TimePLE: Rethinking Temporal Representation for Video Temporal Grounding

Yuhui Zeng[1,4,*†], Xinyu Mao[2,4,*†], Xiaokun Liu[4,‡✉], Xin Tao[4]
Pengfei Wan[4], Jinfa Huang[3], Jiayi Ji[1], Xiawu Zheng[1,✉]

[1]Media Analytics and Computing Lab, Xiamen University, Xiamen, China
[2]The Chinese University of Hong Kong, HKSAR, China
[3]Department of Computer Science, University of Rochester, Rochester, NY, USA
[4]Kling Team, Kuaishou Technology, China

∗ **Equal Contribution**
‡ **Project Leader**
✉ **Corresponding Authors**
† **This work was conducted during the authors' internships at Kling Team.**

Abstract

Video temporal grounding (VTG) aims to localize the continuous video interval described by a natural-language query. However, current VLM-based methods typically produce this interval indirectly through two endpoint outputs, represented either as discrete timestamp tokens or continuous boundary coordinates. These formulations differ in how endpoints are encoded, but not in what is predicted: the event interval remains a derived object, while interval validity, duration, and interval-level similarity are handled only implicitly. We propose **TimePLE**, which reformulates VTG from endpoint prediction to interval-native grounding by predicting a single joint distribution over valid temporal intervals. TimePLE maps each interval to a point in a canonical position–duration square, where every support point corresponds to a valid span and neighboring points represent geometrically similar intervals. Given a video and query, the VLM generates a single latent <|TIMESPAN|> token whose hidden state is decoded into a joint interval distribution, refined through duration-aware coordinate correction, and converted into continuous boundaries. The same interval representation is used to encode input temporal anchors, aligning video-side temporal evidence with output-side span prediction. We further build a model-assisted curation and human-verification pipeline that produces **90K-scale** grounded samples and corrects **3K-scale** noisy annotations. Experiments across four VTG benchmarks show that TimePLE consistently outperforms endpoint prediction baselines, achieving an **average mIoU of 58.9**, with clear gains on short-duration and medium-duration events.

1 Introduction

Multimodal language models have recently made rapid progress in video understanding [2–4, 20, 21, 29, 34, 38, 39, 42, 43]. However, recognizing video content and grounding it precisely in time remain distinct capabilities. Video temporal grounding (VTG) requires a model to localize the continuous temporal interval described by a natural-language query. At its core, the target is a coherent event interval rather than two timestamps considered separately. Both supervision and evaluation are interval-level: a prediction is judged by its overlap with the complete event moment, whose temporal placement and duration jointly determine localization accuracy. Yet current VLM-based formulations typically obtain this interval from two endpoint outputs, creating a mismatch between the task target and the model output.

Existing methods mainly differ in how temporal endpoints are represented. One common direction represents time as discrete timestamp outputs, including textual timestamps [2, 6, 7, 17, 18, 22, 24, 26, 31, 32, 35, 36, 40, 42] and symbolic time tokens [10–12, 19, 28]. These formulations preserve the autoregressive language interface. However, their token-

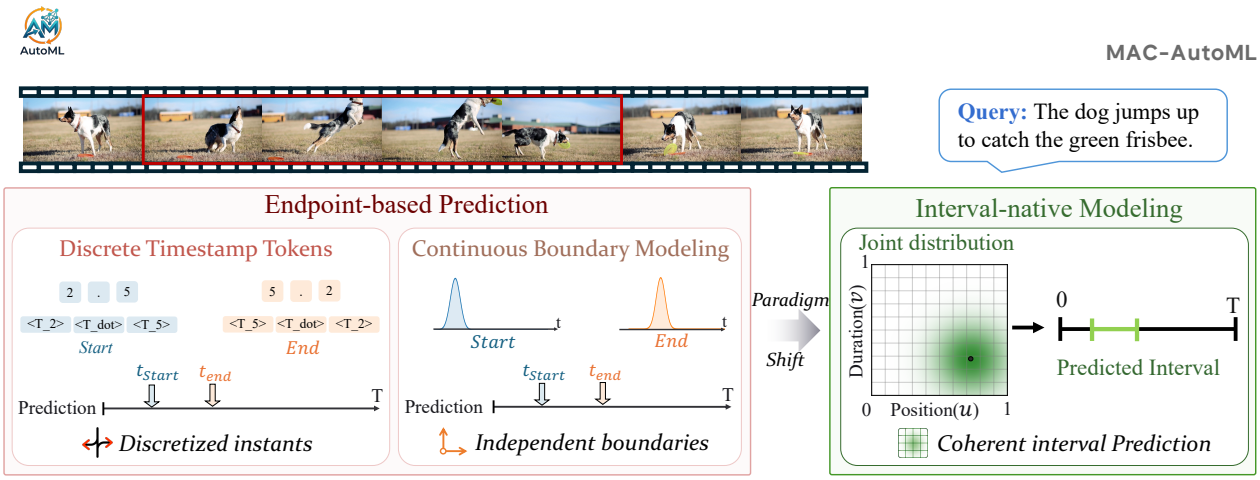


Figure 1. From endpoint prediction to interval-native temporal grounding. Existing methods represent a temporal moment through two endpoint outputs, encoded either as discrete timestamp tokens or continuous boundary coordinates, and assemble the interval afterward. TimePLE instead predicts a single joint distribution over valid temporal intervals, from which the final start and end boundaries are subsequently decoded.

level supervision does not reflect temporal distance on the video timeline: predictions close to and far from the ground-truth moment may both be treated as discrete token errors, despite implying substantially different localization quality. Another direction decodes continuous start and end boundaries [13, 37], providing smoother temporal supervision without discrete time labels. Nevertheless, it retains the same endpoint-centric prediction object. Interval validity must be enforced through ordering constraints or post-processing, duration remains a difference between two coordinates, and interval-level similarity is not directly represented by the native output space. Thus, discrete and continuous endpoint methods differ in how endpoints are encoded, but not in what is predicted: the event interval remains a derived object rather than the model primary output.

Motivated by this perspective, we propose **TimePLE**, which reformulates VTG from predicting two temporal endpoints to predicting a single joint distribution over valid temporal intervals. We parameterize the space of normalized intervals using a canonical position–duration square. For an interval $I = [s, e]$, its duration is represented by $v = e - s$, while $u = s/(1 - v)$ describes the feasible placement of an interval with that duration. This mapping separates how long an event lasts from where an event of that length occurs on the timeline. More importantly, every point in the canonical square corresponds to a valid temporal interval, and neighboring points represent geometrically similar spans. The decoder can therefore assign probability directly to candidate event moments rather than to endpoint pairs.

TimePLE connects this interval-native output space to a VLM through a lightweight latent temporal interface. Given a video and a query, the VLM generates a single <|TIMESPAN|> token whose hidden state parameterizes a joint distribution over the canonical interval square. The expected canonical coordinate is refined through a duration-aware correction and then mapped back to continuous temporal boundaries. On the input side, the temporal coverage of sampled visual units is represented by interval-valued anchors in the same canonical space and inserted through <|TIMESTAMP|> tokens. This shared geometry aligns video-side temporal evidence with output-side interval prediction, while preserving the autoregressive VLM interface.

To support temporally precise supervision, we further construct a multi-model curation and human-verification pipeline that produces **90K-scale** grounded training samples and corrects **3K-scale** noisy benchmark annotations. Across three VLM backbones and four VTG benchmarks, TimePLE consistently outperforms matched timestamp-based baselines. On Qwen3-VL-8B, it achieves an **average mIoU of 58.9**, improving the matched Timestamp-SFT baseline by 4.1 points. The gains are particularly clear on short-duration and medium-duration events, whose interval overlap is more sensitive to small localization errors. Representation-level analyses further show that TimePLE draws more coherent evidence from the target moment and produces a more localized probability landscape over candidate intervals.

Our contributions are summarized as follows:

- We reformulate VLM-based video temporal grounding from endpoint prediction to interval-native grounding. Instead of temporal endpoints prediction, the proposed formulation makes a joint distribution over valid intervals the primary prediction object.

- We introduce **TimePLE**, which parameterizes valid temporal intervals in a canonical position–duration square and decodes the hidden state of a single <|TIMESPAN|> token into an interval distribution.
- We build a multi-model annotation and model-assisted verification pipeline, yielding **90K-scale** grounded training samples and correcting **3K-scale** noisy annotations from existing VTG evaluation benchmarks.
- We evaluate TimePLE across four VTG benchmarks. TimePLE achieves **58.9 average mIoU** across four VTG benchmarks and performs favorably against endpoints prediction methods, with clear gains on short-duration and medium-duration events.

2 Related Work

2.1 VLM-based Temporal Video Grounding

Recent VLM-based VTG methods make time accessible to multimodal language models through timestamp-aware video representations, relative or absolute time tokens, frame-number prompts, interleaved timestamp markers, and continuous time decoders [6, 10–13, 19, 24, 26, 28, 35, 37]. These methods allow instruction-following VLMs to refer to temporal locations through the language interface. Textual timestamps reuse the native vocabulary of language models, while temporal-token methods introduce symbolic time indices for finer temporal reference. Continuous temporal decoders further replace discrete outputs with smoother boundary-level supervision. Despite these differences, existing formulations mainly change how temporal endpoints are represented rather than what is predicted. In contrast, TimePLE makes the interval itself the primary prediction object by decoding a single joint distribution over a canonical space of valid temporal spans.

2.2 Datasets and Benchmark Quality

VTG datasets and benchmarks such as Charades-STA, ActivityNet-Captions, TACoS, DiDeMo, and QVHighlights have supported temporal grounding across diverse video domains, query styles, and evaluation settings [1, 8, 9, 14, 15, 25, 27, 44]. However, VTG is sensitive to boundary quality, query ambiguity, and annotation completeness. Recent work has highlighted data quality issues in existing VTG benchmarks and introduced re-annotated data for more reliable training and evaluation [41]. Our work follows this direction but focuses on supervision for latent interval decoding. We build a multi-model curation and human-verification pipeline that produces caption-grounded training samples and corrects noisy temporal annotations from existing VTG benchmarks, providing reliable supervision for learning canonical interval distributions.

3 Method

3.1 Overview and Problem Formulation

Given a video V with duration T and a natural-language query q, video temporal grounding aims to localize the temporal segment in V that semantically corresponds to q. We denote the target segment in seconds as $\widetilde{I} = [t_s, t_e]$, where $0 \leq t_s \leq t_e \leq T$. For model prediction and supervision, we normalize the boundaries by the video duration and obtain $I = [s, e]$, where $s = t_s/T$, $e = t_e/T$, and $0 \leq s \leq e \leq 1$. TimePLE preserves the autoregressive interface of VLMs while reformulating temporal localization as interval-distribution prediction. As illustrated in Fig. 2, the model generates a response containing <|TIMESPAN|>, whose hidden state parameterizes a latent interval distribution. TimePLE decodes the hidden state into a normalized interval $\hat{I} = [\hat{s}, \hat{e}]$ in a canonical interval space and rescales it to seconds.

