easy answer extraction, we encourage the solver to follow the pre-defined template: `<think>...</think>` and `<answer>...</answer>`, where the `<think>` block contains the model's intermediate reasoning steps and the `<answer>` block contains the predicted temporal moment as start and end timestamps. Based on this, the format reward is defined as:

$$R^{\text{sol}}_{\text{format}}(o_k) = \begin{cases} 1, & \text{if the solver follows the template,} \\ 0, & \text{otherwise.} \end{cases} \tag{9}$$

**Accuracy Reward.** To improve accuracy, we extract the prediction $\hat{m}_k$ from the content within the answer tags and measure accuracy by comparing it with the ground-truth moment $m$:

$$R^{\text{sol}}_{\text{acc}}(o_k) = \text{tIoU}(m, \hat{m}_k). \tag{10}$$

Note that while $R^{\text{sol}}_{\text{acc}}$ and $R^{\text{prop}}_{\text{feedback}}$ share the same formulation, they serve distinct roles: the former optimizes the solver toward accurate grounding, while the latter signals to the proposer whether its generated pairs are learnable.

### 3.3 Optimization

In this section, we describe how EvoGround is optimized. EvoGround alternates between two stages: in the *proposer stage*, the proposer is optimized while the solver is frozen; in the *solver stage*, the solver is updated using query–moment pairs generated by the frozen proposer.

**Objective.** We adopt GDPO [47] as our RL optimizer, which improves upon GRPO [34] by weighting and normalizing each reward component individually before aggregation, rather than summing first and normalizing after. This improves the resolution of the training signals and prevents easier rewards from dominating the optimization, which is critical in our multi-reward setting. We provide the detailed formulation of each objective in Appendix A.1.

**Curriculum Design.** In the first iteration, we warm up the proposer with the format reward only and update the solver, since neither agent is yet tailored to the task. In subsequent iterations, we apply consistency and feedback rewards to the proposer and adopt GDPO for this multi-reward

Table 1: **Results on Charades-STA and ActivityNet-Captions.** EvoGround achieves competitive or superior performance to existing models without using any manual labels, outperforming all SFT-based models and matching the best RL-based models. * denotes our reproduced results.

| Method | Charades-STA | | | | ActivityNet-Captions | | | |
|---|---|---|---|---|---|---|---|---|
| | R1@0.3 | R1@0.5 | R1@0.7 | mIoU | R1@0.3 | R1@0.5 | R1@0.7 | mIoU |
| *SFT-based Models* | | | | | | | | |
| TimeChat [32] | - | 32.2 | 13.4 | 32.2 | 36.2 | 20.2 | 9.5 | 21.8 |
| HawkEye [48] | 50.6 | 31.4 | 14.5 | - | 49.1 | 29.3 | 10.7 | - |
| VTimeLLM [30] | 51.0 | 27.5 | 11.4 | 31.2 | 44.0 | 27.8 | 14.3 | 30.4 |
| Qwen2.5-VL* [45] | 68.5 | 48.8 | 22.5 | 45.0 | 38.7 | 25.6 | 14.9 | 28.6 |
| TimeSuite [33] | 69.9 | 48.7 | 24.0 | - | - | - | - | - |
| TRACE [49] | 58.6 | 40.3 | 19.4 | 38.7 | - | - | - | - |
| ED-VTG [50] | 59.5 | 39.3 | 19.8 | 40.2 | 52.1 | 33.1 | 16.0 | 35.2 |
| *RL-based Models* | | | | | | | | |
| VideoChat-R1 [36] | - | - | - | - | 50.4 | 32.2 | 16.2 | 34.3 |
| VideoChat-R1.5 [37] | - | - | - | - | 52.4 | 32.3 | 16.8 | 35.5 |
| Time-R1 [13] | **78.1** | **60.8** | 35.3 | **58.1** | 58.6 | 39.0 | 21.4 | 40.5 |
| **EvoGround (Ours)** | 77.2 $\pm$1.3 | 60.5 $\pm$1.6 | **35.5** $\pm$1.5 | 53.1 $\pm$0.9 | **62.9** $\pm$0.8 | **43.6** $\pm$0.7 | **25.0** $\pm$0.6 | **43.9** $\pm$0.5 |

setting, assigning higher weights to the feedback reward than to the format reward to emphasize more challenging objectives. We also progressively increase $\delta$ (in Equation 4) across iterations, shifting the focus from coarse matches toward more precise temporal alignments. We validate the effectiveness of these design choices in Section 5. For the solver, we keep the design simple, applying naive GRPO without additional reward designs or the multi-reward weighting scheme (*i.e.*, GDPO).

**Reward Dynamics.** Figure 3 visualizes the reward dynamics across iterations. Both agents quickly saturate the format reward in the first iteration. In subsequent iterations the conditioned format reward drops as $\delta$ increases, while the feedback reward steadily improves, tracking the solver's increasing accuracy. This confirms that our curriculum design effectively guides optimization.

# 4 Experiments

## 4.1 Experiment Setup

**Baselines.** We consider Video-LLMs as baselines, most of which focus on VTG. We categorize them into two folds: SFT-based models, including TimeChat [32], HawkEye [48], VTimeLLM [30], Qwen2.5-VL [45], TimeSuite [33], VideoChat-Flash [51], LITA [52], TRACE [49], and ED-VTG [50], and RL-based models, including VideoChat-R1 [36], VideoChat-R1.5 [37], and Time-R1 [13]. For captioning, we include various Video-LLMs, including Qwen2-VL [38], LLaVA-OneVision [53], LLaVA-NeXT-Video [54], InternLM-XC2.5 [55], VideoLLaVA [56], MiniCPM-V2.6 [57], Phi-3.5.Vision [58], MA-LLM [59], and Matryoshka Multimodal Models ($M^3$) [60]. Models are 7B-scale unless otherwise specified. Among all baselines, EvoGround is the only model explicitly trained for VTG without task-specific annotations.

**Benchmarks.** We conduct experiments on five VTG benchmarks: Charades-STA [6], ActivityNet-Captions [12], TVGBench [13], ReXTime [14], and E.T.Bench [15]. Additionally, we use TemporalBench [16] to evaluate the model's captioning capability. We follow the cross-dataset zero-shot protocol of prior RL-based work [13], where no model is fine-tuned on the evaluated benchmarks.

**Evaluation Metrics.** For grounding, we use R@$n$, IoU=$m$ metric, which measures the percentage of top-$n$ predicted moments whose IoU with the ground-truth moment exceeds a threshold $m \in \{0.3, 0.5, 0.7\}$, and also report the mean IoU (mIoU) and F1 score. For further reliable comparison, we report 95% confidence intervals estimated via nonparametric bootstrap (*i.e.*, 1,000 resamples with replacement of the test set[1]). Point estimates are computed on the full test set, and intervals correspond to the 2.5th–97.5th percentiles of the bootstrap distribution. For captioning, we adopt classical captioning metrics: CIDEr [61], BLEU [62] at various n-gram levels, ROUGE [63], and sentence similarity between generated and ground-truth captions with a pre-trained Sentence Transformer [64].

[1]We exclude ReXTime for this since its ground-truth data is released in the official test server

Table 2: **Results on TVGBench, ReXTime, and E.T.Bench.** Across diverse benchmarks spanning varying video durations and domains, EvoGround surpasses SFT-based models and remains competitive with RL-based models trained on human-annotated data. * denotes our reproduced results.

| Method | TVGBench | | | | ReXTime | | | E.T.Bench |
|---|---|---|---|---|---|---|---|---|
| | R1@0.3 | R1@0.5 | R1@0.7 | mIoU | R1@0.3 | R1@0.5 | mIoU | $TVG_{F1}$ |
| *SFT-based Models* | | | | | | | | |
| TimeChat [32] | 22.4 | 11.9 | 5.3 | - | 14.4 | 7.6 | 11.6 | 26.2 |
| VTimeLLM [30] | - | - | - | - | 28.8 | 17.4 | 20.1 | 7.6 |
| Qwen2.5-VL* [45] | 28.1 | 19.5 | 10.5 | 20.4 | 16.3 | 11.2 | 13.3 | 46.6 |
| LITA-13B [52] | - | - | - | - | 29.4 | 16.2 | 21.4 | 22.2 |
| TimeSuite [33] | 31.1 | 18.0 | 8.9 | - | - | - | - | - |
| VideoChat-Flash [51] | 32.8 | 19.8 | 10.4 | - | - | - | - | - |
| TRACE [49] | 37.0 | 25.5 | 14.6 | - | - | - | - | - |
| *RL-based Models* | | | | | | | | |
| VideoChat-R1.5* [37] | 31.5 | 20.8 | 11.3 | 22.0 | 28.3 | 18.0 | 20.5 | 50.3 |
| Time-R1* [13] | <u>39.3</u> | <u>28.0</u> | **16.0** | <u>27.8</u> | <u>32.2</u> | <u>22.1</u> | <u>24.1</u> | **69.4** |
| **EvoGround (Ours)** | **42.1** ±3.3 | **28.5** ±3.1 | <u>15.3</u> ±2.6 | **29.4** ±2.1 | **33.5** | **22.4** | **25.5** | <u>69.0</u> ±2.2 |

Table 3: **Results on TemporalBench.** EvoGround outperforms all prior models across all captioning metrics, demonstrating broader video understanding beyond temporal grounding.

