

TF-TimeCoVe: Zero-Shot Video Temporal Grounding via Chain of Verification

Achraf Hafsaoui

Alexey Maxyuta

Georg Matthes

Kevin Ament

Ahmed Babay

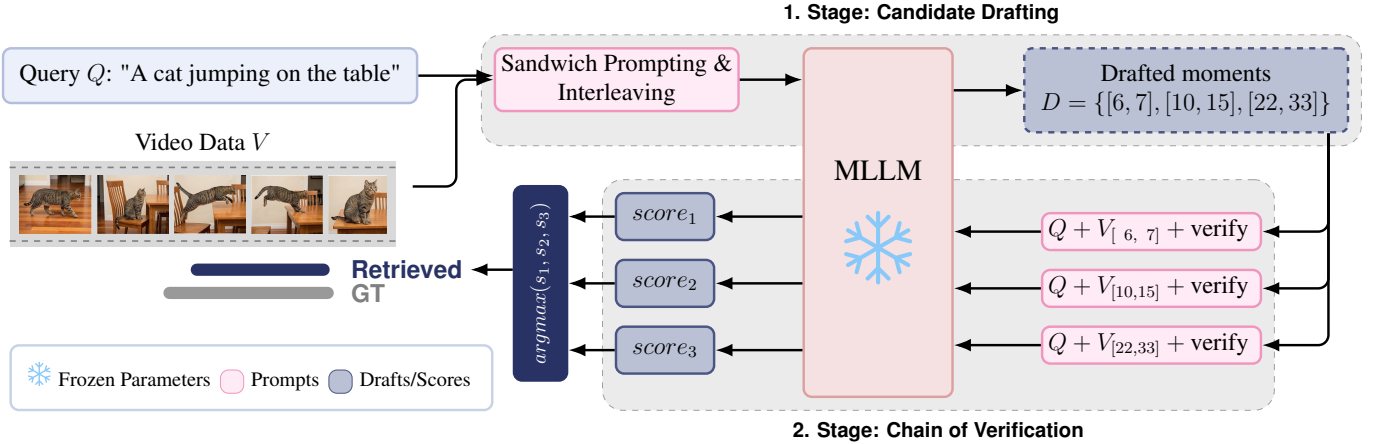


Figure 1. **Overview of TF-TimeCoVe:** A Training-Free (TF) Video Temporal Grounding (VTG) pipeline that enhances the zero-shot capabilities of Multimodal Large Language Models (MLLMs). In Stage 1, given a video V and a natural language query Q , the model drafts candidate moments using timestamp-interleaved frames and sandwich prompting. In Stage 2 (Chain of Verification), these candidate drafts are individually verified by re-prompting the frozen MLLM. The final moment is then retrieved based on the generated verification scores.

Abstract

Recent progress in Multimodal Large Language Models has expanded their ability at joint visual-textual interpretation, but precise Video Tempo Grounding remains a critical challenge. These models suffer from two primary limitations. First, they lack an explicit notion of time, resulting in weak temporal perception. Second, they rely on single-pass generation paradigms, forcing the model to predict timestamps without verifying its own outputs. Furthermore, while Training-Free (TF) approaches eliminate the computational costs of fine-tuning, they still struggle to bridge this temporal gap. To overcome these limitations, we introduce TF-TimeCoVe, a fully TF framework that treats temporal grounding as a draft-then-verify process. Our method builds on the Chrono blueprint and strengthens temporal awareness through structured prompting, including query sandwiching, which reduces long-context forgetting. We then decouple moment retrieval into two stages. In the candidate drafting stage, the model proposes plausible temporal windows. In the Chain-of-Verification stage, we evaluate candidates using three different verification techniques and produce the final prediction. Through extensive experiments we demonstrate that both 8B and 32B variants of TF-TimeCoVe significantly outperform all existing TF approaches. Specifically, our method yields a novel approach and establishes new TF SOTA scores across three standard benchmarks for $R1@0.5$: Charades-STA (+7.5%), ActivityNet Captions (+9.2%), and the multi-moment QVH dataset (+10.9%).