3.2 Canonical Interval Square

To parameterize a joint distribution over valid temporal intervals, TimePLE maps the feasible interval space to a canonical position–duration square. Let $d = e - s$ denote the normalized duration. For $d < 1$, the feasible start position lies in $[0, 1 - d]$. We define the canonical coordinate as

$$\mathcal{C}(I) = (u, v), \qquad u = \frac{s}{1 - d}, \qquad v = d. \tag{1}$$

Here, v directly represents the interval duration, while u represents the relative placement of an interval with that duration within its feasible start range. For the full-video interval, whose feasible start range collapses to $s = 0$, we

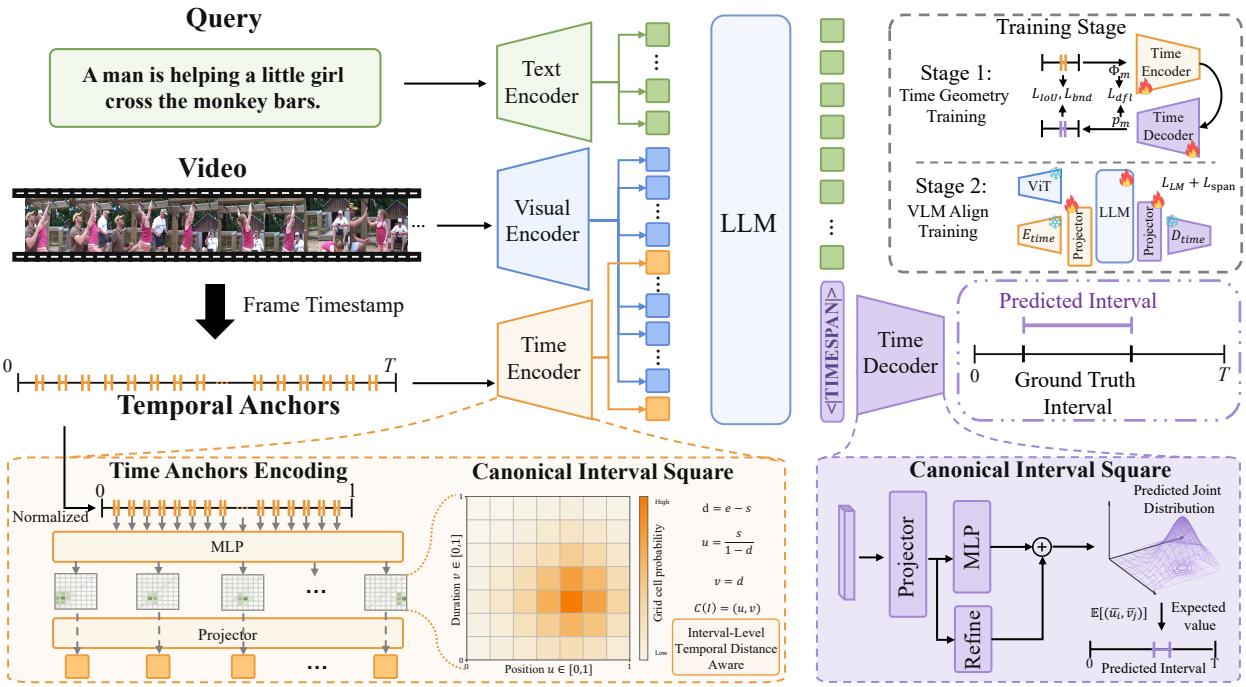


Figure 2. Overview of TimePLE. Temporal anchors are encoded in a canonical position–duration space and inserted into the VLM through <|TIMESTAMP|> tokens. The hidden state of a generated <|TIMESPAN|> token is decoded into a joint distribution over valid intervals, refined, and mapped to continuous temporal boundaries.

define $\mathcal{C}([0,1]) = (0,1)$. The inverse mapping reconstructs the normalized interval as

$$\mathcal{C}^{-1}(u,v) = [u(1-v),\ u(1-v)+v]\,. \tag{2}$$

We discretize the canonical square into an $N_u \times N_v$ grid $\mathcal{G} = \{(i,j)\}$. The grid centers and corresponding intervals are

$$\bar{u}_i = \frac{i-1}{N_u - 1}, \qquad \bar{v}_j = \frac{j-1}{N_v - 1}, \qquad I_{ij} = \mathcal{C}^{-1}(\bar{u}_i, \bar{v}_j), \tag{3}$$

where $1 \le i \le N_u$ and $1 \le j \le N_v$.

Given a continuous canonical coordinate $c = (u,v)$, we construct one-dimensional Gaussian distributions along the position and duration axes:

$$\begin{aligned} p_i^u &= \mathrm{softmax}_i\left(-\frac{(\bar{u}_i - u)^2}{2\sigma_u^2}\right), \\ p_j^v &= \mathrm{softmax}_j\left(-\frac{(\bar{v}_j - v)^2}{2\sigma_v^2}\right). \end{aligned} \tag{4}$$

The canonical soft target is their outer product:

$$\Phi_{ij}(u,v) = p_i^u p_j^v. \tag{5}$$

3.3 TimePLE: Duration-Aware Interval Codec

Given the canonical mapping $\mathcal{C}$ and the soft density function Φ, TimePLE uses a lightweight interval codec to connect temporal intervals with the hidden space of the VLM. On the input side, temporal anchors are encoded as soft interval densities and inserted into the input token sequence. On the output side, the hidden state of <|TIMESPAN|> is decoded into a distribution over the same canonical grid and refined through a duration-aware coordinate correction.

Distributional Span Decoding. Let h_{span} denote the hidden state of a generated <|TIMESPAN|> token. We project this hidden state into the interval feature space and decode grid logits:

$$f_{\text{span}} = A_{\text{out}}(h_{\text{span}}), \quad \boldsymbol{\ell} = D_{\text{int}}(f_{\text{span}}) \in \mathbb{R}^{N_u \times N_v}, \tag{6}$$

where A_{out} is the output projection layer and D_{int} is the interval decoder.

The logits define a joint span distribution over the canonical grid:

$$p_\theta(i, j \mid h_{\text{span}}) = \text{softmax}_{\mathcal{G}}(\boldsymbol{\ell})_{ij}. \tag{7}$$

We obtain the initial continuous coordinate by taking the expectation over the grid:

$$(\hat{u}_0, \hat{v}_0) = \sum_{i=1}^{N_u} \sum_{j=1}^{N_v} p_\theta(i, j \mid h_{\text{span}})(\bar{u}_i, \bar{v}_j). \tag{8}$$

This expectation decoding preserves uncertainty in the predicted span distribution while producing a continuous interval coordinate.

Duration-Aware Coordinate Refinement. Although grid expectation provides a continuous prediction, its localization precision remains affected by grid discretization. We therefore use the predicted duration coordinate $\hat{v}_0$ to determine the refinement strength:

$$g_0 = \frac{1}{\kappa_{\text{cap}}} \text{clip}\left(\frac{1}{\sqrt{\hat{v}_0 + \epsilon_g}}, 0, \kappa_{\text{cap}}\right). \tag{9}$$

The gate assigns larger correction strength to shorter predicted intervals while keeping the refinement bounded. We use $\epsilon_g = 10^{-4}$ and $\kappa_{\text{cap}} = 10$.

Using the interval feature f_{span}, the refinement module predicts a bounded coordinate offset:

$$\Delta\mathbf{c} = \alpha g_0 \tanh\left(R_{\text{int}}(f_{\text{span}})\right), \tag{10}$$

where $\Delta\mathbf{c} = (\Delta u, \Delta v)$, R_{int} denotes the refinement module, and α controls the maximum correction scale. The refined coordinate and the final normalized interval are obtained by

$$\hat{\mathbf{c}} = \text{clamp}\left((\hat{u}_0, \hat{v}_0) + \Delta\mathbf{c}, 0, 1\right), \quad \hat{I} = \mathcal{C}^{-1}(\hat{\mathbf{c}}). \tag{11}$$

Temporal Anchor Encoding. For the r-th sampled visual unit, we assign a normalized temporal anchor $a_r = [s_r^a, e_r^a]$ according to its temporal coverage in the video. The anchor is converted into a canonical soft density and projected into the VLM embedding space:

$$\begin{aligned} \phi_r^a &= \Phi(\mathcal{C}(a_r)), \\ \mathbf{x}_{p_r} &\leftarrow A_{\text{in}}\left(E_{\text{int}}(\phi_r^a)\right), \end{aligned} \tag{12}$$

where E_{int} denotes the interval encoder, A_{in} is the input projection layer, and p_r denotes the position of the corresponding <|TIMESTAMP|> token. This operation represents the temporal coverage of each visual unit using the same canonical interval geometry employed for output prediction.

3.4 Training Objectives

TimePLE is trained in two stages: interval geometry training and interval-supervised fine-tuning. The first stage learns the canonical interval codec independently of the VLM, while the second stage aligns the generated <|TIMESPAN|> representation with the learned interval space.

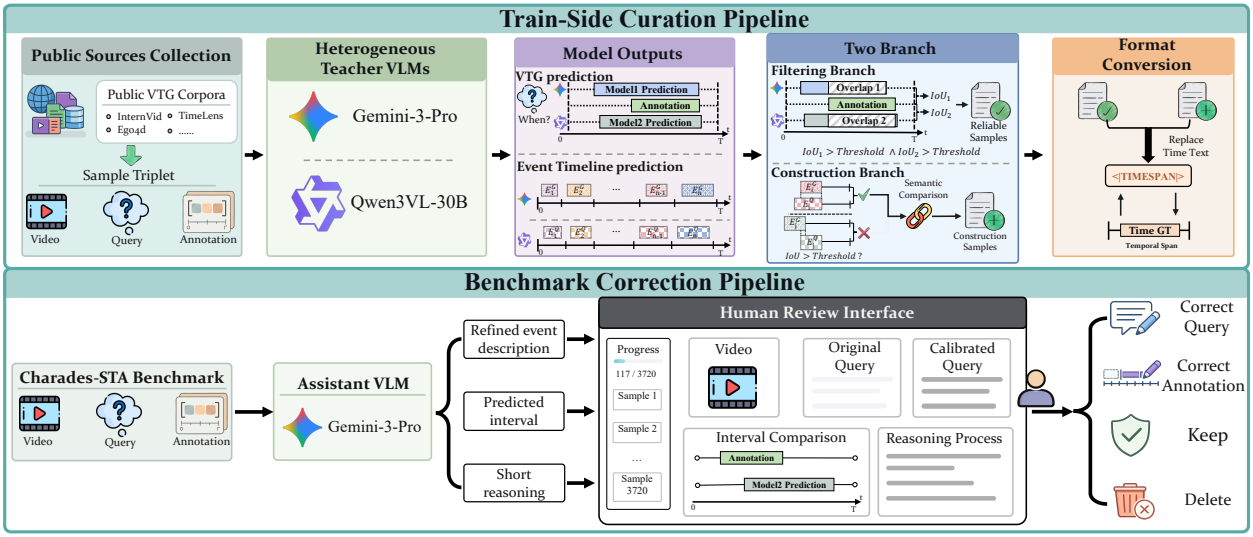


Figure 3. Data curation and benchmark correction. Heterogeneous teacher VLMs filter existing annotations by cross-model agreement and construct new samples through temporal and semantic consensus. Accepted moments are converted into TimePLE supervision, while model outputs serve only as auxiliary evidence for human benchmark correction.