| Method | TemporalBench | | | | | | |
|---|---|---|---|---|---|---|---|
| | Similarity | CIDEr | ROUGE | BLEU_1 | BLEU_2 | BLEU_3 | BLEU_4 |
| Qwen2-VL [38] | 51.9 | 6.9 | 18.0 | 12.5 | 6.1 | 3.0 | 1.6 |
| LLaVA-OneVision [53] | 50.1 | 0.3 | 14.5 | 11.1 | 5.1 | 2.2 | 1.1 |
| LLaVA-NeXT-Video [54] | 50.1 | 2.3 | 15.8 | 18.1 | 7.0 | 2.6 | 1.1 |
| InternLM-XC2.5 [55] | 52.4 | 2.3 | 15.9 | 17.8 | 7.1 | 2.8 | 1.2 |
| VideoLLaVA [56] | 46.0 | 4.5 | 16.9 | 12.6 | 5.4 | 2.3 | 1.0 |
| MiniCPM-V2.6 [57] | 47.2 | 1.5 | 14.2 | 15.5 | 5.4 | 1.9 | 0.8 |
| Phi-3.5-Vision [58] | 42.9 | 3.7 | 16.5 | 20.4 | 8.4 | 3.4 | 1.6 |
| MA-LMM [59] | 38.7 | 3.1 | 15.0 | 10.1 | 4.8 | 2.2 | 1.1 |
| $M^3$ [60] | 47.8 | 3.0 | 16.4 | 16.7 | 6.9 | 2.8 | 1.2 |
| **EvoGround (Ours)** | **53.7** | **11.1** | **19.8** | **26.7** | **12.3** | **5.7** | **2.7** |

**Implementation Details.** We use Qwen2.5-VL-7B [45] as a backbone for a fair comparison with prior RL-based models [36, 37, 13] and utilize 2.5K raw videos from TimeRFT [13]. We perform three iterations to train both agents, and each iteration generates ∼9K query–moment pairs, completing in ∼24 hours on 4×A100-80GB. For the proposer, we progressively increase $\delta$ ($0 \rightarrow 0.3 \rightarrow 0.5$) and assign reward weights of $w_{\text{format}} = 0.5$, $w_{\text{consistency}} = 0.5$, and $w_{\text{feedback}} = 1.0$. Following [13], video frames are sampled at 2 FPS and adaptively resized to 2.8M pixels to control efficiency during both training and testing; results marked with * in the following tables are obtained from the official checkpoint under the same setup. Please refer to Appendix A.2 for more implementation details.

### 4.2 Main Results

Despite never seeing any manual labels, EvoGround demonstrates strong performance across all benchmarks. In Tables 1 and 2, EvoGround surpasses all SFT-based models and achieves first- or second-best performance among RL-based models, which, however, rely on manually labeled data. In Table 3, we evaluate the proposer on TemporalBench, which reflects the model's fine-grained captioning capability to describe a sequence of events throughout the video. All models here use the single-pass protocol of the original benchmark. EvoGround outperforms all prior models across every metric, with particularly large margins on CIDEr and BLEU, despite never being explicitly trained for captioning. Beyond performance, EvoGround is significantly more data-efficient than existing models, which typically involve data generation and curation pipelines. For instance, VideoChat-R1.5 [37] constructs VTTS-80K from 11 public datasets, and ED-VTG [50] collects 136K samples from 8 public datasets, whereas we use only 2.5K raw videos. We also compare our training budget against Time-R1 [13] and ED-VTG [50] in Appendix Table 15. Overall, these results demonstrate that EvoGround effectively learns temporal information without manual labels.

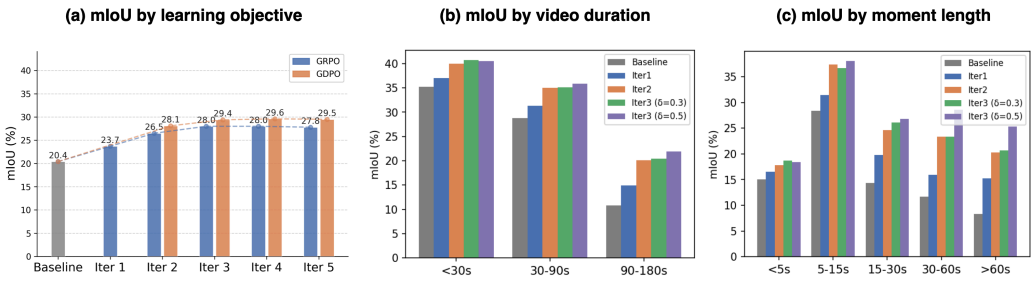


Figure 4: **Improvements across iterations on TVGBench.** (a) shows performance using different learning objectives. (b) and (c) show performance across different video and moment lengths.

Table 4: **Ablation on data scaling.** FT indicates fine-tuning on Charades-STA training data. In contrast, EvoGround (7.5K) additionally uses 5K raw videos from Charades-STA without manual labels.

| Method | FT | Charades-STA | | | |
|---|---|---|---|---|---|
| | | R1@0.3 | R1@0.5 | R1@0.7 | mIoU |
| Hawkeye [48] | ✓ | 72.5 | 58.3 | 28.8 | 49.3 |
| TimeChat [32] | ✓ | - | 46.7 | 23.7 | - |
| TimeChat-VT [11] | ✓ | - | 58.4 | 34.7 | - |
| ED-VTG [50] | ✓ | 78.2 | 62.1 | 35.0 | 52.6 |
| VideoChat-R1 [36] | ✓ | - | 71.7 | **50.2** | 60.8 |
| VideoChat-R1.5 [37] | ✓ | **82.8** | 71.6 | 48.3 | 60.6 |
| Time-R1 [13] | ✓ | **82.8** | **72.2** | 50.1 | **60.9** |
| **EvoGround** (2.5K) | ✗ | 77.2 | 60.5 | 35.5 | 53.1 |
| **EvoGround** (7.5K) | ✗ | 81.7 | 67.7 | 45.6 | 60.3 |

Table 5: **Ablation on rewards.** Each reward contributes incrementally, with all rewards combined achieving the best performance.

| Iter | $R_{\text{fmt}}$ | $R_{\text{con}}$ | $R_{\text{feed}}$ | TVGBench | | |
|---|---|---|---|---|---|---|
| | | | | R1@0.3 | R1@0.5 | R1@0.7 |
| 1 | ✓ | | | 34.0 | 21.2 | 12.0 |
| 2 | ✓ | | | 37.0 | 23.8 | 12.5 |
| 2 | ✓ | ✓ | | 38.7 | 26.2 | 14.6 |
| 2 | ✓ | | ✓ | 38.1 | 26.0 | 14.4 |
| 2 | ✓ | ✓ | ✓ | **40.1** | **27.4** | **14.9** |
| 3 | ✓ | | | 37.2 | 23.6 | 13.5 |
| 3 | ✓ | ✓ | | 41.7 | 27.3 | 15.0 |
| 3 | ✓ | | ✓ | 41.8 | 27.4 | 14.8 |
| 3 | ✓ | ✓ | ✓ | **42.1** | **28.5** | **15.3** |

## 5 Analysis

In this section, we provide in-depth analyses of EvoGround.

**GDPO outperforms GRPO.** Figure 4 (a) reports the performance using different RL learning objectives across iterations. GDPO consistently outperforms GRPO across all iterations and already surpasses GRPO's best performance at the second iteration, achieving faster convergence. These results support our claim that GDPO is well-suited for our multi-reward settings.

**Increasing $\delta$ shifts the model's focus toward more accurate grounding.** We compare the cases where $\delta$ is increased to 0.5 versus kept at 0.3. In Figure 4 (b) and (c), $\delta = 0.5$ shows a more balanced improvement than $\delta = 0.3$ across different video durations and moment lengths. Specifically, $\delta = 0.3$ yields only marginal improvement over the previous iteration, and its effectiveness diminishes particularly for longer videos and moments. This suggests that increasing $\delta$ encourages the proposer to foster a more robust solver. This does not bias toward trivially easy samples, as the improvement is most pronounced for longer videos and moments where grounding is inherently harder.

**Co-evolution, not repeated self-training, drives the gains.** Because the solver trains on proposer outputs, EvoGround might reduce to one round of RL on filtered self-generated data. We test this with three controls on the same videos, backbone and objective, matched in solver update steps: random moments captioned by the frozen base model (B0), one-shot pseudo-labels from the frozen base model (B1), and a proposer frozen after one bootstrap iteration (B2), where each later iteration is another pass over the same fixed data. In Table 6 the static pipelines plateau ($+3.3$ R1@0.3 for B2) while EvoGround gains $+8.1$, so the benefit comes from updating the generator, not from further solver training.

**Adding more videos restarts the loop.** Figure 4 (a) shows that gains plateau by iteration 3. With only 2.5K raw videos, the proposer and solver have already learned what they can from this small pool, and later iterations mostly recycle similar pairs. In Table 4, training EvoGround on 5K additional raw videos from Charades-STA (7.5K total, no overlap) for one extra iteration yields further gains, remaining competitive with models fine-tuned on human annotations. Appendix Table 12 shows smaller but positive transfer to other benchmarks.