1. Introduction

The ability to accurately localize specific events within videos, known as Video Temporal Grounding (VTG), is a core challenge in Multimodal Artificial Intelligence. Unlike coarse video understanding, VTG requires precise alignment between natural language descriptions and fine-grained temporal boundaries. It forms the foundation for numerous real-world applications, including video retrieval, assistive robotics, and downstream tasks such as video question answering. Crucially, VTG requires reasoning over visual sequences where relevant events may be sparse, visually subtle, or temporally ambiguous, making reliable localization fundamentally more difficult than coarse-grained video understanding.

Concurrently, the rapid evolution of Large Language Models (LLMs) has caused a paradigm shift in multi-modal learning, transitioning the field from task-specific specialists to generalist Multimodal Large Language Models (MLLMs) such as GPT-5 [18], Gemini-3 [6] and Qwen3-VL [29]. By leveraging sophisticated modality connectors to align visual, auditory, and sensor data with the textual embedding space, MLLMs have demonstrated remarkable zero-shot capabilities in Video Understanding, Question Answering, and VTG [13]. Unlike traditional fully-supervised methods that rely on pre-defined temporal boundary regression, MLLM-based approaches leverage the inherent linguistic capabilities of LLMs to localize video segments corresponding to natural language queries. To further bridge the gap between coarse understanding and fine-grained localization, recent works have introduced time-aware visual cues [8, 32] or interleaving of timestamps and frames [16, 33] to effectively bring

videos and time closer.

Despite these architectural advances, state-of-the-art MLLMs still struggle severely with precise temporal localization [16, 33]. Rather than relying on robust temporal visual evidence, MLLMs frequently hallucinate plausible timestamps based on language priors [1]. This limitation stems primarily from their underlying vision encoders; because they are heavily optimized for static image-text alignment, these models suffer from pervasive static feature biases [4, 23]. For example, when tasked with grounding the query "a man opens a refrigerator," MLLMs frequently utilize the mere spatial presence of the target objects as a shortcut [21, 24]. Consequently, they confidently anchor their predictions to the earliest frames where the man and refrigerator co-occur, failing to localize the actual temporal onset of the kinematic action. Correcting this object-centric bias through standard fine-tuning incurs prohibitive computational and data annotation costs. To circumvent this, recent literature explored different Training-Free (TF) approaches [11, 16, 20, 25, 27, 32, 34]. While these methods effectively utilize MLLMs, their VTG precision remains limited by two primary reasons. First, most TF methods suffer from a perception-time gap: they process frame embeddings to establish semantic similarity with the text but lack an explicit temporal representation [11, 34]. Second, current methods predominantly rely on a single-pass generation strategy. Requiring the model to interpret visual content and predict timestamps, without self-verification [16, 33], which can cause contextual amnesia and produce repetitive or degenerate outputs [3].

To overcome these limitations, we do not merely propose a single heuristic fix, but we systematically explore the components required to push TF temporal grounding beyond single-pass generation. By deconstructing the inference pipeline into prompt design, temporal grounding cues, verification architectures, and scoring signals, we critically assess how each element bridges the gap between general MLLM reasoning and precise temporal boundary prediction. Following this exploration, we introduce the **Training Free Time Chain-of-Verification** (TF-TimeCoVe) framework, built upon the Qwen3-VL [29] architecture. As illustrated in Fig. 1, TF-TimeCoVe fundamentally shifts VTG from a purely autoregressive guessing task to a rigorous, evidence-backed draft-then-verify pipeline [3]. To address the perception-time gap, our drafting phase instills an explicit feeling of time [16, 33] to generate candidate windows, actively mitigating long-context amnesia via prompt optimization techniques such as sandwich prompting. Furthermore, to overcome the flaws of single-pass generation, we discard the standard practice of accepting initial predictions. Instead, our verification phase evaluates each candidate window independently to filter out temporal hallucinations. We systematically benchmark multiple verification mechanisms, including confidence, semantic similarity, and their fusion. Crucially, we introduce a novel contrastive mechanism utilizing Textual and Visual Log-Likelihood Ratios to formally measure the model’s underlying support for its own generated boundaries