Interval Geometry Training. We construct synthetic interval samples by uniformly sampling canonical coordinates within balanced grid cells. Each coordinate $c_m = (u_m, v_m)$ is converted into a normalized interval $I_m = \mathcal{C}^{-1}(c_m)$, paired with a sampled video duration T_m, and represented by the target density $\phi_m = \Phi(c_m)$. The interval encoder and decoder reconstruct the corresponding span distribution p_m and refined interval $\hat{I}_m$.

For a target interval $I = [s, e]$, predicted interval $\hat{I} = [\hat{s}, \hat{e}]$, target density ϕ, predicted distribution p, and video duration T, we define the shared span objective as

$$\begin{aligned}\mathcal{L}_{\text{span}} = \lambda_{\text{dfl}}\,\text{CE}(\phi, p) + \lambda_{\text{iou}}\left(1 - \text{IoU}(\hat{I}, I)\right) \\ + \lambda_{\text{bnd}}\bar{w}(D)\frac{|\hat{s} - s| + |\hat{e} - e|}{2}, \quad D = T(e - s),\end{aligned} \tag{13}$$

where $\bar{w}(D)$ denotes the batch-normalized absolute-duration weight. The three terms respectively supervise the canonical interval distribution, interval overlap and boundary accuracy.

The interval geometry objective is

$$\mathcal{L}_{\text{geo}} = \frac{1}{M}\sum_{m=1}^{M}\mathcal{L}_{\text{span}}\left(p_m, \hat{I}_m; \phi_m, I_m, T_m\right). \tag{14}$$

Supervised Fine-Tuning. In the second stage, each training sample contains a video V, query q, target response $x_{1:N}$, target interval $\widetilde{I}^*$, and video duration T. The target response contains one <|TIMESPAN|> token aligned with the continuous temporal label. The standard autoregressive language modeling loss is

$$\mathcal{L}_{\text{LM}} = -\sum_{t=1}^{N}\log p_\theta\left(x_t \mid x_{<t}, V, q\right). \tag{15}$$

Let h_{span} denote the hidden state at the <|TIMESPAN|> position. The interval codec decodes it into a span distribution p_θ and a refined interval $\hat{I}$. The target segment $\widetilde{I}^* = [t_s^*, t_e^*]$ is normalized by T to obtain $I^* = [s^*, e^*]$, and its distributional target is $\phi^* = \Phi(\mathcal{C}(I^*))$. We apply the shared span objective in Eq. (13) to obtain $\mathcal{L}_{\text{span}}^{\text{SFT}}$. The final supervised fine-tuning objective is

$$\mathcal{L}_{\text{SFT}} = \mathcal{L}_{\text{LM}} + \lambda_{\text{span}}\mathcal{L}_{\text{span}}^{\text{SFT}}. \tag{16}$$

Table 1. Main results on VTG benchmarks. We report overall and duration-stratified mIoU. Rep. denotes textual timestamps (T), symbolic time tokens (S), continuous boundary decoding (B), and interval-native decoding (I). Avg. is the mean overall mIoU across the four benchmarks.

Model	Rep.	Charades-STA			ActivityNet-Captions				QVHighlights			Charades-TimePLE				Avg.
		mIoU	mIoU_S	mIoU_M	**mIoU**	mIoU_S	mIoU_M	mIoU_L	**mIoU**	mIoU_M	mIoU_L	**mIoU**	mIoU_S	mIoU_M	mIoU_L	
Proprietary Models																
GPT-4o [23]	T	22.5	19.8	31.9	19.2	19.5	18.7	19.3	14.9	14.5	15.2	28.7	24.9	39.2	**74.7**	21.3
Gemini-3.1-Pro [5]	T	45.1	43.4	51.1	46.3	**46.1**	47.6	45.5	59.7	58.6	60.9	52.8	49.8	60.9	72.7	51.0
Open-Source Models																
VTG-LLM [10]	S	34.2	33.2	37.7	16.7	19.3	22.8	10.3	10.0	9.4	10.7	33.9	32.2	39.0	34.7	23.7
Grounded-VideoLLM [28]	S	42.2	39.9	50.2	40.9	27.4	40.7	50.3	46.9	50.6	42.7	44.2	41.1	53.4	58.2	43.6
NumPro [35]	T	24.7	22.9	30.9	16.4	13.8	21.2	14.7	24.2	16.3	33.2	25.9	22.4	35.8	66.3	22.8
DisTime [37]	B	49.4	46.1	61.0	46.1	27.1	44.4	**62.1**	50.7	47.2	54.7	53.4	48.9	66.9	73.4	49.9
Time-R1-7B [32]	T	28.5	28.6	27.9	16.3	20.1	20.4	10.6	17.8	15.4	20.7	28.4	27.8	30.3	28.5	22.8
TimeLens-8B [41]	T	42.6	40.7	48.9	44.0	42.2	46.7	43.3	61.7	62.2	59.3	51.6	48.9	59.0	71.7	50.0
InternVL3.5-8B [30]	T	21.9	18.6	33.3	27.9	25.2	27.2	30.3	47.2	46.4	48.3	28.1	23.9	39.7	66.4	31.3
Qwen3VL-8B [2]	T	47.9	46.4	53.1	42.3	39.7	43.8	42.8	46.0	44.5	47.8	55.8	53.9	61.2	69.8	48.0
TimePLE-8B	**I**	**57.2**	**55.5**	**62.9**	**49.6**	37.2	**50.1**	57.6	**65.3**	**65.9**	**64.2**	**63.4**	**60.8**	**71.1**	70.9	**58.9**

3.5 Data Curation

We curate training data from public VTG sources [9, 33, 41] using Qwen3-VL-30B [2] and Gemini-3-Pro [5] as heterogeneous teacher VLMs. Existing annotations are retained only when both teachers agree with the labeled interval, while new grounded samples require temporal overlap and semantic consistency between teacher predictions. Each accepted moment is converted into one <|TIMESPAN|> token paired with a continuous interval label. Separately, model predictions and calibrated queries are presented as auxiliary evidence in a human review interface for benchmark correction. The pipeline produces 90K training samples and corrects 3K-scale noisy annotations. Additional details are provided in the Appendix.

4 Experiment

4.1 Experimental Setup

We evaluate on Charades-STA, QVHighlights, ActivityNet-Captions, and the Charades-TimePLE benchmark. TimePLE is built on Qwen3-VL-8B and uses 2-FPS video sampling with at most 200 frames and 64 visual tokens per frame. We report overall mIoU and duration-stratified mIoU over Short $(0, 10]$s, Medium $(10, 30]$s, and Long $(30, \infty)$s moments. More setup details are provided in the Appendix.

4.2 Main Results on VTG Benchmarks

Table 1 compares four temporal-output paradigms for VLM-based VTG. TimePLE achieves **the best average performance of 58.9 mIoU** and improves Qwen3-VL-8B from 48.0 to 58.9. The gains are strongest on short and medium moments, where small temporal errors cause larger degradation in interval overlap, while TimePLE remains competitive on long moments. These results show that duration-conditioned interval prediction improves fine-grained localization without sacrificing coarse temporal grounding.

4.3 Scaling Across VLM Backbones

We further apply the same interval-native formulation to LLaVA-OV-0.5B [16] and Qwen2.5-VL-3B [3]. For each backbone, TimePLE is compared with a matched Timestamp-SFT baseline using the same training data and video input configuration, with only the temporal prediction interface changed. Table 2 shows higher mIoU for TimePLE in all twelve backbone–benchmark comparisons. The average improvements are 20.6 points on LLaVA-OV-0.5B, 0.9 points on Qwen2.5-VL-3B, and 4.1 points on Qwen3-VL-8B. The consistent gains across model families and scales support the benefit of replacing endpoint prediction with interval-distribution prediction. The smaller improvements on the Qwen-VL backbones may result from their larger-scale pretraining and stronger language priors, which make Timestamp-SFT a more competitive baseline.

Table 2. Scaling across VLM backbones. TimePLE is compared with Timestamp-SFT baselines under the same training data and video input configuration. All entries are mIoU. C-STA, A-Net, QVH, and C-TPLE denote Charades-STA, ActivityNet-Captions, QVHighlights, and Charades-TimePLE, respectively. Parenthesized numbers report mIoU gains over the matched Timestamp-SFT baseline. The Avg. gain is averaged across the four benchmarks.

Backbone	Method	C-STA	A-Net	QVH	C-TPLE	Avg.
LLaVA-OV-0.5B	Timestamp-SFT	23.8	10.8	6.5	24.7	16.5
	TimePLE	**40.2** (↑ 16.4)	**33.2** (↑ 22.4)	**32.3** (↑ 25.8)	**42.3** (↑ 17.6)	**37.0** (↑ 20.6)
Qwen2.5-VL-3B	Timestamp-SFT	51.2	41.6	59.3	57.8	52.5
	TimePLE	**52.3** (↑ 1.1)	**43.3** (↑ 1.7)	**59.4** (↑ 0.1)	**58.5** (↑ 0.7)	**53.4** (↑ 0.9)
Qwen3-VL-8B	Timestamp-SFT	51.6	47.3	63.6	56.7	54.8
	TimePLE	**57.2** (↑ 5.6)	**49.6** (↑ 2.3)	**65.3** (↑ 1.7)	**63.4** (↑ 6.7)	**58.9** (↑ 4.1)

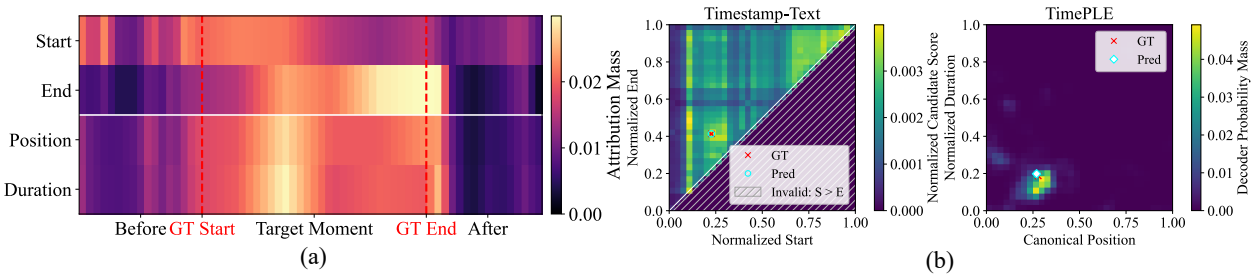


Figure 4. Representation-level analysis. (a) Gradient × Input attribution for Timestamp-Text start–end and TimePLE position–duration targets. Dashed lines denote ground-truth boundaries. (b) Timestamp-Text likelihood reconstructed over valid start–end candidates versus TimePLE's native distribution over the canonical interval square. Crosses and diamonds denote ground truth and prediction.