Table 6: **Co-evolution vs. static pseudo-labeling on TVGBench** (R1@0.3 / R1@0.5 / R1@0.7). All rows use the same 2.5K raw videos, backbone and RL objective, matched in total solver update steps. For the static controls (B0–B2), each "iteration" is an additional pass over the same fixed data.

| Method | Iter 1 | Iter 2 | Iter 3 |
|---|---|---|---|
| B0 (random moment + captioning) | 30.2 / 19.8 / 10.8 | 32.4 / 20.2 / 10.9 | 33.8 / 21.6 / 11.3 |
| B1 (one-shot pseudo-label) | 31.3 / 20.2 / 10.6 | 32.0 / 20.3 / 11.0 | 32.3 / 20.5 / 11.1 |
| B2 (frozen iter-1 bootstrap proposer) | 34.0 / 21.2 / 12.0 | 35.2 / 24.8 / 12.5 | 37.3 / 25.5 / 13.3 |
| **EvoGround (full co-evolution)** | 34.0 / 21.2 / 12.0 | **40.1 / 27.4 / 14.9** | **42.1 / 28.5 / 15.3** |

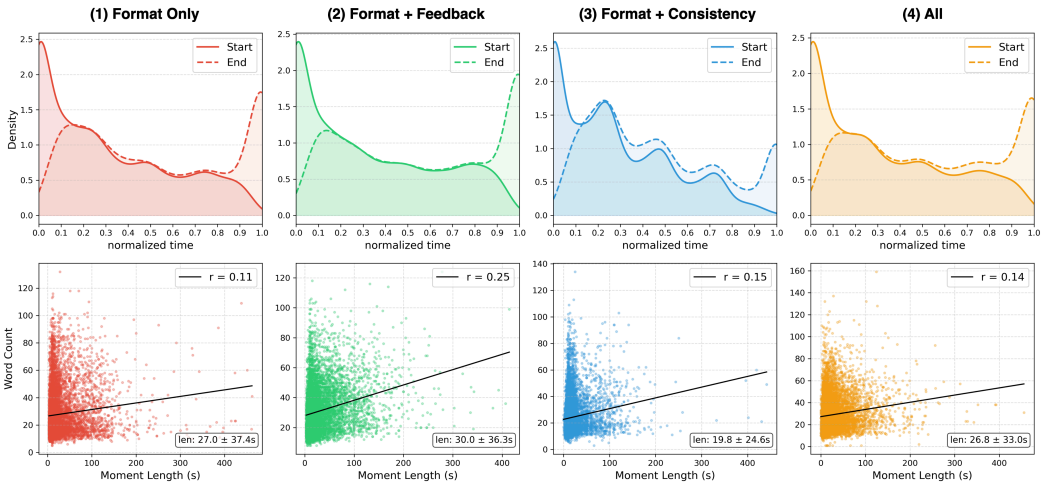


Figure 5: **Generated data distributions across different reward configurations.** *Top:* Kernel density estimates of normalized start and end times of the moment, shown as solid and dashed lines, respectively. All times are normalized by video duration. *Bottom:* We report the correlation ($r$) between query and moment length, along with the mean $\pm$ standard deviation of moment lengths.

**Each reward plays a distinct role.** We optimize our model under different reward configurations: (1) format only, (2) format + feedback, (3) format + consistency, and (4) all rewards, and visualize their moment distributions and query–moment length correlation ($r$) in Figure 5, with performance in Table 5. Each reward addresses a specific failure mode of the proposer, jointly preventing collapse to trivial outputs. The format reward enforces structural validity, but formatting alone yields the worst performance, suggesting the proposer collapses into biased, repetitive data without quality constraints. The feedback reward enriches query descriptiveness ($r$: 0.11 $\rightarrow$ 0.25), though $r$ is a diagnostic signal rather than a quality metric: a longer query does not necessarily describe the moment better. The consistency reward sharpens the moment distribution, producing tighter temporal moments (27.0 $\pm$ 37.4 $\rightarrow$ 19.8 $\pm$ 24.6) and reducing extension toward the video end. The tri-modal start time distribution reflects the consistency reward's tendency to anchor moments at discriminative positions, avoiding overlap with other candidates. Combining all rewards moderates $r$ to 0.14 and yields the balanced distribution and best performance overall.

**Gains transfer across backbone scales and families.** Across four backbones spanning 3B–8B parameters and two families (*i.e.*, QwenVL and InternVL), EvoGround delivers nearly identical gains and benefits from larger capacity, with Qwen3-VL-8B [65] reaching the highest absolute scores. We will explore more aggressive scaling in future work.

**EvoGround generalizes beyond grounding.** We evaluate the proposer on dense video captioning [12]. Table 8 reports results on ActivityNet-Captions, showing that EvoGround achieves the highest METEOR score among all models. Despite a lower SODA_c score, it remains competitive without involving any caption data, unlike prior methods fine-tuned on human caption supervision. We further evaluate the solver on representative VideoQA benchmarks, Video-MME [69] and TempCompass [70], in Appendix Table 11, where EvoGround achieves slight improvements over the backbone without disrupting its original QA capabilities.

Table 7: **Ablation on backbones.** Gray columns show initial performance, and blue indicates improvements from EvoGround. Gains are stable across backbone size and family, suggesting the framework rather than backbone-specific behavior drives improvement.

| Backbone | TVGBench | | | | | |
|---|---|---|---|---|---|---|
| | R1@0.3 | | R1@0.5 | | R1@0.7 | |
| Qwen2.5-VL-3B [45] | 17.5 | **31.3** (+13.8↑) | 10.0 | **19.4** (+9.4↑) | 5.2 | **9.9** (+4.7↑) |
| Intern3.5-VL-4B [66] | 14.2 | **30.6** (+16.4↑) | 8.5 | **18.1** (+9.6↑) | 4.0 | **9.6** (+5.6↑) |
| Qwen2.5-VL-7B [45] | 28.1 | **42.1** (+14.0↑) | 19.5 | **28.5** (+9.0↑) | 10.5 | **15.3** (+4.8↑) |
| Qwen3-VL-8B [65] | 30.2 | **44.6** (+14.4↑) | 20.8 | **30.1** (+9.3↑) | 10.5 | **17.2** (+6.7↑) |

Table 8: **Dense video captioning results.** EvoGround remains competitive with existing models. FT denotes training on dense captions in ActivityNet-Captions.

| Method | FT | ActivityNet-Captions | |
|---|---|---|---|
| | | SODA_c | METEOR |
| Momentor [67] | ✗ | 2.3 | 4.7 |
| VTimeLLM [30] | ✓ | 5.8 | 6.8 |
| TRACE [49] | ✓ | **6.4** | 6.0 |
| Grounded-VideoLLM [68] | ✓ | 6.2 | 6.4 |
| **EvoGround (Ours)** | ✗ | 4.1 | **7.0** |

Table 9: **Sensitivity of the consistency reward to encoder choice.** We report reward statistics (mean $\pm$ standard deviation) alongside TVGBench performance using SigLIP-2 and CLIP ViT-B/32. The last row reports cross-encoder Spearman correlations ($\rho$) for the reward components and final reward.

| Encoder | Similarity ($\mu$) | Inter-consistency | Intra-consistency | Final reward | mIoU |
|---|---|---|---|---|---|
| SigLIP-2 [46] | $0.015 \pm 0.023$ | $0.206 \pm 0.027$ | $0.732 \pm 0.117$ | $0.150 \pm 0.028$ | 29.1 |
| CLIP ViT-B/32 [17] | $0.270 \pm 0.042$ | $0.233 \pm 0.080$ | $0.556 \pm 0.198$ | $0.127 \pm 0.060$ | 29.4 |
| Spearman correlations ($\rho$) | 0.09 | 0.12 | 0.50 | 0.36 | - |

**The consistency reward remains comparable in scale across encoders.** We investigate the effect of different visual encoders on consistency reward computation. In Table 9, we sample 500 query–moment pairs from TimeRFT and evaluate the components of the consistency reward using SigLIP-2 and CLIP ViT-B/32. Although the mean similarities differ by more than an order of magnitude (0.015 vs. 0.270), both the mean consistency rewards (0.150 vs. 0.127) and performance (29.1 vs. 29.4) remain comparable. This suggests that the softmax-based inter-term and bounded intra-consistency help mitigate differences in similarity scales across visual encoders. Together with their comparable downstream performance, this suggests the consistency reward can be computed stably with different visual encoders, although the softmax temperature $\tau$ may require recalibration per encoder.

## 6 Conclusion

We presented EvoGround, a framework of coupled self-evolving agents that learns temporal grounding from raw videos without any task-specific grounding labels. Through a proposer-solver loop driven entirely by reinforcement learning, the two agents iteratively challenge and improve each other, yielding a system that matches or outperforms most fully supervised models across multiple benchmarks for both grounding and captioning. We hope this work inspires a broader shift toward annotation-free video understanding systems that learn directly from the raw visual world.

**Limitations.** Despite the effectiveness of EvoGround, several challenges remain. First, it becomes costly when handling ultra-long videos, as it generates a large number of query–moment pairs and needs the solver's feedback for each pair. Second, gains are front-loaded and saturate by iteration 3 on a fixed corpus. Scaling the corpus restarts the loop, but we have not verified beyond 7.5K videos that the ceiling is imposed by the corpus rather than the mechanism (see Appendix A.4). Third, the loop is annotation-free but not prior-free: the consistency reward uses a general-purpose image–text encoder, and a non-web-aligned encoder would be the cleanest test of where the signal originates. Fourth, the solver's reward treats the proposer's moment as ground truth, so the objective is self-referential, mitigated but not replaced by our external-judge audit.

## Acknowledgements and Disclosure of Funding

This work was partly supported by the IITP (RS-2021-II212068-AIHub/10%, RS-2021-II211343-GSAI/15%, RS-2022-II220951-LBA/15%, RS-2022-II220953-PICA/20%), NRF (RS-2024-00353991-SPARC/20%, RS-2023-00274280-HEI/10%), and KEIT (RS-2024-00423940/10%) grant funded by the Korean government

**Competing interests.** The authors declare no competing interests. No third-party funding beyond the grants listed above supported this work during the preceding 36 months, and the authors have no financial relationships outside the submitted work that could be perceived as influencing it.

## References

[1] M. Jung, S. Choi, J. Kim, J.-H. Kim, and B.-T. Zhang, "Modal-specific pseudo query generation for video corpus moment retrieval," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 7769–7781, Dec. 2022.