4.4 Representation-Level Analysis

Beyond benchmark performance, we examine how endpoint and interval-distribution prediction differ in temporal representation, considering both supporting visual evidence and probability organization over candidate intervals. More implementation details are provided in the Appendix.

4.4.1 Temporal evidence attribution

We first examine how the two temporal interfaces draw evidence from the video using Gradient × Input attribution on visual embeddings. Timestamp-Text uses the teacher-forced likelihoods of the ground-truth start and end tokens, while TimePLE uses the soft marginal likelihoods of the ground-truth canonical position and duration. As shown in Fig. 4(a), endpoint prediction relies on temporally separated cues associated with the two boundaries. In contrast, position and duration receive coherent support throughout the target moment, suggesting that TimePLE forms its prediction from the event segment as a whole rather than composing it from two endpoint decisions.

4.4.2 Interval likelihood landscape

We further visualize how temporal uncertainty is organized in the output space. For Timestamp-Text, we enumerate valid boundary candidates and normalize their teacher-forced numeric-token likelihoods to reconstruct an interval landscape. TimePLE directly provides a joint probability distribution over the canonical interval square. In Fig. 4(b), the reconstructed timestamp landscape is diffuse and fragmented over the triangular valid region, whereas TimePLE concentrates probability around the target in a fully valid interval space. This comparison illustrates that interval-native decoding provides a more structured and localized distribution over candidate moments.

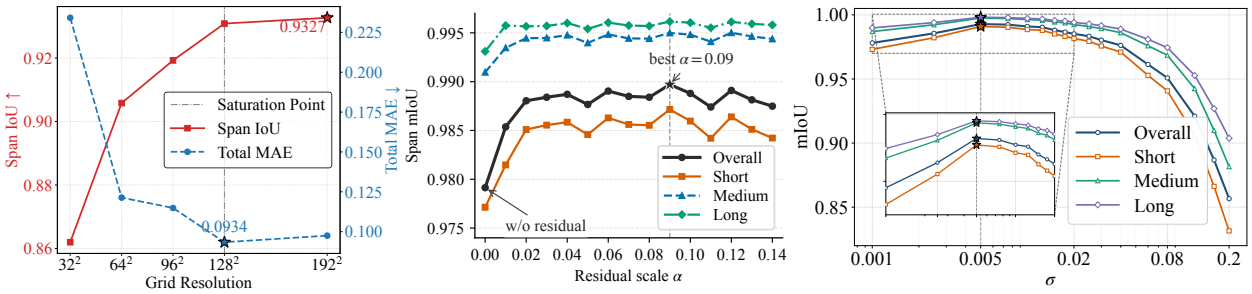
4.5 Ablation Studies

4.5.1 Interval-Space Reconstruction Analysis

We analyze whether the canonical interval space can support accurate continuous interval reconstruction under different discretization and smoothing settings.

Table 3. Ablation of progressive training stages. All entries are mIoU. C-STA, A-Net, QVH, and C-TPLE denote Charades-STA, ActivityNet-Captions, QVHighlights, and Charades-TimePLE, respectively. Parenthesized numbers report mIoU changes relative to the preceding training setting. The Avg. gain is averaged across the four benchmarks.

Training setting	C-STA	A-Net	QVH	C-TPLE	Avg.
Baseline	47.9	42.3	46.0	55.8	48.0
+ SFT	51.6 (↑ 3.7)	47.3 (↑ 5.0)	63.6 (↑ 17.6)	56.7 (↑ 0.9)	54.8 (↑ 6.8)
TimePLE (SFT)	**57.2** (↑ 5.6)	**49.6** (↑ 2.3)	**65.3** (↑ 1.7)	**63.4** (↑ 6.7)	**58.9** (↑ 4.1)
TimePLE (GRPO)	57.2 (0.0)	49.6 (0.0)	65.1 (↓ 0.2)	63.3 (↓ 0.1)	58.8 (↓ 0.1)



(a) Effect of Grid Resolution (b) Effect of Residual Scale (c) Effect of Gaussian Bandwidth

Figure 5. Interval-space reconstruction analysis. From left to right, we study the canonical grid resolution, residual refinement scale α, and Gaussian bandwidth σ. Reconstruction largely saturates at 128^2, benefits from a small residual correction with $\alpha = 0.09$, and performs best under moderately local supervision with $\sigma = 0.005$.

Grid resolution. As shown in Fig. 5 (a), reconstruction quality improves substantially as the grid increases from 32^2 to 128^2. Increasing the resolution to 192^2 yields only a marginal span-IoU gain, while the total MAE slightly increases and the number of grid support points grows by $2.25\times$. We use 128^2 as the default resolution, which provides a favorable balance between reconstruction accuracy and decoding complexity.

Residual refinement. Setting $\alpha = 0$ corresponds to direct grid-expectation decoding without coordinate refinement. As shown in Fig. 5 (b), nonzero residual scales consistently improve reconstruction, with the largest gain observed for short intervals, which are more sensitive to small coordinate errors. Performance remains stable across a broad range of scales and we set $\alpha = 0.09$ as default.

Gaussian bandwidth. We further analyze the Gaussian bandwidth σ used for soft interval-grid supervision. Figure 5 (c) shows that reconstruction improves as σ increases from an overly sharp target to a moderate bandwidth. Larger bandwidths consistently degrade reconstruction, especially for short intervals, because excessive smoothing reduces the distinction between neighboring spans. We therefore set $\sigma = 0.005$ as default.

Findings

Accurate interval reconstruction requires both sufficient coverage and localized distributional supervision. Residual refinement complements grid-based decoding by correcting the remaining continuous localization error, particularly for short intervals.

4.5.2 Effect of Interval-Supervised Training

Table 3 separates ordinary task adaptation from interval-space supervision. Timestamp-SFT raises the average mIoU from 48.0 to 54.8, confirming the benefit of supervised adaptation to the VTG response format. TimePLE-SFT further improves the average to 58.9 and yields consistent gains on all four benchmarks, isolating the effect of interval-level supervision beyond ordinary SFT. In contrast, GRPO produces no additional improvement, with the average mIoU slightly changing from 58.9 to 58.8. A more detailed analysis of post-SFT optimization is provided in the Appendix.

Findings

The main gain of TimePLE comes from directly aligning the span-token representation with the canonical interval space. Once this alignment is learned through interval-supervised SFT, outcome-level GRPO provides little additional benefit.

5 Conclusion

We present TimePLE, an interval-native framework for VLM-based video temporal grounding. Rather than generating timestamps or boundary pairs, TimePLE decodes a generated span token into a distribution over duration-conditioned event intervals. This representation preserves the autoregressive VLM interface while making duration, placement, and interval-level supervision explicit in the prediction space. Experiments across four VTG benchmarks show consistent gains, especially on short and medium-duration events.

References

[1] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017.

[2] Shuai Bai, Yuxuan Cai, Ruizhe Chen, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[3] Shuai Bai, Keqin Chen, Xuejing Liu, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[4] Yukang Chen, Fuzhao Xue, Dacheng Li, et al. Longvila: Scaling long-context visual language models for long videos. *arXiv preprint arXiv:2408.10188*, 2024.

[5] Google DeepMind. Gemini 3.1 pro model card, 2026. URL https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-1-Pro-Model-Card.pdf.

[6] Andong Deng, Zhongpai Gao, et al. Seq2time: Sequential knowledge transfer for video llm temporal grounding. *arXiv preprint arXiv:2411.16932*, 2024.

[7] Lu Dong, Haiyu Zhang, Han Lin, Ziang Yan, Xiangyu Zeng, Hongjie Zhang, Yifei Huang, Yi Wang, Zhen-Hua Ling, Limin Wang, and Yali Wang. Videotg-r1: Boosting video temporal grounding via curriculum reinforcement learning on reflected boundary annotations. *arXiv preprint arXiv:2510.23397*, 2025.

[8] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 5267–5275, 2017.

[9] Kristen Grauman et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[10] Yongxin Guo, Jingyu Liu, Mingda Li, et al. Vtg-llm: Integrating timestamp knowledge into video llms for enhanced video temporal grounding. *arXiv preprint arXiv:2405.13382*, 2024.

[11] Bin Huang, Xin Wang, Hong Chen, et al. Vtimellm: Empower llm to grasp video moments. *arXiv preprint arXiv:2311.18445*, 2024.

[12] De-An Huang, Shijia Liao, Subhashree Radhakrishnan, et al. Lita: Language instructed temporal-localization assistant. *arXiv preprint arXiv:2403.19046*, 2024.

[13] Shengji Jin, Yuanhao Zou, Victor Zhu, Zhengping Ji, and Chen Chen. How should video llms output time? an analysis of efficient temporal grounding paradigms. *arXiv preprint arXiv:2604.08966*, 2026.

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

[15] Jie Lei, Tamara L Berg, and Mohit Bansal. Detecting moments and highlights in videos via natural language queries. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[16] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, et al. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[17] Jiaze Li, Hao Yin, Haoran Xu, Boshen Xu, Wenhui Tan, Zewen He, Jianzhong Ju, Zhenbo Luo, and Jian Luan. Video-opd: Efficient post-training of multimodal large language models for temporal video grounding via on-policy distillation. *arXiv preprint arXiv:2602.02994*, 2026.

[18] Yunheng Li, Jing Cheng, Shaoyong Jia, Hangyi Kuang, Shaohui Jiao, Qibin Hou, and Ming-Ming Cheng. Tempsamp-r1: Effective temporal sampling with reinforcement fine-tuning for video llms. *arXiv preprint arXiv:2509.18056*, 2025.

[19] Zeqian Li, Shangzhe Di, Zhonghua Zhai, et al. Universal video temporal grounding with generative multi-modal large language models. *arXiv preprint arXiv:2506.18883*, 2025.

[20] Bin Lin, Yang Ye, Bin Zhu, Jiaxi Cui, Munan Ning, Peng Jin, and Li Yuan. Video-llava: Learning united visual representation by alignment before projection. *arXiv preprint arXiv:2311.10122*, 2023.

[21] Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Shahbaz Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. *arXiv preprint arXiv:2306.05424*, 2024.

[22] X Mo et al. Tar-tvg: Enhancing vlms with timestamp anchor-constrained reasoning for temporal video grounding. *arXiv preprint arXiv:2508.07683*, 2025.

[23] OpenAI. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[24] Long Qian et al. Momentor: Advancing video large language model with fine-grained temporal reasoning. *arXiv preprint arXiv:2402.11435*, 2024.

[25] Michaela Regneri, Marcus Rohrbach, Dominikus Wetzel, Stefan Thater, Bernt Schiele, and Manfred Pinkal. Grounding action descriptions in videos. *Transactions of the Association for Computational Linguistics (TACL)*, 1:25–36, 2013.

[26] Shuhuai Ren et al. Timechat: A time-sensitive multimodal large language model for long video understanding. *arXiv preprint arXiv:2312.02051*, 2024.

[27] Mattia Soldan, Alejandro Pardo, Juan León Alcázar, Fabian Caba Heilbron, Chen Zhao, Silvio Giancola, and Bernard Ghanem. Mad: A scalable dataset for language grounding in videos from movie audio descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[28] Haoran Wang et al. Grounded-videollm: Sharpening fine-grained temporal grounding in video large language models. *arXiv preprint arXiv:2410.03290*, 2024.

[29] Peng Wang, Shuai Bai, Sinan Tan, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[30] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[31] Xizi Wang et al. Timerefine: Temporal grounding with time refining video llm. *arXiv preprint arXiv:2412.09601*, 2024.

[32] Ye Wang, Ziheng Wang, Boshen Xu, et al. Time-r1: Post-training large vision language model for temporal video grounding. *arXiv preprint arXiv:2503.13377*, 2025.

[33] Yi Wang, Yinan He, Yizhuo Li, Kunchang Li, Jiashuo Yu, Xin Ma, Xinhao Li, Guo Chen, Xinyuan Chen, Yaohui Wang, Conghui He, Ping Luo, Ziwei Liu, Yali Wang, Limin Wang, and Yu Qiao. Internvid: A large-scale video-text dataset for multimodal understanding and generation. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[34] Yi Wang, Xinhao Li, Ziang Yan, et al. Internvideo2.5: Empowering video mllms with long and rich context modeling. *arXiv preprint arXiv:2501.12386*, 2025.

[35] Yongliang Wu et al. Number it: Temporal grounding videos like flipping manga. *arXiv preprint arXiv:2411.10332*, 2024.

[36] Feng Yue, Zhaoxing Zhang, Junming Jiao, Zhengyu Liang, Shiwen Cao, Feifei Zhang, and Rong Shen. Tempo-r0: A video-mllm for temporal video grounding through efficient temporal sensing reinforcement learning. *arXiv preprint arXiv:2507.04702*, 2025.

[37] Y Zeng et al. Distime: Distribution-based time representation for video large language models. *arXiv preprint arXiv:2505.24329*, 2025.

[38] Boqiang Zhang, Kehan Li, Zesen Cheng, Zhiqiang Hu, Yuqian Yuan, Guanzheng Chen, Sicong Leng, Yuming Jiang, Hang Zhang, Xin Li, et al. Videollama 3: Frontier multimodal foundation models for image and video understanding. *arXiv preprint arXiv:2501.13106*, 2025.

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

[40] Jinglei Zhang, Yuanfan Guo, Rolandos Alexandros Potamias, et al. Vtimecot: Thinking by drawing for video temporal grounding and reasoning. *arXiv preprint arXiv:2510.14672*, 2025.

[41] Jun Zhang, Teng Wang, Yuying Ge, Yixiao Ge, Xinhao Li, Ying Shan, and Limin Wang. Timelens: Rethinking video temporal grounding with multimodal llms. *arXiv preprint arXiv:2512.14698*, 2025.

[42] Peiyuan Zhang et al. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*, 2024.

[43] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data, 2024.

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

Appendix

A Implementation Details

We provide the implementation details of TimePLE but are not expanded in the main paper.

A.1 VLM Backbone and Temporal Token Interface

TimePLE is implemented on top of Qwen3-VL-8B-Instruct. To support interval-aware temporal grounding, we add two temporal special tokens, <|TIMESTAMP|> and <|TIMESPAN|>. The former is used for input-side temporal anchoring, while the latter provides the output-side latent interface for continuous interval decoding. The newly introduced temporal-token embeddings are initialized from the mean and covariance statistics of the existing token embeddings. This provides stable initial representations for the temporal tokens without perturbing the original vocabulary embedding space.

A.2 Interval Codec Configuration

The interval encoder, interval decoder, and CIS span projector are implemented as lightweight MLP modules. The encoder maps the input interval-grid representation into the 4096-dimensional hidden space of the base VLM, the decoder maps the 4096-dimensional interval feature back to the canonical interval-grid prediction space, and the CIS span projector transforms the hidden state associated with <|TIMESPAN|> before interval decoding.

B Additional Details of the Canonical Interval Space

B.1 Canonical Position-Duration Parameterization

The canonical position-duration parameterization follows directly from the feasible geometry of temporal intervals. For a fixed duration, the start position is only valid within the duration-conditioned range $[0, 1 - d]$ rather than the full unit interval. Therefore, normalizing the start position by this feasible range yields a duration-conditioned placement coordinate, while keeping the duration itself as an explicit coordinate. This design represents a temporal moment by where an interval of a given length can be placed, rather than by two raw boundary values whose feasibility must be handled implicitly.

B.2 Theoretical Analysis of the Canonical Interval Square

Building on the parameterization above, the key theoretical property of the canonical interval square is the alignment between the decoder output support and the feasible interval space. Since each point in the canonical square corresponds to a valid temporal interval, a dense grid decoder can be interpreted directly as a distribution over valid spans. This avoids assigning probability mass to invalid boundary configurations and removes the need for boundary ordering, masking, or post-hoc interval repair. This prediction geometry is more structured than directly modeling raw start and end boundaries jointly. In a boundary-coordinate space, interval validity, duration, and placement are entangled in two boundary values, and the rectangular output space of a neural decoder does not naturally coincide with the feasible set of ordered intervals. In contrast, the canonical interval square explicitly separates temporal extent from feasible placement, allowing the model to express uncertainty over "how long the moment lasts" and "where a moment of that length is located" on two semantically meaningful axes. This provides a cleaner and more interval-native prediction domain for duration-aware temporal grounding.

C Additional Details of Representation-Level Analysis

We provide implementation details and additional qualitative examples for the representation-level analysis. The analysis examines the two temporal interfaces from complementary perspectives: the visual evidence supporting their temporal targets and the organization of probability mass over candidate intervals.

C.1 Temporal Evidence Attribution

We use Gradient $\times$ Input attribution to measure how each temporal portion of the video supports the target temporal prediction. We register a hook after the visual projection layer and extract the visual-token embeddings $\mathbf{E} \in \mathbb{R}^{N_{\text{visual}} \times d}$. Model parameters are frozen, while gradients are retained for $\mathbf{E}$.

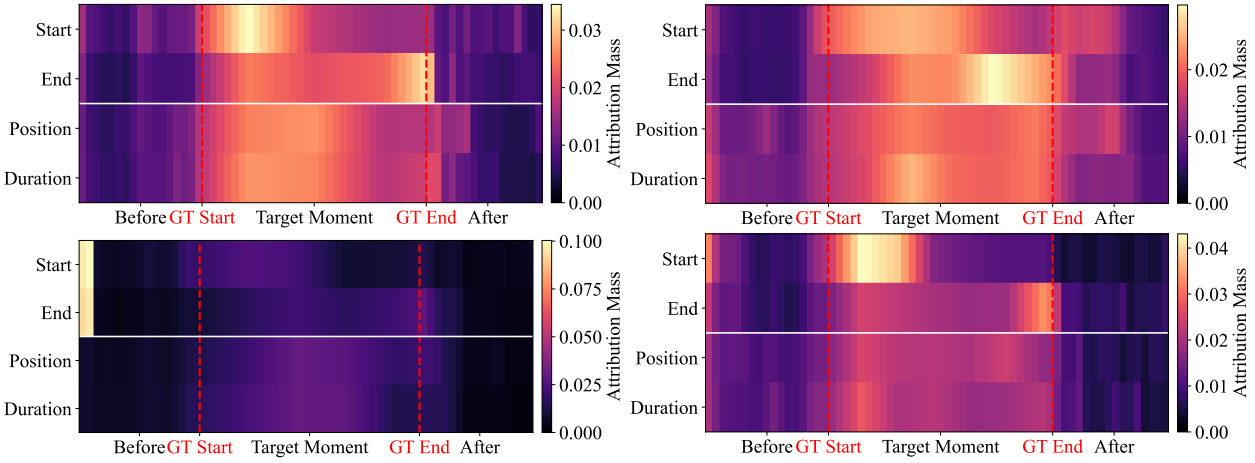


Figure 6. Additional temporal evidence attribution cases. Timestamp-Text attribution is computed from the teacher-forced likelihoods of the ground-truth start and end timestamp tokens, while TimePLE attribution is computed from the soft marginal likelihoods of the ground-truth canonical position and duration. Dashed lines denote the ground-truth temporal boundaries.

For Timestamp-Text, the model is evaluated with teacher forcing using the ground-truth response. Let $\mathcal{I}_s$ and $\mathcal{I}_e$ denote the numeric-token positions of the ground-truth start and end timestamps. Their attribution targets are the mean token log-likelihoods

$$\begin{aligned} y_s &= \frac{1}{|\mathcal{I}_s|} \sum_{k \in \mathcal{I}_s} \log p_\theta(z_k \mid z_{<k}, V, Q), \\ y_e &= \frac{1}{|\mathcal{I}_e|} \sum_{k \in \mathcal{I}_e} \log p_\theta(z_k \mid z_{<k}, V, Q). \end{aligned} \tag{17}$$

Averaging avoids attribution-scale differences caused by different numbers of numerical tokens.

For TimePLE, the ground-truth interval is mapped to its canonical position and duration. From the predicted joint distribution $P(u, v)$, we obtain the corresponding marginal distributions and evaluate them using Gaussian soft targets q_u and q_v with $\sigma = 0.015$:

$$\begin{aligned} y_u &= \sum_i q_u(i) \log \sum_j P(u_i, v_j), \\ y_v &= \sum_j q_v(j) \log \sum_i P(u_i, v_j). \end{aligned} \tag{18}$$

For each scalar target y, the attribution of visual token n is

$$a_n = \sum_{c=1}^{d} \left| E_{n,c} \frac{\partial y}{\partial E_{n,c}} \right|. \tag{19}$$

The attribution values of spatial tokens belonging to the same temporal position are averaged, producing a one-dimensional attribution sequence over the video timeline. Larger values indicate that the corresponding visual content is more sensitive to the specified temporal target. Additional examples are shown in Fig. 6.