[2] J. Lei, T. L. Berg, and M. Bansal, "Detecting moments and highlights in videos via natural language queries," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11846–11858, 2021.

[3] J. Xiao, A. Yao, Y. Li, and T.-S. Chua, "Can i trust your answer? visually grounded video question answering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13204–13214, 2024.

[4] Z. Shangguan, C. Li, Y. Ding, Y. Zheng, Y. Zhao, T. Fitzgerald, and A. Cohan, "Tomato: Assessing visual temporal reasoning capabilities in multimodal foundation models," *arXiv preprint arXiv:2410.23266*, 2024.

[5] M. Jung, J. Xiao, J. Kim, B.-T. Zhang, and A. Yao, "Egoexo-con: Exploring view-invariant video temporal understanding," in *European Conference on Computer Vision*, pp. 355–373, Springer, 2026.

[6] J. Gao, C. Sun, Z. Yang, and R. Nevatia, "Tall: Temporal activity localization via language query," in *Proceedings of the IEEE international conference on computer vision*, pp. 5267–5275, 2017.

[7] R. Zeng, H. Xu, W. Huang, P. Chen, M. Tan, and C. Gan, "Dense regression network for video grounding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10287–10296, 2020.

[8] H. Li, M. Cao, X. Cheng, Y. Li, Z. Zhu, and Y. Zou, "Exploiting auxiliary caption for video grounding," in *Proceedings of the AAAI conference on artificial intelligence*, 2024.

[9] Y. Yuan, X. Lan, X. Wang, L. Chen, Z. Wang, and W. Zhu, "A closer look at temporal sentence grounding in videos: Dataset and metric," in *Proceedings of the 2nd international workshop on human-centric multimedia analysis*, pp. 13–21, 2021.

[10] M. Otani, Y. Nakashima, E. Rahtu, and J. Heikkilä, "Uncovering hidden challenges in query-based video moment retrieval," *arXiv preprint arXiv:2009.00325*, 2020.

[11] M. Jung, J. Xiao, B.-T. Zhang, and A. Yao, "On the consistency of video large language models in temporal comprehension," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 13713–13722, 2025.

[12] R. Krishna, K. Hata, F. Ren, L. Fei-Fei, and J. C. Niebles, "Dense-captioning events in videos," in *International Conference on Computer Vision (ICCV)*, 2017.

[13] Y. Wang, Z. Wang, B. Xu, Y. Du, K. Lin, Z. Xiao, Z. Yue, J. Ju, L. Zhang, D. Yang, *et al.*, "Time-r1: Post-training large vision language model for temporal video grounding," *arXiv preprint arXiv:2503.13377*, 2025.

[14] J.-J. Chen, Y.-C. Liao, H.-C. Lin, Y.-C. Yu, Y.-C. Chen, and F. Wang, "Rextime: A benchmark suite for reasoning-across-time in videos," *Advances in Neural Information Processing Systems*, vol. 37, pp. 28662–28673, 2024.

[15] Y. Liu, Z. Ma, Z. Qi, Y. Wu, Y. Shan, and C. W. Chen, "Et bench: Towards open-ended event-level video-language understanding," *Advances in Neural Information Processing Systems*, vol. 37, pp. 32076–32110, 2024.

[16] M. Cai, R. Tan, J. Zhang, B. Zou, K. Zhang, F. Yao, F. Zhu, J. Gu, Y. Zhong, Y. Shang, *et al.*, "Temporalbench: Benchmarking fine-grained temporal understanding for multimodal video models," *arXiv preprint arXiv:2410.10818*, 2024.

[17] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763, PmLR, 2021.

[18] S. Zhang, H. Peng, J. Fu, and J. Luo, "Learning 2d temporal adjacent networks for moment localization with natural language," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 12870–12877, 2020.

[19] H. Zhang, A. Sun, W. Jing, and J. T. Zhou, "Span-based localizing network for natural language video localization," *arXiv preprint arXiv:2004.13931*, 2020.

[20] J. Mun, M. Cho, and B. Han, "Local-global video-text interactions for temporal grounding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10810–10819, 2020.

[21] M. Jung, Y. Jang, S. Choi, J. Kim, J.-H. Kim, and B.-T. Zhang, "Background-aware moment detection for video moment retrieval," in *Proceedings of the Winter Conference on Applications of Computer Vision*, pp. 8575–8585, February 2025.

[22] J. Nam, D. Ahn, D. Kang, S. J. Ha, and J. Choi, "Zero-shot natural language video localization," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 1470–1479, 2021.

[23] X. Deng, H. Tang, H. Jiang, Q. Zheng, and J. Zhu, "Boundary-aware temporal dynamic pseudo-supervision pairs generation for zero-shot natural language video localization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 2717–2725, 2025.

[24] P. Bao, C. Kong, S. Yang, Z. Shao, X. Jiang, B. P. Ng, M. H. Er, and A. Kot, "Vid-group: Temporal video grounding pretraining from unlabeled videos in the wild," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 20541–20550, 2025.

[25] W.-L. Chiang, Z. Li, Z. Lin, Y. Sheng, Z. Wu, H. Zhang, L. Zheng, S. Zhuang, Y. Zhuang, J. 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)*, 2023.

[26] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[27] M. Maaz, H. Rasheed, S. Khan, and F. S. Khan, "Video-chatgpt: Towards detailed video understanding via large vision and language models," in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL 2024)*, 2024.

[28] K. Li, Y. He, Y. Wang, Y. Li, W. Wang, P. Luo, Y. Wang, L. Wang, and Y. Qiao, "Videochat: Chat-centric video understanding," *arXiv preprint arXiv:2305.06355*, 2023.

[29] E. Song, W. Chai, G. Wang, Y. Zhang, H. Zhou, F. Wu, H. Chi, X. Guo, T. Ye, Y. Zhang, *et al.*, "Moviechat: From dense token to sparse memory for long video understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18221–18232, 2024.

[30] B. Huang, X. Wang, H. Chen, Z. Song, and W. Zhu, "Vtimellm: Empower llm to grasp video moments," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14271–14280, 2024.

[31] Y. Guo, J. Liu, M. Li, X. Tang, X. Chen, and B. Zhao, "Vtg-llm: Integrating timestamp knowledge into video llms for enhanced video temporal grounding," *arXiv preprint arXiv:2405.13382*, 2024.

[32] S. Ren, L. Yao, S. Li, X. Sun, and L. Hou, "Timechat: A time-sensitive multimodal large language model for long video understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14313–14323, 2024.

[33] X. Zeng, K. Li, C. Wang, X. Li, T. Jiang, Z. Yan, S. Li, Y. Shi, Z. Yue, Y. Wang, *et al.*, "Timesuite: Improving mllms for long video understanding via grounded tuning," *arXiv preprint arXiv:2410.19702*, 2024.

[34] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi, *et al.*, "Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[35] K. Feng, K. Gong, B. Li, Z. Guo, Y. Wang, T. Peng, J. Wu, X. Zhang, B. Wang, and X. Yue, "Video-r1: Reinforcing video reasoning in mllms," *arXiv preprint arXiv:2503.21776*, 2025.

[36] X. Li, Z. Yan, D. Meng, L. Dong, X. Zeng, Y. He, Y. Wang, Y. Qiao, Y. Wang, and L. Wang, "Videochat-r1: Enhancing spatio-temporal perception via reinforcement fine-tuning," *arXiv preprint arXiv:2504.06958*, 2025.

[37] Z. Yan, X. Li, Y. He, Z. Yue, X. Zeng, Y. Wang, Y. Qiao, L. Wang, and Y. Wang, "Videochat-r1. 5: Visual test-time scaling to reinforce multimodal reasoning by iterative perception," *arXiv preprint arXiv:2509.21100*, 2025.

[38] A. Y. Qwen, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Li, D. Liu, F. Huang, H. Wei, *et al.*, "Qwen2. 5 technical report," *arXiv preprint*, 2024.

[39] Z. Yue, K. Upasani, X. Yang, S. Ge, S. Nie, Y. Mao, Z. Liu, and D. Wang, "Dr. zero: Self-evolving search agents without training data," *arXiv preprint arXiv:2601.07055*, 2026.

[40] C. Huang, W. Yu, X. Wang, H. Zhang, Z. Li, R. Li, J. Huang, H. Mi, and D. Yu, "R-zero: Self-evolving reasoning llm from zero data," *arXiv preprint arXiv:2508.05004*, 2025.

[41] A. Zhao, Y. Wu, Y. Yue, T. Wu, Q. Xu, M. Lin, S. Wang, Q. Wu, Z. Zheng, and G. Huang, "Absolute zero: Reinforced self-play reasoning with zero data," *arXiv preprint arXiv:2505.03335*, 2025.

[42] Z. Li, H. Du, C. Huang, X. Wu, L. Yu, Y. He, J. Xie, X. Wu, Z. Liu, J. Zhang, *et al.*, "Mm-zero: Self-evolving multi-model vision language models from zero data," *arXiv preprint arXiv:2603.09206*, 2026.

[43] Y. He, C. Huang, Z. Li, J. Huang, and Y. Yang, "Visplay: Self-evolving vision-language models from images," *arXiv preprint arXiv:2511.15661*, 2025.

[44] O. Thawakar, S. Venkatraman, R. Thawkar, A. Shaker, H. Cholakkal, R. M. Anwer, S. Khan, and F. Khan, "Evolmm: Self-evolving large multimodal models with continuous rewards," *arXiv preprint arXiv:2511.16672*, 2025.

[45] S. Bai, K. Chen, X. Liu, J. Wang, W. Ge, S. Song, K. Dang, P. Wang, S. Wang, J. Tang, *et al.*, "Qwen2. 5-vl technical report," *arXiv preprint arXiv:2502.13923*, 2025.

[46] M. Tschannen, A. Gritsenko, X. Wang, M. F. Naeem, I. Alabdulmohsin, N. Parthasarathy, T. Evans, L. Beyer, Y. Xia, B. Mustafa, *et al.*, "Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features," *arXiv preprint arXiv:2502.14786*, 2025.

[47] S.-Y. Liu, X. Dong, X. Lu, S. Diao, P. Belcak, M. Liu, M.-H. Chen, H. Yin, Y.-C. F. Wang, K.-T. Cheng, *et al.*, "Gdpo: Group reward-decoupled normalization policy optimization for multi-reward rl optimization," *arXiv preprint arXiv:2601.05242*, 2026.

[48] Y. Wang, X. Meng, J. Liang, Y. Wang, Q. Liu, and D. Zhao, "Hawkeye: Training video-text llms for grounding text in videos," *arXiv preprint arXiv:2403.10228*, 2024.

[49] Y. Guo, J. Liu, M. Li, Q. Liu, X. Chen, and X. Tang, "Trace: Temporal grounding video llm via causal event modeling," *arXiv preprint arXiv:2410.05643*, 2024.

[50] S. Pramanick, E. Mavroudi, Y. Song, R. Chellappa, L. Torresani, and T. Afouras, "Enrich and detect: Video temporal grounding with multimodal llms," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 24297–24308, 2025.

[51] X. Li, Y. Wang, J. Yu, X. Zeng, Y. Zhu, H. Huang, J. Gao, K. Li, Y. He, C. Wang, *et al.*, "Videochat-flash: Hierarchical compression for long-context video modeling," *arXiv preprint arXiv:2501.00574*, 2024.

[52] D.-A. Huang, S. Liao, S. Radhakrishnan, H. Yin, P. Molchanov, Z. Yu, and J. Kautz, "Lita: Language instructed temporal-localization assistant," *arXiv preprint arXiv:2403.19046*, 2024.

[53] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, *et al.*, "Llava-onevision: Easy visual task transfer," *arXiv preprint arXiv:2408.03326*, 2024.

[54] H. Liu, C. Li, Y. Li, B. Li, Y. Zhang, S. Shen, and Y. J. Lee, "Llava-next: Improved reasoning, ocr, and world knowledge," January 2024.

[55] P. Zhang, X. Dong, Y. Zang, Y. Cao, R. Qian, L. Chen, Q. Guo, H. Duan, B. Wang, L. Ouyang, *et al.*, "Internlm-xcomposer-2.5: A versatile large vision language model supporting long-contextual input and output," *arXiv preprint arXiv:2407.03320*, 2024.

[56] B. Lin, B. Zhu, Y. Ye, M. Ning, P. Jin, and L. Yuan, "Video-llava: Learning united visual representation by alignment before projection," *arXiv preprint arXiv:2311.10122*, 2023.

[57] Y. Yao, T. Yu, A. Zhang, C. Wang, J. Cui, H. Zhu, T. Cai, H. Li, W. Zhao, Z. He, *et al.*, "Minicpm-v: A gpt-4v level mllm on your phone," *arXiv preprint arXiv:2408.01800*, 2024.

[58] M. Abdin, J. Aneja, H. Behl, S. Bubeck, R. Eldan, S. Gunasekar, M. Harrison, R. J. Hewett, M. Javaheripi, P. Kauffmann, *et al.*, "Phi-4 technical report," *arXiv preprint arXiv:2412.08905*, 2024.

[59] B. He, H. Li, Y. K. Jang, M. Jia, X. Cao, A. Shah, A. Shrivastava, and S.-N. Lim, "Ma-lmm: Memory-augmented large multimodal model for long-term video understanding," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13504–13514, 2024.

[60] M. Cai, J. Yang, J. Gao, and Y. J. Lee, "Matryoshka multimodal models," *arXiv preprint arXiv:2405.17430*, 2024.

[61] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4566–4575, 2015.

[62] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pp. 311–318, 2002.

[63] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text summarization branches out*, pp. 74–81, 2004.

[64] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," in *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*, pp. 3982–3992, 2019.

[65] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, *et al.*, "Qwen3-vl technical report," *arXiv preprint arXiv:2511.21631*, 2025.

[66] W. Wang, Z. Gao, L. Gu, H. Pu, L. Cui, X. Wei, Z. Liu, L. Jing, S. Ye, J. Shao, *et al.*, "Internvl3. 5: Advancing open-source multimodal models in versatility, reasoning, and efficiency," *arXiv preprint arXiv:2508.18265*, 2025.

[67] L. Qian, J. Li, Y. Wu, Y. Ye, H. Fei, T.-S. Chua, Y. Zhuang, and S. Tang, "Momentor: Advancing video large language model with fine-grained temporal reasoning," *arXiv preprint arXiv:2402.11435*, 2024.

[68] H. Wang, Z. Xu, Y. Cheng, S. Diao, Y. Zhou, Y. Cao, Q. Wang, W. Ge, and L. Huang, "Grounded-videollm: Sharpening fine-grained temporal grounding in video large language models," *arXiv preprint arXiv:2410.03290*, 2024.

[69] C. Fu, Y. Dai, Y. Luo, L. Li, S. Ren, R. Zhang, Z. Wang, C. Zhou, Y. Shen, M. Zhang, *et al.*, "Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis," *arXiv preprint arXiv:2405.21075*, 2024.

[70] Y. Liu, S. Li, Y. Liu, Y. Wang, S. Ren, L. Li, S. Chen, X. Sun, and L. Hou, "Tempcompass: Do video llms really understand videos?," *arXiv preprint arXiv:2403.00476*, 2024.

[71] J. Skalse, N. Howe, D. Krasheninnikov, and D. Krueger, "Defining and characterizing reward gaming," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9460–9471, 2022.

[72] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou, *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24824–24837, 2022.

[73] A. Singh, A. Fry, A. Perelman, A. Tart, A. Ganesh, A. El-Kishky, A. McLaughlin, A. Low, A. Ostrow, A. Ananthram, *et al.*, "Openai gpt-5 system card," *arXiv preprint arXiv:2601.03267*, 2025.

[74] R. Zeng, W. Huang, C. Gan, M. Tan, Y. Rong, P. Zhao, and J. Huang, "Graph convolutional networks for temporal action localization," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7093–7102, 2019.

# Appendix

In this section, we provide details that are not included in the main manuscript due to the page limit.

## A Details on EvoGround

### A.1 Group reward-Decoupled Normalization Policy Optimization (GDPO)

We describe GDPO [47], an RL optimization algorithm derived from GRPO [34]. We first briefly review GRPO. Given a prompt $p_i$, the model samples $G$ candidate responses $\{o_1, \ldots, o_G\}$ from the behavior policy $\pi_{\theta_{\text{old}}}$. GRPO computes a group-normalized advantage $A_{\text{sum}}^{(i,j)}$ by first summing reward components into $R_{\text{sum}}^{(i,j)}$ and then normalizing:

$$A_{\text{sum}}^{(i,j)} = \frac{R_{\text{sum}}^{(i,j)} - \text{mean}\{R_{\text{sum}}^{(i,1)}, \ldots, R_{\text{sum}}^{(i,G)}\}}{\text{std}\{R_{\text{sum}}^{(i,1)}, \ldots, R_{\text{sum}}^{(i,G)}\}}, \quad R_{\text{sum}}^{(i,j)} = R_1^{(i,j)} + \cdots + R_k^{(i,j)}. \tag{11}$$

GRPO then optimizes the policy to favor responses with higher relative advantage. However, such naively summing rewards overlooks differences in their difficulty and scale, often biasing optimization toward easier objectives while neglecting more challenging ones [71]. GDPO addresses this by reversing the order: it normalizes each reward component *individually before* aggregation, rather than after. Specifically, for each reward $R_k$, GDPO computes a per-reward normalized advantage $A_k^{(i,j)}$:

$$A_k^{(i,j)} = \frac{R_k^{(i,j)} - \text{mean}\{R_k^{(i,1)}, \ldots, R_k^{(i,G)}\}}{\text{std}\{R_k^{(i,1)}, \ldots, R_k^{(i,G)}\}}, \tag{12}$$

and then combines them with defined weights before a final normalization:

$$A_{\text{sum}}^{(i,j)} = w_1 A_1^{(i,j)} + w_2 A_2^{(i,j)} + \cdots + w_k A_k^{(i,j)}, \tag{13}$$

$$\hat{A}_{\text{sum}}^{(i,j)} = \frac{A_{\text{sum}}^{(i,j)} - \text{mean}\{A_{\text{sum}}^{(i',j')} | i' \in \mathcal{B}, j' = 1, ..., G\}}{\text{std}\{A_{\text{sum}}^{(i',j')} | i' \in \mathcal{B}, j' = 1, ..., G\}}, \tag{14}$$

which applies batch-wise ($\mathcal{B}$) advantage normalization. This design enables prioritization of more critical objectives and is particularly suitable for multi-reward settings. We refer the reader to [34] and [47] for the full derivations of GRPO and GDPO, respectively.

### A.2 Additional Implementation Details.

**Hyperparameters.** For both agents, we use the AdamW optimizer with a learning rate of 1e-6. We set the number of rollouts (G) to 2. We use a batch size of 1 with gradient accumulation of 4. We set the max token lengths to 512 for the proposer and to 200 for the solver. This considers their different role, as the proposer likely outputs longer answers than the solver since we require both natural language queries and corresponding timestamps. For the proposer, we set $\tau_{\text{length}} = 3$ and $\gamma = 30$. No labeled validation set is used at any stage, since the setting assumes no annotations, and no benchmark score was consulted for any hyperparameter decision. The $\delta$ schedule ($0 \rightarrow 0.3 \rightarrow 0.5$) was chosen from training-internal signals alone: with $\delta$ fixed at 0.3 the accuracy and feedback reward curves stagnated, and raising $\delta$ across iterations restored a healthy progression (cf. Figure 3). The reward weights and $\tau_{\text{length}}$ were set to conventional values without tuning, and identical hyperparameters are used across all five benchmarks.