C.2 Interval Likelihood Landscape

Timestamp-Text does not natively produce a probability distribution over complete temporal intervals. We uniformly discretize the video duration into 32 time points and enumerate the 528 valid start–end candidates. Each candidate is

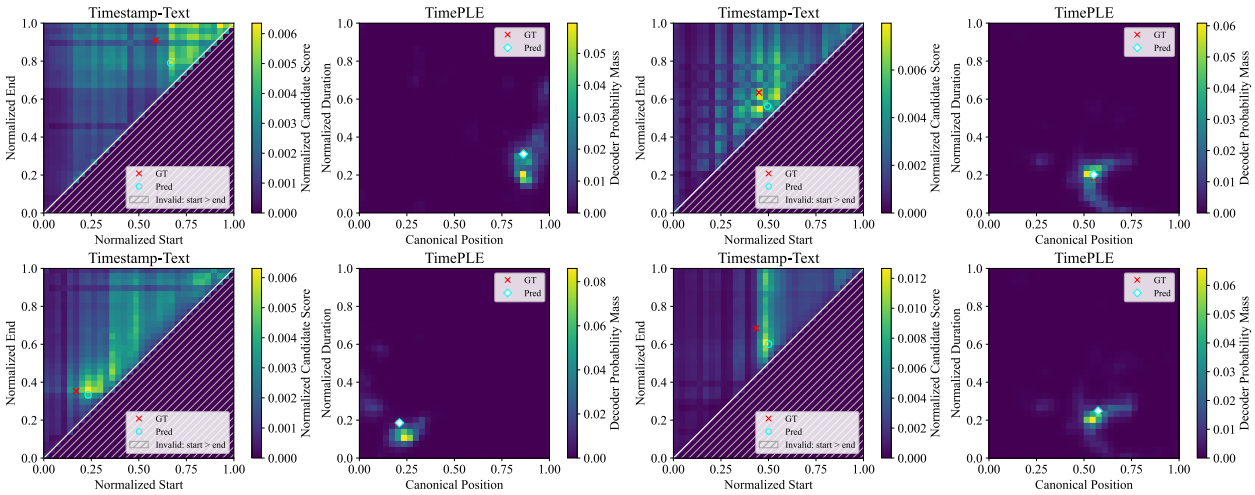


Figure 7. Additional interval likelihood landscapes. Timestamp-Text landscapes are reconstructed by enumerating 528 valid start–end candidates and normalizing their teacher-forced numerical-token likelihoods. TimePLE directly outputs a joint probability distribution over the canonical interval square. Crosses and diamonds denote the ground truth and model prediction, respectively.

formatted as a textual timestamp response and evaluated through teacher-forced numerical-token likelihood:

$$
\begin{aligned}
S(i,j) &= \frac{1}{N_{ij}} \sum_{k \in \mathcal{I}_{ij}} \log p_\theta(z_k \mid z_{<k}, V, Q), \\
P_{\text{text}}(i,j) &= \frac{\exp S(i,j)}{\sum_{a \le b} \exp S(a,b)}, \qquad i \le j.
\end{aligned}
\tag{20}
$$

The region $i > j$ contains invalid intervals and is masked in the visualization. Averaging over numerical tokens prevents candidate scores from being affected by timestamp-token length.

TimePLE instead directly outputs joint logits $L \in \mathbb{R}^{128\times128}$ over the canonical position–duration square. Its native interval distribution is

$$
P_{\text{TimePLE}}(u,v) = \frac{\exp L(u,v)}{\sum_{a,b} \exp L(a,b)}. \tag{21}
$$

Every support point corresponds to a valid temporal interval. For visualization, the distribution is reduced to 32×32 by summing each non-overlapping 4×4 block, which preserves the total probability mass. Additional cases are provided in Fig. 7.

D Data Curation and Benchmark Correction Details

D.1 Data Pipeline Overview

Our data pipeline consists of two complementary branches: training-side data curation and evaluation-side benchmark correction. The training-side branch uses strong open-source and closed-source video VLMs to filter existing temporal annotations and construct additional grounded samples, which are then converted into the TimePLE supervision format with explicit <|TIMESPAN|> tokens and continuous interval labels. The evaluation-side branch uses model-assisted human review to correct noisy benchmark annotations, producing a human-verified benchmark for assessing temporally precise grounding. These two branches serve different purposes: the former improves supervision quality for training, while the latter improves evaluation fidelity.

D.2 Training Data Curation and Construction

We construct the TimePLE training set through two complementary operations: filtering existing annotated samples and constructing additional grounded samples from model-proposed event candidates.

You are an expert in video temporal grounding. Based on the video content and the Query, first provide an objective understanding of the video content, then give the accurate temporal segment for the Query in the video.
Query: {query}
Requirements:
1) event_timeline is the model's objective understanding of the entire video content and must be based only on video evidence.
2) event_timeline is a list. Each event must contain description, start_time, and end_time.
3) event_timeline should cover the major events in the video in chronological order, with temporal boundaries as accurate as possible.
4) query_prediction is the final temporal localization prediction for the original Query and should best match the Query.
5) If the Query does not match the video content, set query_prediction.start_time and query_prediction.end_time to null.
6) All timestamps must use absolute seconds on the video timeline.
Output format:

```
{
"event_timeline": [
   {
    "description": "...",
    "start_time": xx.xx,
    "end_time": xx.xx
   }
 ],
 "query_prediction": {
  "start_time": xx.xx,
  "end_time": xx.xx
 }
}
```

Figure 8. The video VLMs take the video and query as input, produce an event-level timeline and a query-specific temporal prediction, and the resulting fields are used for existing-sample filtering and new grounded-sample construction.

You are an expert in video temporal grounding and benchmark annotation correction.
Task:
Based on video evidence, determine whether the given query accurately matches the video content, provide precise temporal localization, and revise the query if necessary.
Output JSON schema:

```
{
 "original_query": "...",
 "refined_query": "...",
 "refined_segment": {
   "start_time": xx.xx,
   "end_time": xx.xx
 },
 "reason": "..."
}
```

Rules:
1) All judgments must be based on video evidence; do not judge based only on the query text.
2) If the original query is semantically accurate and the temporal boundaries are reasonable, set refined_query equal to original_query.
3) If the original query is semantically inaccurate, too broad, too narrow, ambiguous, or missing important information, rewrite it as a more accurate expression in refined_query.
4) If the query does not match the video content, set refined_segment.start_time and refined_segment.end_time to null.
5) refined_segment must use absolute timestamps on the video timeline, in seconds.

Original query: {query}

Figure 9. Structured prompting protocol for model-assisted benchmark correction. Gemini-3-Pro takes the video, original query, and original temporal annotation as input, and outputs a calibrated query, refined temporal segment, and concise reasoning. These outputs are displayed in the review interface as auxiliary evidence for human correction.

Data Sources, Video VLMs, and Prompting Protocol We build the candidate pool from public video grounding and video-text sources, including InternVid, TimeLens, Ego4D, the Charades-STA training split, and the ActivityNet-Captions training split. We use Qwen3-VL-30B and Gemini-3-Pro as the open-source and closed-source video VLMs for training data curation. These models are selected according to our temporal grounding evaluation, where they show the strongest empirical grounding quality among the evaluated candidates. Their heterogeneous model families also provide complementary candidate intervals for agreement-based filtering and construction. For training data curation, the

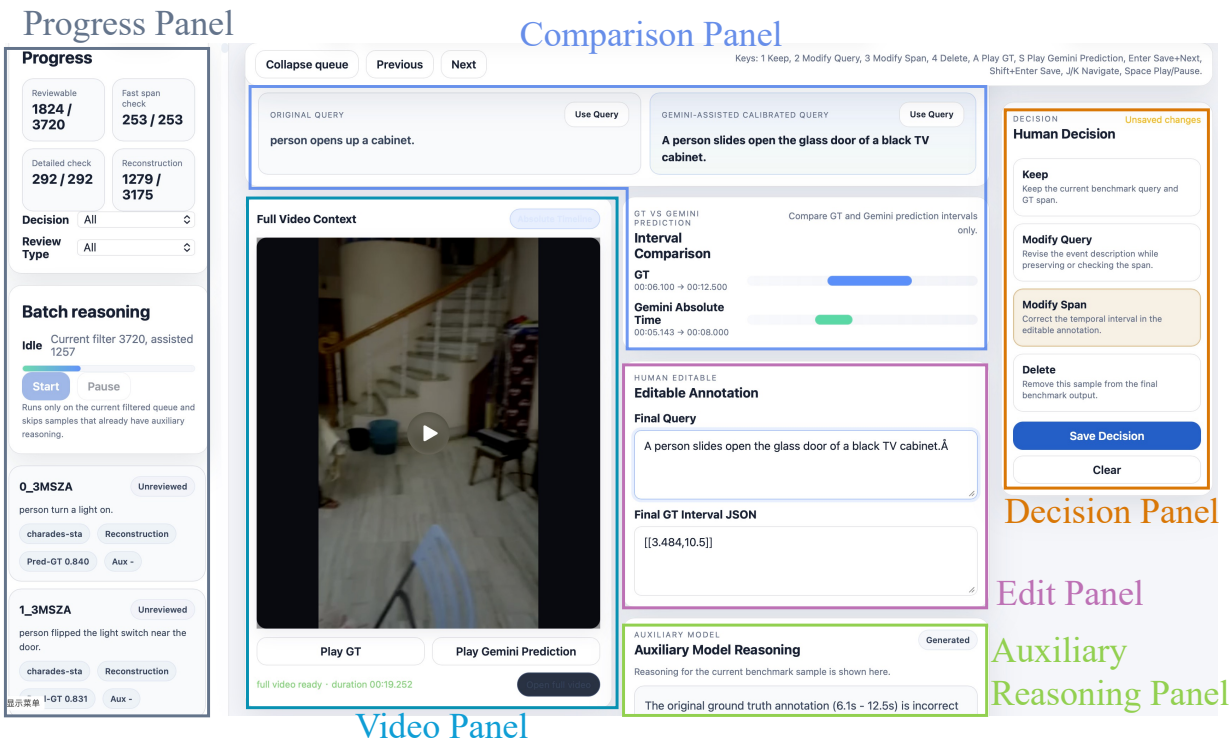


Figure 10. Web-based interface for benchmark correction. The interface contains a *Progress Panel* for tracking the review queue, a *Comparison Panel* for comparing the original query, Gemini-assisted calibrated query, and temporal intervals, a *Video Panel* for inspecting the full video and localized clips, an *Edit Panel* for modifying the final query and temporal interval, a *Decision Panel* for recording the human decision, and an *Auxiliary Reasoning Panel* for displaying model-generated reasoning.

video VLMs are prompted to produce structured outputs, including the predicted temporal interval, the corresponding event description, and the query-event correspondence. The complete prompt is illustrated in Fig. 8.

Filtering Existing Samples. Given an existing annotated sample (v, q, y), where v is the video, q is the query, and y is the original temporal annotation, we ask the video VLMs to predict temporal intervals for the same video-query pair. A sample is retained when both model predictions reach the filtering IoU threshold with the original annotation. This step uses video VLM outputs as reliability evidence for existing annotations, rather than as replacements for the original temporal labels.

Constructing New Grounded Samples. In addition to filtering existing annotations, we construct new grounded samples from event-level model predictions. A new sample is accepted when the two video VLMs identify a matched event, where matching requires both sufficient temporal overlap between the predicted intervals and semantic consistency between the event or query descriptions. This operation expands the training set with additional grounded moments beyond the originally annotated samples.