**Full prompts.** In Figure 6, we provide a designed prompt for the proposer and solver, respectively. The proposer is encouraged to generate query–moment samples under valid conditions, and the solver aims to temporally ground the video from the given query sentence. Here, we set $N$ to 4 for the proposer. If the proposer generates valid query–moment pairs less than $N$, the remaining entries are padded with zero moments to penalize insufficient sampling and encourage the generation of multiple pairs. For instance, if the proposer generates only one valid pair (*e.g.*, covering the entire video), the format reward is reduced to 0.25 since the remaining three pairs are assigned 0.

**Memory peak.** During trainnig, the peak memory is $\sim$60 GB/GPU (proposer) and $\sim$30 GB/GPU (solver).

Table 10: **Ablation on the variants of inter-consistency.**

| Intra | TVGBench | | |
|---|---|---|---|
| | R1@0.3 | R1@0.5 | R1@0.7 |
| hard | 41.7 | 27.3 | 15.0 |
| exp | 40.8 | 26.9 | 14.3 |
| softmax | **42.1** | **28.5** | **15.3** |

Table 11: **Results on Video-MME and TempCompass.** * denotes our reproduced results.

| Method | Video-MME | TempCompass |
|---|---|---|
| | w/o subs | multi-choice QA |
| Qwen2.5-VL-7B* [45] | 61.4 | 71.3 |
| Time-R1* [13] | 61.2 | 71.2 |
| **EvoGround (Ours)** | **62.3** | **71.5** |

Table 12: **Results on Charades-STA, ActivityNet, and TVGBench.**

| Method | Charades-STA | | | | ActivityNet-Captions | | | | TVGBench | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | R1@0.3 | R1@0.5 | R1@0.7 | mIoU | R1@0.3 | R1@0.5 | R1@0.7 | mIoU | R1@0.3 | R1@0.5 | R1@0.7 | mIoU |
| **EvoGround (2.5K)** | 77.2 | 60.5 | 35.5 | 53.1 | 62.9 | 43.6 | 25.0 | 43.9 | 42.1 | 28.5 | 15.3 | 29.4 |
| **EvoGround (7.5K)** | 81.7 | 67.7 | 45.6 | 60.3 | 64.8 | 44.3 | 25.8 | 44.7 | 41.7 | 28.6 | 15.1 | 29.5 |
| Δ | +4.5 | +7.2 | +10.1 | +7.2 | +1.9 | +0.7 | +0.8 | +0.8 | -0.4 | +0.1 | -0.2 | +0.1 |

### A.3 Details on the Reward Designs

**SigLIP-2.** SigLIP-2 [46] is a widely used vision-language encoder that extends the success of the original SigLIP with enhanced pre-training objectives. We use the 0.4B-scale model, which is lightweight and incurs negligible computations, for the consistency reward during training.

**Intra-Consistency.** In Equation 5, the exponential formulation is designed to map the variance of frame-level similarities to a sharply discriminative score. Since similarity values lie in $[-1, 1]$, the resulting variance $\sigma_n$ is bounded within a limited range. In the ideal case, where all frames within a moment are perfectly aligned with the query, the variance becomes zero, yielding $\exp(0) = 1$. In contrast, highly inconsistent moments exhibit large fluctuations in similarity (*i.e.*, alternating between 1 and $-1$), resulting in $\sigma_n \approx 1$. Such cases are mapped to a near-zero value via the exponential function, $\exp(-1 \times 30) \approx 9.3 \times 10^{-14}$. This design ensures that well-aligned moments are preserved with high scores, while even moderate inconsistencies are strongly penalized, providing a clear separation between coherent and incoherent moment predictions. Additionally, we determine $|m_n|$ in Equation 5 as:

$$|m_n| = \text{clip}\left((e_n - s_n) \cdot \rho, T_{\min}, T_{\max}\right), \tag{15}$$

where $\rho = 1$, $T_{\min} = 8$, and $T_{\max} = 32$. This ensures that longer moments are represented with more sampled frames.

**Variants of Inter-Consistency.** We consider two variants of inter-consistency: a hard ranking-based formulation and an exponential formulation:

$$\text{Inter}_n^{\text{hard}} = \frac{1}{N} \sum_{j=1}^{N} \mathbb{I}\big(\mu_n \geqslant \mu_{n,j}\big), \tag{16}$$

which measures the fraction of candidates that the $n$-th sample outperforms, capturing its relative ranking within the group. We also design the function with an exponential formulation:

$$\text{Inter}_n^{\text{exp}} = \begin{cases} 1 - \exp\left(-\gamma_{\text{inter}} \cdot \frac{1}{N-1} \sum_{j \neq n} \max(0,\, \mu_n - \mu_{n,j})\right), & \text{if } N > 1, \\ 0, & \text{otherwise.} \end{cases} \tag{17}$$

which provides a smooth and more sensitive measure of relative superiority, where larger margins between $\mu_n$ and other candidates lead to exponentially higher scores. We set $\gamma_{\text{inter}}$ to 30. In Table 10, we compare the performance of applying these inter-consistency variants and confirm that computing the inter-consistency using a softmax demonstrates the best performance. This is likely due to its smooth and more discriminative treatment of relative differences among candidates compared to other variants.

### A.4 Impact of Training Data and Comparison with Time-R1

A potential concern is that the performance of EvoGround may be influenced by the underlying video corpus used for training. We acknowledge that the choice of data can affect model performance.

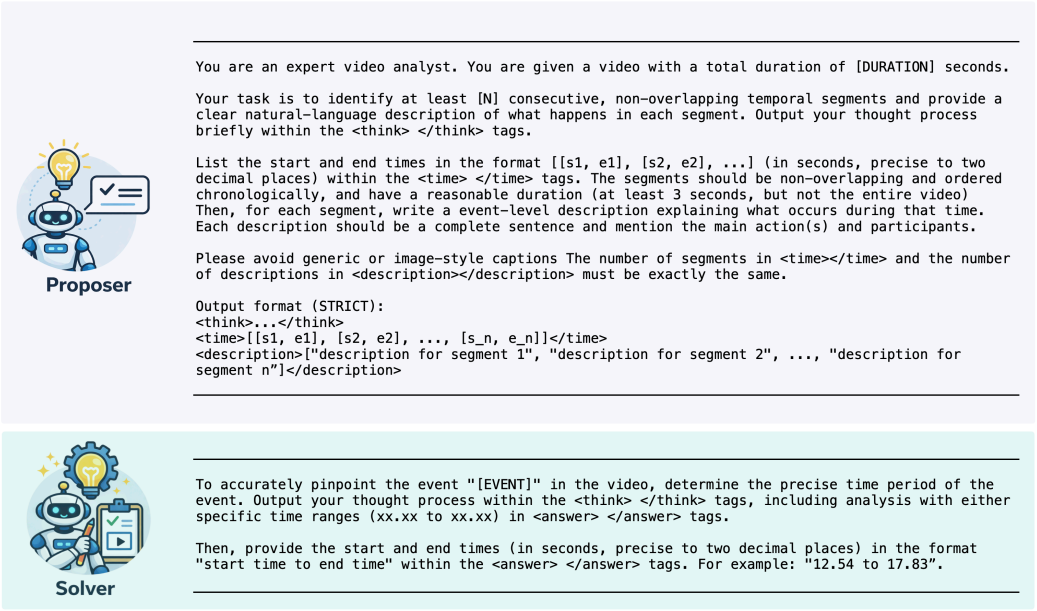


Figure 6: **Prompt designs of EvoGround.** We show the prompts used for the proposer (top) and solver (bottom). The proposer is instructed to generate consecutive, non-overlapping query–moment pairs from a raw video, while the solver is instructed to localize a given query within the video.

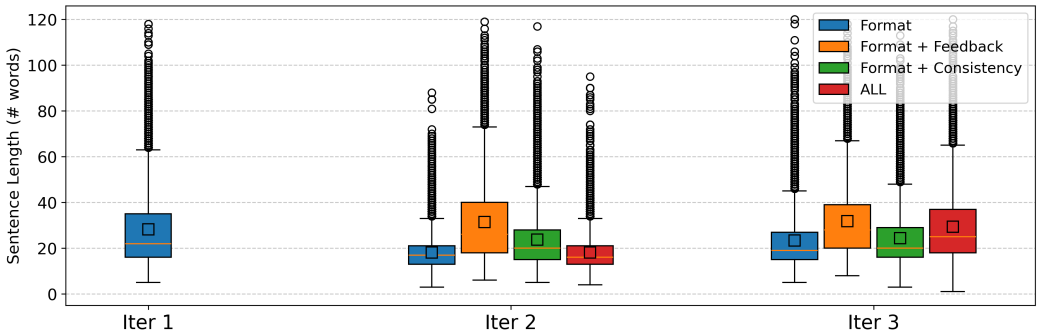


Figure 7: **Query length distribution across iterations.** We visualize the query length distributions across different reward configurations and iterations. As previously discussed in Section 5, the feedback reward increases the length of queries compared to others.

To control for this factor, we note that both EvoGround and Time-R1 [13] are trained on the same underlying video corpus (*i.e.*, TimeRFT). Therefore, any performance differences cannot be attributed solely to dataset selection.