Conversion to TimePLE Supervision. The retained existing samples and newly constructed samples are converted into the same TimePLE supervision format. Each grounded moment is represented in the textual response by a <|TIMESPAN|> token, and its continuous temporal interval is stored as the corresponding numerical label. Thus, each <|TIMESPAN|> token is aligned with exactly one temporal interval.

D.3 Benchmark Correction with Human Review

The evaluation-side branch corrects noisy benchmark annotations through a model-assisted human review process, where Gemini-3-Pro provides auxiliary temporal evidence and human annotators determine the final temporal bound-

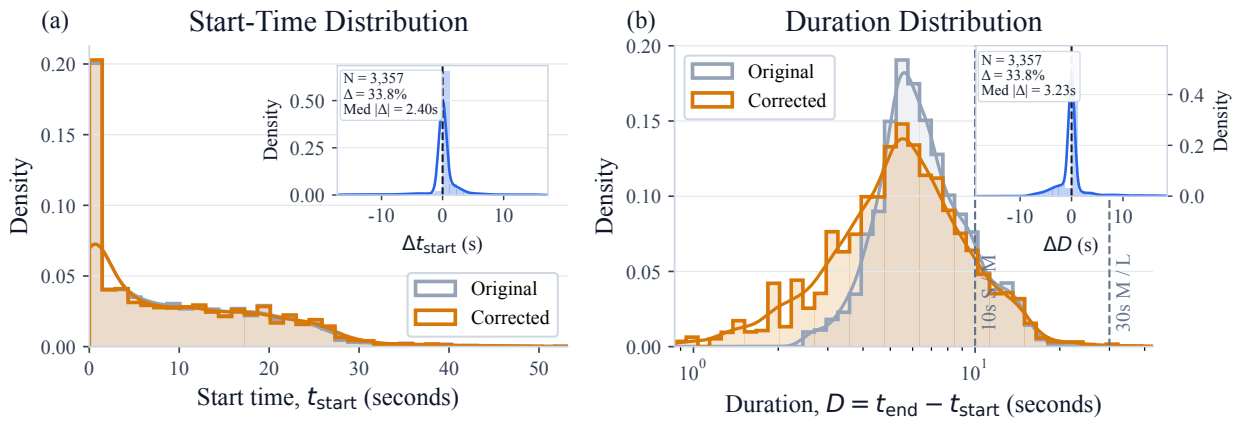


Figure 11. Temporal distribution changes after benchmark correction. **(a)** Start-time distributions of the original and corrected annotations, with the inset showing the paired start-time shift. **(b)** Duration distributions of the original and corrected annotations, with the inset showing the paired duration shift. Positive shifts indicate later corrected start times or longer corrected durations. The dashed vertical lines in the duration plot mark the 10s and 30s thresholds used for duration-stratified evaluation.

aries.

Correction Protocol and Model Assistance. For each candidate benchmark sample, the input consists of a video, a natural language query, and the original temporal annotation. We prompt Gemini-3-Pro to produce auxiliary temporal predictions, including a calibrated query, a refined temporal interval in absolute time, and concise reasoning about the queried event. Fig. 9 illustrates the prompting protocol. These model outputs are used as auxiliary evidence and are presented to annotators together with the original annotation.

Review Interface and Case Study. Fig. 10 shows the web-based review interface and a correction case. The interface is organized into several functional panels. The *Progress Panel* tracks the review queue, annotation progress, and the status of the current sample. The *Comparison Panel* presents the original query, the Gemini-assisted calibrated query, and a visual comparison between the original ground-truth interval and the Gemini-predicted interval. The *Video Panel* provides the full-video context and localized playback controls for both the original ground-truth span and the Gemini-predicted span. The *Edit Panel* allows annotators to revise the final query and temporal interval, while the *Decision Panel* records the final human decision as *Keep*, *Modify Query*, *Modify Span*, or *Delete*. The *Auxiliary Reasoning Panel* displays the model-generated reasoning used as auxiliary evidence during review. The interface also provides keyboard shortcuts for efficient annotation. These shortcuts allow annotators to rapidly compare localized clips while still making the final correction through direct video inspection. In the illustrated case, the original query is *"person opens up a cabinet."*, while the Gemini-assisted calibrated query specifies the event as *"A person slides open the glass door of a black TV cabinet."* The original annotation spans 6.1–12.5 seconds, whereas the Gemini-predicted interval spans 5.143–8.0 seconds. After comparing the localized clips in the *Video Panel* and inspecting the full-video context, the annotator revises the final interval to $[3.484, 10.5]$. The final annotation differs from both the original benchmark annotation and the auxiliary Gemini prediction, illustrating that the corrected boundary is determined by human review rather than directly copied from model output.

D.4 Data Statistics

We further analyze the temporal distributions of the curated training set and the human-verified benchmark. We focus on two temporal factors: the start time of a grounded moment and its duration.

The former reflects where supervision is distributed along the video timeline, while the latter reflects the diversity of temporal scales covered by the data. Fig. 12 shows the joint distribution between ground-truth start time and moment duration in the curated 90K-scale training set. The heatmap indicates that the training samples cover a broad range oftemporal locations and moment durations, rather than concentrating only on a narrow temporal regime. The duration

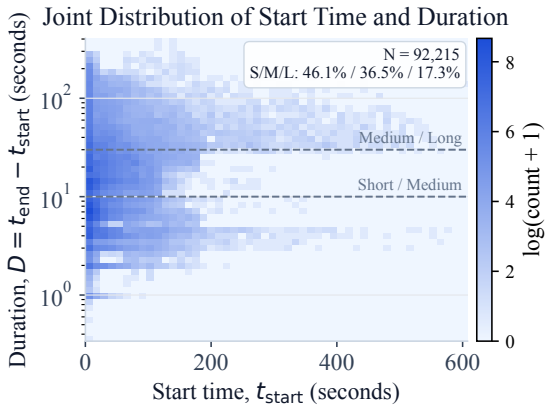


Figure 12. Temporal distribution of the curated 90K-scale training set. The x-axis denotes the ground-truth start time in seconds, and the y-axis denotes the moment duration $D = t_{\text{end}} - t_{\text{start}}$ in seconds. The color intensity represents $\log(\text{count} + 1)$. The duration axis is plotted on a logarithmic scale, and dashed horizontal lines indicate the 10s and 30s thresholds used for duration-stratified evaluation.

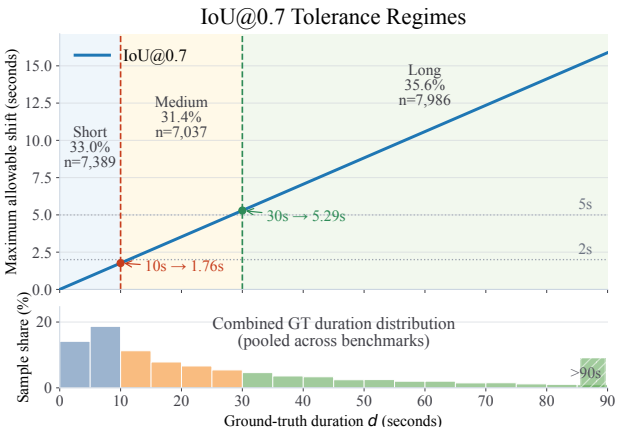


Figure 13. Boundary-tolerance regimes and pooled ground-truth duration distribution used for duration-stratified mIoU. The top panel plots the maximum allowable equal-duration temporal shift under IoU@0.7 as a function of ground-truth duration, with vertical dashed lines marking the 10s and 30s thresholds. The bottom panel shows the pooled ground-truth duration histogram across benchmarks using the same Short, Medium, and Long regimes.

axis is shown on a logarithmic scale to make both short and long moments visible, and the dashed horizontal lines mark the 10s and 30s thresholds used for duration-stratified evaluation.

Fig. 11 compares the original and corrected benchmark annotations. The main plots show the global start-time and duration distributions before and after correction, while the insets show the paired correction shifts. This visualization captures both the distributional change induced by benchmark correction and the magnitude of sample-level boundary adjustments. The corrected annotations preserve the overall temporal coverage of the benchmark while revising noisy temporal boundaries through human review.

E Evaluation Metric Design

E.1 Standard Temporal Grounding Metrics

We summarize the standard metrics used for video temporal grounding. For the n-th sample, let the predicted interval be $\hat{I}_n = [\hat{t}_n^s, \hat{t}_n^e]$ and the ground-truth interval be $I_n = [t_n^s, t_n^e]$. The temporal intersection and union lengths are

$$\ell_\cap(\hat{I}_n, I_n) = \max\left(0, \min(\hat{t}_n^e, t_n^e) - \max(\hat{t}_n^s, t_n^s)\right), \tag{22}$$

$$\ell_\cup(\hat{I}_n, I_n) = \max(\hat{t}_n^e, t_n^e) - \min(\hat{t}_n^s, t_n^s). \tag{23}$$

The temporal Intersection-over-Union is

$$\text{IoU}(\hat{I}_n, I_n) = \frac{\ell_\cap(\hat{I}_n, I_n)}{\ell_\cup(\hat{I}_n, I_n)}. \tag{24}$$

For a test set with N samples, mean IoU is computed as

$$\text{mIoU} = \frac{1}{N}\sum_{n=1}^{N} \text{IoU}(\hat{I}_n, I_n). \tag{25}$$

mIoU measures the average continuous overlap quality of predicted temporal intervals.

Recall@IoU measures thresholded localization success. Given a threshold η, it is defined as

$$\text{Recall@}\,\eta = \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}\left[\text{IoU}(\hat{I}_n, I_n) \geq \eta\right], \tag{26}$$

where common thresholds are $\eta \in \{0.3, 0.5, 0.7\}$. For methods that output multiple candidate intervals, Recall@K, IoU=η is computed as

$$\text{Recall@}\ K, \text{IoU} = \eta = \frac{1}{N}\sum_{n=1}^{N} \mathbf{1}\left[\max_{1\le k\le K} \text{IoU}(\hat{I}_{n,k}, I_n) \ge \eta\right]. \tag{27}$$

The single-interval prediction setting corresponds to $K = 1$.

E.2 Duration-Stratified mIoU and Boundary-Tolerance Analysis

Aggregate mIoU measures the average overlap quality over all test samples, but it mixes temporal moments with different intrinsic localization difficulty. In particular, short ground-truth intervals are more sensitive to absolute boundary errors: the same temporal shift can severely reduce IoU for a short action while having a much smaller effect on a long event. To make the evaluation more diagnostic, we report duration-stratified mIoU over Short, Medium, and Long ground-truth moments.