Despite sharing the same video corpus, EvoGround differs from Time-R1 in several aspects. First, EvoGround does not rely on any human-annotated data, whereas Time-R1 is trained with labeled supervision. Second, EvoGround does not require a cold-start fine-tuning stage or the construction of curated chain-of-thought (CoT) [72] examples, both of which are involved in Time-R1 to guide training. Time-R1 further requires dynamic hard sampling during its training that mines hard samples (*i.e.*, low-IoU samples) on a curated dataset for each training epoch. In contrast, EvoGround learns directly from raw videos through a self-evolving proposer–solver loop driven by RL.

Beyond, EvoGround is more data-efficient than existing models, achieving strong performance with only a small amount of raw video data. As mentioned in Section 4.2, existing models typically rely on multi-stage training pipelines with curated datasets tailored for each stage. For example, VTimeLLM [30] adopts a three-stage framework (feature alignment, boundary perception, and

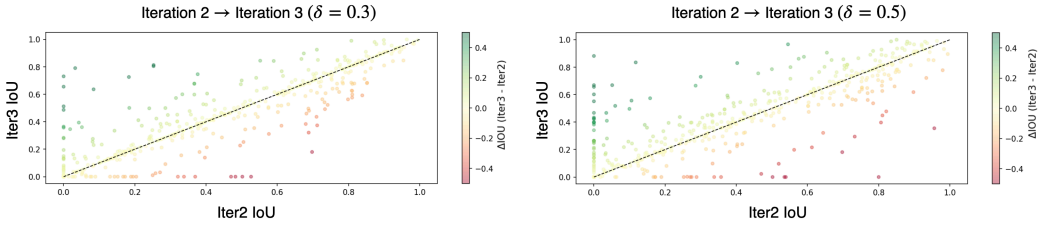


Figure 8: **Per-sample IoU improvements under different thresholds $\delta$.** The dashed line represents no change, and the upper-left regions represent improved cases.

instruction tuning), while Grounded-VideoLLM [68] employs a similar multi-stage pipeline that involves large-scale caption supervision (*i.e.*, 1.28M video-caption pairs) in early stages.

Taken together, these differences suggest that the improvement of EvoGround stems from the proposed self-evolution framework rather than reliance on curated video quality or dataset-specific advantages. While the underlying corpus still plays a role, our results demonstrate that strong temporal grounding performance can be achieved without annotated data or additional manual curation.

# B Further Analysis

**Results on VideoQA benchmarks.** We evaluate EvoGround on representative video question answering (QA) benchmarks, including Video-MME [69] and TempCompass [70] in Table 11. We confirm that EvoGround does not disrupt the backbone (*i.e.*, Qwen2.5-VL-7B) and demonstrates a slight improvement.

**Query length distributions across iterations and different reward configurations.** In Figure 7, we visualize the query length distribution for each reward configuration across iterations. As previously mentioned, the feedback reward enriches query descriptiveness, showing the highest average query length.

**Per-sample comparison across different $\delta$.** In Figure 8, we compare per-sample improvements across different $\delta$ on TVGBench. $\delta = 0.5$ results in a denser distribution in the high-IoU region (*i.e.*, upper-left), compared to $\delta = 0.3$, along with higher average improvements (+1.29% vs. +0.21%).

**Evaluation on TemporalBench.** As shown in Figure 9, to evaluate on TemporalBench [16], we first generate query–moment pairs from given videos and concatenate them into a single paragraph. We then compare it with the ground-truth sentence and confirm that EvoGround effectively captures consecutive video events.

**Visualization of generated data.** We also visualize generated data from the proposer at the third iteration in Figure 10. Each query is well-aligned with the corresponding frames within the moment and densely covers throughout the video.

**Results of EvoGround with scaling up video data.** Extending the results in Table 4, we report the performance of EvoGround (7.5K) on additional benchmarks including Charades-STA in Table 12. While we observe some improvements across benchmarks and metrics, the gains are modest, particularly on out-of-distribution datasets such as TVGBench and ActivityNet-Captions. This is expected, as the additional training videos are sourced from Charades-STA, which primarily consists of short videos. Consequently, incorporating such data helps the model better handle similar domains, but is less effective for broader generalization.

# C Additional Experiments

## C.1 Generated pairs become more faithful across iterations

We evaluate both EvoGround-generated query–moment pairs and human-annotated pairs from TimeRFT using two external judges, Qwen3-VL-32B [65] and GPT-5 [73], neither of which par-



**Ground-truth**: Women dressed in blue uniforms and hats walk in two lines while holding hands. They are smiling and waving at the crowd of people who are cheering them on by clapping their hands on the side. A group of men wearing heavy dark jackets and hat walk through while shaking hands and smiling at the crowd.

**EvoGround:** Students in robes walk through a grand hall, greeted by applause from others. The group of students continues walking, with some students in uniform hats and robes. The students pass by a group of adults in robes, who are clapping and smiling. The students continue walking, with the crowd of students and adults still applauding and smiling.

Figure 9: **Captioning results on TemporalBench.**

ticipates in training. We report the verification pass rate that measures the proportion of pairs for which the judge confirms that the query accurately describes the corresponding moment. Although generated pairs by EvoGround and ground-truth annotations share the same source videos, they describe different events and temporal segments. Thus, this analysis is not a direct comparison of data quality; a higher verification pass rate should not be interpreted as evidence of superior data quality. For efficiency, we sample 1,000 and 100 pairs for evaluation by Qwen3-VL-32B and GPT-5, respectively. In Figure 11, pass rates increase monotonically across iterations under both judges. The untrained proposer starts below 60%, but reaches 70.3% by iteration 3, suggesting that our proposer and its data evolves across iterations, rather than remaining static or becoming increasingly unfaithful. Since both our proposer and the external judges operate exclusively on sampled video frames, human annotations that rely on non-visual cues, such as audio or speech, may receive lower verification scores despite being correctly annotated.

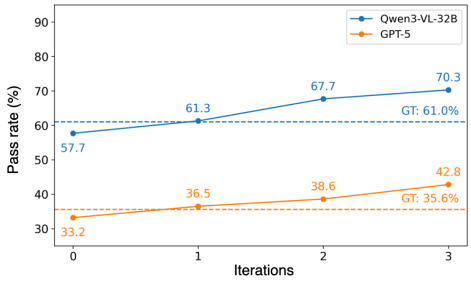


Figure 11: **Verification pass rate across different iterations and judge models.**

### C.2 Controls: Does Co-evolution Drive the Gains?

Section 5 reports the headline comparison (Table 6). We give the settings here. All controls use the same 2.5K TimeRFT videos, the same Qwen2.5-VL-7B backbone and GRPO as the solver objective, so that only the data pipeline varies. **B0** (random moment + captioning) samples moments uniformly from each video and prompts the frozen base model to caption each sampled clip, yielding pseudo query–moment pairs with no proposer at all. **B1** (one-shot pseudo-label) uses the frozen base model as the proposer, generating pairs once with a tailored prompt and no solver feedback, after which the solver is trained via GRPO. We keep GRPO rather than SFT so that the objective is identical across rows. **B2** (frozen proposer) bootstraps the proposer for one iteration with the format reward only and then freezes it, training the solver on its fixed output for the full budget. These correspond to a basic caption-and-pseudo-label pipeline, one-shot pseudo query–moment generation, and a frozen proposer without iterative feedback, respectively. Matching is on solver update steps, so every row receives the same number of gradient steps on the solver. The only difference is whether the generator is updated.

### C.3 Isolating the Reward Design from the Self-Play Scaffold

To separate our contribution from the challenger–solver scaffold itself, we hold the scaffold fixed and swap in R-Zero's [40] reward recipe. Since R-Zero's released models are text-only reasoning LLMs, direct initialization from R-Zero is not applicable. We therefore train an R-Zero-style proposer with the same GDPO machinery, the same backbone and the same TimeRFT videos, but with R-Zero's

**gA5sihQiKrE_00:00:02:700_00:00:40:600.mp4 (37.9 s)**




[0.0s, 3.9s] The video begins with a collage of cooking scenes, introducing the topic of selecting the right cookware for an induction cooktop.




[3.9s, 9.9s] A black induction cooktop appears with various cookware items, including pots and pans, being arranged on it, demonstrating the compatibility of different cookware.



[9.9s, 19.8s] A close-up of a black frying pan with a handle is shown, followed by a shot of a stainless steel pan being turned, highlighting the features of the cookware.



[19.8s, 37.9s] The video concludes with a demonstration of the pans and pots being placed on the induction cooktop, showcasing the process of using the cookware effectively.

Figure 10: **Generated query–moment pairs from EvoGround.**

reward composition: format plus an uncertainty-band solvability term $1 - 2 \cdot |\hat{p} - 0.5|$, where $\hat{p}$ is the solve rate over the completion's proposed windows, each binarized at $\mathrm{tIoU} \geqslant 0.5$, and with no consistency term. Both proposers are initialized from the iteration-1 checkpoint and trained for 150 optimizer steps, single seed, single GPU, and we score their generated pairs with the external Qwen3-VL-32B judge of Section 5.