The duration buckets are determined by the ground-truth duration rather than the predicted duration, preventing a model from changing its evaluation bucket by altering its prediction length. For the n-th sample, let $d_n = t_n^e - t_n^s$ denote the ground-truth duration. We define

$$\begin{aligned} \mathcal{D}_S &= \{\, n : 0 < d_n \le 10 \,\}, \\ \mathcal{D}_M &= \{\, n : 10 < d_n \le 30 \,\}, \\ \mathcal{D}_L &= \{\, n : d_n > 30 \,\}. \end{aligned} \tag{28}$$

For each bucket $b \in \{S, M, L\}$, the duration-stratified mIoU is

$$\text{mIoU}_b = \frac{1}{|\mathcal{D}_b|}\sum_{n\in\mathcal{D}_b} \text{IoU}(\hat{I}_n, I_n). \tag{29}$$

We use a boundary-tolerance analysis under IoU@0.7 to interpret the 10s and 30s thresholds. Consider a ground-truth interval of duration d and a predicted interval with the same duration but shifted by δ seconds. Under this equal-duration shift model,

$$\text{IoU} = \frac{d - |\delta|}{d + |\delta|}. \tag{30}$$

Requiring IoU $\ge \eta$ gives

$$|\delta| \le d \cdot \frac{1-\eta}{1+\eta}. \tag{31}$$

For $\eta = 0.7$, this becomes $|\delta| \le 0.1765d$. Therefore, a 10-second moment allows only approximately 1.76s shift error, while a 30-second moment allows approximately 5.29s.

Figure 13 combines this boundary-tolerance curve with the pooled ground-truth duration distribution across benchmarks. The three regimes contain comparable portions of the evaluation samples, with 33.0% Short, 31.4% Medium, and 35.6% Long moments. Thus, the proposed duration split is both interpretable under IoU@0.7 and empirically meaningful for evaluating temporal grounding models across samples with different precision requirements.

F Additional Exploration of Progressive Interval Optimization

F.1 Motivation and Setup

Post-SFT optimization has become a common strategy for improving LLMs and VLMs beyond supervised fine-tuning, typically by using reward-guided policy updates, on-policy distillation, or response-level preference signals. This paradigm is also attractive for video temporal grounding, where the final evaluation depends on the quality of the interval decoded from generated responses. We therefore explore whether TimePLE can benefit from additional post-SFT optimization after the interval-supervised SFT stage.

However, post-SFT optimization for TimePLE is different from ordinary response-level optimization. TimePLE (SFT) already provides direct supervision to the generated <|TIMESPAN|> token through interval distributional loss, overlap loss, and boundary loss. As shown in the main ablation, applying a standard GRPO-style objective with commonly used

VTG rewards does not further improve over TimePLE (SFT). This suggests that outcome-level rewards computed from the final decoded interval may provide limited additional guidance once the latent interval interface has already been aligned by span-level supervision.

Motivated by this observation, we explore two interval-aware post-SFT strategies: *Counterfactual Span Distribution Optimization* (CSDO) and *Trust-Region Span Posterior Distillation* (TR-SPD). CSDO constructs a reward-improved target distribution over the canonical span grid by evaluating counterfactual movements around the old-policy prediction, while TR-SPD constructs posterior span targets under a trust-region constraint relative to the self-prior and distills the current span distribution toward accepted targets. All variants are initialized from the same TimePLE (SFT) checkpoint and trained with the same data, video processing settings, and evaluation protocol. The interval decoder is frozen during post-SFT optimization, so the additional objectives mainly affect the VLM hidden representation and span interface.

F.2 Explored Post-SFT Training Algorithms

We explore two post-SFT training algorithms that operate directly on the canonical span distribution associated with the generated <|TIMESPAN|> token. Both methods keep the interval decoder frozen and optimize the actor through the span distribution predicted from the current VLM hidden state. They differ in how the post-SFT target distribution is constructed.

Counterfactual Span Distribution Optimization. CSDO augments the response-level policy update with an auxiliary span-distribution objective. For each rollout sample, we first obtain a detached old-policy span distribution p_{old} and its corresponding span feature at the <|TIMESPAN|> position. The old-policy distribution is used to compute an expected canonical coordinate μ_{old}. Then, for each canonical grid cell c with coordinate x_c, we construct a local counterfactual coordinate by moving a small step from the old expectation toward that cell:

$$\mu_+(c) = (1-\eta)\mu_{\text{old}} + \eta x_c. \tag{32}$$

Using the detached old-policy span feature and the frozen duration-adaptive decoder, each counterfactual coordinate is decoded into a temporal interval and scored against the ground-truth span. The resulting reward improvement

$$A(c) = R(\text{decode}(\mu_+(c))) - R(\text{decode}(\mu_{\text{old}})) \tag{33}$$

measures whether moving probability mass toward cell c would improve the decoded interval. We then form a reward-improved target distribution over the canonical span grid:

$$q(c) = \text{softmax}\left(\log p_{\text{old}}(c) + \frac{\widetilde{A}(c)}{\tau}\right), \tag{34}$$

where $\widetilde{A}(c)$ denotes the normalized counterfactual advantage. The current actor distribution p_{cur} is trained to match this detached target through a cross-entropy loss, optionally regularized by a span-level KL term to the reference distribution.

This design aims to provide a more geometry-aware signal than scalar response-level rewards. Instead of assigning a single reward to the final decoded interval, CSDO estimates which regions of the canonical span grid would locally improve the decoded interval and uses this information to reshape the current span distribution. However, the supervision is still derived from outcome-level interval rewards and therefore provides only indirect credit assignment to the hidden-state geometry.

Trust-Region Span Posterior Distillation. TR-SPD follows a different post-SFT strategy. Rather than constructing a local counterfactual target during each actor update, it builds posterior span targets from the old-policy span distribution under a self-prior trust-region constraint. Let p_b denote the detached old-policy span distribution used as the self-prior. For a set of temperature candidates, TR-SPD constructs posterior candidates q_τ and accepts a candidate only when its deviation from the self-prior satisfies

$$D_{\text{KL}}(q_\tau \,\|\, p_b) \le \rho. \tag{35}$$

If no posterior candidate satisfies the trust-region budget, the sample falls back to weak self-prior retention. The accepted posterior target is then used as a detached teacher distribution for span-level distillation.

Table 4. Post-SFT optimization results initialized from the same TimePLE (SFT) checkpoint. C-STA, A-Net, QVH, and C-TPLE denote Charades-STA, ActivityNet-Captions, QVHighlights, and Charades-TimePLE, respectively. Numbers in parentheses report mIoU changes relative to TimePLE (SFT).

Training setting	C-STA	A-Net	QVH	C-TPLE	Avg.
TimePLE (SFT)	**57.2**	**49.6**	**65.3**	**63.4**	**58.9**
+ GRPO	57.2 (0.0)	49.6 (0.0)	65.1 (↓ 0.2)	63.3 (↓ 0.1)	58.8 (↓ 0.1)
+ CSDO	56.9 (↓ 0.3)	48.9 (↓ 0.7)	63.4 (↓ 1.9)	62.5 (↓ 0.9)	57.9 (↓ 1.0)
+ TR-SPD	57.2 (0.0)	48.4 (↓ 1.2)	63.4 (↓ 1.9)	62.9 (↓ 0.5)	58.0 (↓ 0.9)

In the online training stage, TR-SPD optimizes the current span distribution toward the accepted posterior target while using text-level KL regularization to preserve the response behavior. Unlike CSDO, this algorithm does not rely on the standard response-level policy-gradient loss as the main optimization signal. Instead, it treats post-SFT training primarily as a posterior distillation problem over the canonical interval grid. The trust-region constraint is introduced to avoid moving the teacher distribution too far from the already learned TimePLE (SFT) span prior, while still allowing reward-improving posterior targets to guide the actor.

The two algorithms therefore represent complementary attempts to improve TimePLE after interval-supervised SFT. CSDO performs online counterfactual reward shaping over the span grid, whereas TR-SPD performs trust-region posterior distillation from self-prior span distributions. Both are specifically designed for the interval-native prediction space, but neither assumes that ordinary response-level rewards alone are sufficient to refine the latent <|TIMESPAN|> representation.

F.3 Experimental Results

Table 4 reports the results of the explored post-SFT optimization strategies. The results show that none of the post-SFT methods provides a stable improvement over the interval-supervised SFT checkpoint. Standard GRPO remains nearly unchanged, while CSDO and TR-SPD lead to small performance drops on average despite their interval-aware target construction.

Overall, the post-SFT results support the observation from the main ablation: after interval-supervised SFT, additional reward-style or posterior-distillation training does not reliably improve TimePLE. Although CSDO and TR-SPD introduce span-distribution targets that are more aligned with the canonical interval space than ordinary response-level rewards, their improvements do not transfer to consistent benchmark-level gains.

F.4 Mechanistic Analysis of Post-SFT Optimization

The post-SFT results suggest that the main optimization challenge in TimePLE is not response selection, but latent interval geometry. In conventional instruction tuning or reward-guided VLM training, the optimized object is usually close to the model's native output space: token probabilities, response preferences, or scalar rewards assigned to complete generations. In contrast, TimePLE localizes a moment through the hidden state of a generated <|TIMESPAN|> token, which is decoded into a distribution over the canonical interval space. Therefore, effective optimization must shape not only the generated response format, but also the geometry of the span-token representation before decoding.

This explains why interval-supervised SFT remains the strongest training signal in our pipeline. During TimePLE (SFT), the <|TIMESPAN|> hidden state is directly aligned with ground-truth interval distributions through distributional, overlap, and boundary losses. These losses provide dense supervision in the same interval space used by the decoder. By contrast, CSDO and TR-SPD construct post-SFT targets from the model's own old-policy span distribution. CSDO moves probability mass according to counterfactual decoded reward improvements, while TR-SPD distills posterior targets constrained by a self-prior trust region. Although both targets are more interval-aware than ordinary response-level rewards, they are still bootstrapped from the current model's span prior and decoded outcomes rather than from new external interval evidence.

The limited gains therefore indicate a structural limitation of the explored post-SFT objectives. If the target remains too close to the self-prior, it adds little information beyond the already supervised SFT solution. If the target moves too far from the self-prior, it can disturb the learned interface between language generation and interval decoding. This tension is especially important for TimePLE because the interval decoder is frozen during post-SFT training, so all

additional optimization must be absorbed by the VLM hidden representation and span interface. The results suggest that further improvement may require post-SFT signals that are denser and less self-referential, such as correction-aware interval distillation, external teacher distributions, or geometry-consistent supervision that directly constrains the canonical interval distribution.

Findings

Post-SFT optimization for TimePLE is fundamentally different from ordinary reward-guided response optimization. Since the prediction is mediated by the latent <|TIMESPAN|> representation, the most effective signal is one that directly supervises the canonical interval distribution. CSDO and TR-SPD make the post-SFT objective more interval-aware, but their targets are still derived from the model span prior and decoded outcomes. This explains why they do not consistently improve over interval-supervised SFT and suggests that future post-SFT training should rely on denser, externally grounded, and geometry-consistent interval targets.