Table 13: **Reward design vs. self-play scaffold.** Both rows use the same scaffold, backbone and corpus, and differ only in the proposer's reward composition. Short-budget comparison (150 optimizer steps), not fully trained models.

| Method | Reward composition | Qwen3-VL-32B pass % (conf.) |
|---|---|---|
| R-Zero-style [40] | format + solver-uncertainty band, no faithfulness | 58.2 (0.51) |
| **EvoGround reward** | format + consistency + solver feedback | **63.4 (0.56)** |

Table 14: **Supervised fine-tuning from the self-evolved solver** on Charades-STA (R1@0.3 / R1@0.5 / R1@0.7 / mIoU). Only the initialization differs between the first and last rows.

| Initialization | Charades-STA |
|---|---|
| Qwen2.5-VL + FT | 76.1 / 42.9 / 26.2 / 43.6 |
| Time-R1 [13] (reported) | **82.8 / 72.2 / 50.1 / 60.9** |
| **EvoGround + FT** | 80.9 / **72.9 / 48.9 / 61.3** |

Table 15: **Training cost and supervision.** EvoGround trains with substantially less compute than ED-VTG and fewer GPUs than Time-R1 while requiring no labeled pairs. The annotation cost is not shifted to computation.

| | **EvoGround** | **Time-R1 [13]** | **ED-VTG [50]** |
|---|---|---|---|
| Params | 7B | 7B | 7B |
| External models | SigLIP-2 (0.4B) | - | LLaVA-OneVision (72B) |
| Annotations | **0 labeled pairs** | ∼80K labeled pairs | ∼136K labeled pairs |
| Pipeline stages | no cold-start SFT, no CoT curation, no hard-sample mining | cold-start SFT + dynamic hard-sample mining | SFT |
| Training cost | ∼72 h wall-clock (3 iterations × ∼24 h) ≈288 GPU-hours on 4×A100-80GB | 8×A100 GPUs, training time not reported | ∼2 days on 16 nodes of 8×V100-32GB ≈ 6,144 GPU-hours |
| Peak memory | ∼60 GB/GPU (proposer), ∼30 GB/GPU (solver) | - | - |

The R-Zero-style reward produces less faithful pairs at a matched budget, quantifying the contribution of the VTG-specific reward design beyond the self-play scaffold.

### C.4 Self-Evolved Solver as Initialization for Supervised Fine-Tuning

A natural question is whether the self-evolution loop yields a better starting point for supervised adaptation than the raw backbone. We fine-tune on the 12K Charades-STA training labels starting from our self-evolved solver, keeping GRPO as the learning objective so that only the initialization differs, and compare against identical fine-tuning from the raw backbone.

Initializing from the self-evolved solver substantially outperforms initializing from the raw backbone under an otherwise identical fine-tuning setup (+17.7 mIoU), indicating that the loop provides a stronger starting point for supervised adaptation. The resulting model attains the best mIoU and R1@0.5 overall while forgoing Time-R1's cold-start SFT and dynamic hard-sample mining.

### C.5 Efficiency Comparison with Annotation-Based Competitors

Per-iteration breakdown: proposer ∼14 h, solver ∼10 h, pair generation ∼1 h. Because frames are sampled at 2 FPS and adaptively resized to a fixed pixel budget, per-video inference cost is roughly constant with video length, and cost scales linearly with corpus size (cf. Table 4, 2.5K → 7.5K).

### C.6 Comparison with Pseudo-Label and Zero-Shot Grounding Pipelines

Table 16: **Comparison with annotation-free grounding pipelines** under the zero-shot protocol (R1@0.3 / R1@0.5 / R1@0.7 / mIoU). Numbers for ReCorrect and BTDP are taken from their respective papers. Unlike these methods, which train specialized grounding heads that output temporal intervals only, EvoGround yields a general-purpose Video-LLM that also improves broader video understanding such as dense captioning (Table 8).

| Method | Charades-STA | ActivityNet-Captions |
|---|---|---|
| ReCorrect [24] | 66.5 / 51.2 / 28.5 / 45.6 | 54.7 / 33.4 / 15.2 / 36.0 |
| BTDP [23] | 58.3 / 40.0 / 20.9 / 39.0 | 50.6 / 30.5 / 17.5 / 36.5 |
| **EvoGround** | **77.2 / 60.5 / 35.5 / 53.1** | **62.9 / 43.6 / 25.0 / 43.9** |

ReCorrect pretrains on 50K videos with 200K pseudo labels from a fixed generation pipeline, then filters mismatched pairs and corrects boundaries from a memory of its own predictions during pretraining. The generating pipeline itself is never updated. BTDP likewise relies on a hand-designed

pipeline with no trained components. In contrast, EvoGround mutually evolves both agents on 2.5K videos with no pretraining stage. We exclude R-17pV's reference [A], graph-convolutional temporal action localization [74], from this table because the protocols do not align: it performs closed-set action localization evaluated by mAP rather than language-conditioned grounding evaluated by R1@IoU.

### C.7 Captioning Evaluation Protocol

To verify that the captioning results of Table 3 reflect the model rather than the evaluation pipeline, we evaluate in both directions: (1) the base model is prompted with our proposer's prompt and its per-segment outputs are concatenated through the identical pipeline used for EvoGround, and (2) both models are evaluated under the single-pass protocol of the original TemporalBench release, where each model generates one description of the video.

Table 17: **TemporalBench under both protocols** (Similarity / CIDEr / ROUGE). The gap between protocols is negligible and Similarity is near-saturated. The more discriminative metrics favour EvoGround under both.

| Method | Concatenation | Single-pass |
|---|---|---|
| Qwen2.5-VL-7B [45] | 52.4 / 9.8 / 19.3 | **53.8 / 9.9 / 19.5** |
| **EvoGround** | **53.8 / 11.4 / 20.5** | 53.7 / **11.1 / 19.8** |

### C.8 Preference Study Against Human Annotations on Matched Moments

The pass-rate audit in Section 5 scores generated and human-annotated pairs independently. To compare them on the *same* temporal content, we additionally match each generated pair to its most similar ground-truth pair on the same video by $\mathrm{IoU} \geqslant 0.7$, keeping only matches whose moment lengths differ by less than 30%. This yields at least 300 matched pairs per iteration. We judge a seeded random sample of 100 per iteration. The judge (Qwen3-VL-32B-Instruct) sees the union clip of the two moments with 8 uniformly sampled frames and is asked which query better describes the clip. To control for position bias, each pair is judged twice with the two options swapped. A win or loss is counted only when both orders agree, and disagreements count as ties.

Table 18: **Preference study on temporally matched pairs.** Strict counting: a win requires both presentation orders to agree, and ties count against EvoGround. The judge selected option (a) 45.5–51% of the time, indicating no appreciable position bias.

| Comparison | Win % (strict) | Tie % | Loss % | Win % (decisive only) |
|---|---|---|---|---|
| Iteration 1 vs. GT | 58.0 | 14.0 | 28.0 | 67.4 |
| Iteration 2 vs. GT | 62.0 | 19.0 | 19.0 | 76.5 |
| Iteration 3 vs. GT | 66.0 | 12.0 | 22.0 | 75.0 |

We do not read these results as evidence that generated queries are better than human annotations. Automatic judges are known to favor longer, more detailed descriptions, which disadvantages ground-truth queries that rely on audio or full-video context, and the tie rates (12–19%) themselves support "comparable" over "strictly better." The consistent $\geqslant$50% preference indicates only that, for nearly identical temporal moments, our generated queries are at least comparable to human-written annotations in visual faithfulness and specificity.

### C.9 Pretrained Components and Potential Benchmark Exposure

For transparency, we list every pretrained component: Qwen2.5-VL-7B (plus Qwen2.5-VL-3B, Qwen3-VL-8B and InternVL3.5-4B in the backbone ablation), SigLIP-2 (0.4B) for the consistency reward, and the 2.5K-video TimeRFT corpus. Qwen's pretraining corpus is not fully disclosed, so prior exposure to ActivityNet or Charades videos and captions cannot be categorically excluded. Two observations bound how far this can affect our conclusions. First, the zero-shot backbone performs far worse than EvoGround (28.1 $\rightarrow$ 42.1 R1@0.3 on TVGBench, Table 7, and the Charades-STA rows

of Table 1), so benchmark memorization cannot explain the gains. Second, every RL baseline we compare against (Time-R1, VideoChat-R1/R1.5) shares the same backbone, so any leakage advantage is common-mode across the compared methods.

### C.10 Controlling the Moment-Length Distribution

Several components influence the length distribution of the generated moments: (i) $\tau_{\text{length}}$, the minimum segment length in the validity constraint, (ii) the duration constraints stated in the proposer prompt (Figure 6), and (iii) the frame-budget clip $T_{\min} = 8, T_{\max} = 32$. Empirically, the consistency-reward weight also tightens the distribution ($27.0 \pm 37.4 \rightarrow 19.8 \pm 24.6$, Figure 5), so the framework can be biased toward shorter or longer target moments through these knobs.

## NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Abstract and introduction clearly include the main claims.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses the limitations of the work in Section 6.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

Justification: The content does not include theoretical results, requiring proof.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides detailed experiment settings for reproducibility.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will make all resources publicly available.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies all details for training and testing.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report 95% bootstrap confidence intervals (1,000 resamples with replacement of the test set) for our model in Tables 1 and 2, except for ReXTime, whose ground truth is held on the official test server. Methodology is described in Section 4.2 (Evaluation Metrics).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper provides such details to reproduce the experiments.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research follows the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: While EvoGround is designed for academic video understanding, the ability to automatically retrieve and localize video content without annotations could be misused for surveillance or privacy-invasive monitoring. We acknowledge this risk and note that our model and code will be released under a license prohibiting malicious use.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: We build only on publicly released pre-trained models and datasets, and we release the trained EvoGround model and the generated query–moment pairs under a license prohibiting malicious use (see Question 10), which is the safeguard we rely on for the released assets.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The paper properly left all references.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the trained EvoGround model and generated query–moment pairs as new assets. These will be made publicly available with documentation on training details, intended use, and licensing upon acceptance.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not conduct crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not conduct crowdsourcing.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: LLMs are a core component of EvoGround: both the proposer and solver are built on Qwen2.5-VL-7B [45], a large vision-language model, and are optimized via reinforcement learning to perform video temporal grounding without human annotations.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.