Through extensive experiments, we demonstrate that TF-TimeCoVe consistently outperforms all existing state-of-the-art (SOTA) TF methods on Charades-STA [5], QVHighlights [12], and ActivityNet Captions [10]. On Charades-STA, our 32B model achieves an R1@0.5 of 58.55, marking an absolute gain of 7.53%. On ActivityNet Captions, it reaches an R1@0.5 of 40.24, yielding a 9.18% improvement. Finally, on the multi-

moment QVHighlights dataset, it achieves 72.58 R1@0.5, a notable 10.9% improvement over the best prior TF approach. In summary, our main contributions are as follows: (1) We introduce TF-TimeCoVe, a fully TF framework that shifts VTG from a single-pass prediction task to a robust, self-verifying pipeline. To strengthen the drafting phase, we use structural prompt interventions, such as query sandwiching, to mitigate long-context amnesia during dense frame processing. (2) We systematically design and evaluate multiple zero-shot verification architectures to decouple candidate generation from temporal evaluation. Crucially, we introduce a novel contrastive scoring mechanism based on Visual Log-Likelihood Ratios, identifying it as an effective discriminative signal for filtering temporal hallucinations in complex benchmarks like QVHighlights [12]. (3) We demonstrate through extensive evaluations that our approach establishes a SOTA for TF VTG across multiple datasets.

Through this systematic exploration, we derive actionable insights and best practices that address key bottlenecks in zero-shot VTG for MLLMs. TF-TimeCoVe provides a foundation for self-verifying MLLMs, enabling highly accurate VTG without the need for fine-tuning.

2. Related Work

2.1. Training-based VTG

Current MLLM-based VTG methodologies can be broadly categorized into three training paradigms: Pre-training, Fine-tuning, and Training-Free approaches [26]. Pre-training and Fine-tuning frameworks typically introduce learnable temporal adapters or instruction-tuning datasets to align visual features with temporal concepts explicitly [7, 20, 30, 33].

However, the reliance on large-scale supervision is problematic. Zhang et al. [33] recently exposed significant noise and ambiguity in standard VTG benchmark annotations. By introducing the TimeLens-100K dataset, they show that data quality can substantially alter model rankings, suggesting that previously reported state-of-the-art improvements may partially stem from noisy ground truth.

To address this issue, TimeLens modifies existing datasets such as ActivityNet Captions, Charades-STA, and QVHighlights. In particular, QVHighlights is reformulated to enforce a *single-moment* grounding setting, although the dataset was originally designed to evaluate *multiple-moment* grounding. While this improves annotation consistency, it simplifies the task by removing scenarios where a query corresponds to multiple disjoint temporal segments. In contrast, our work evaluates VTG in its original setting using the unmodified QVHighlights benchmark. Furthermore, the authors note that reasoning-intensive VTG scenarios remain largely unexplored. In particular, they state that certain grounding tasks may require stronger reasoning capabilities beyond simple visual matching. Our approach directly addresses this challenge by introducing a TF *generate-then-verify* framework that enables MLLMs to reason over multiple candidate moments and systematically verify their temporal validity.

Another training-based approach VTimeLLM [7] that is designed to improve temporal understanding of video content by introducing a boundary-aware three-stage training strategy. The model integrates a visual encoder and adapter to extract video features, which are then processed by a large language model capable of jointly reasoning over visual and textual information. VTimeLLM presents several limitations: First, the approach re-

lies on a complex multi-stage supervised training pipeline that requires large-scale image-text and temporally annotated video datasets. Training-based approaches also require substantial computational resources and re-training when adapting to new tasks or datasets. In contrast, our approach follows a TF paradigm that directly leverages the reasoning capabilities of pre-trained multi-modal large language models.

2.2. Training-Free VTG

Although in contrast, *Training-Free* approaches have emerged as a robust, compute-efficient alternative, bypassing expensive model updates by exploiting the zero-shot capabilities of frozen LLMs [27, 34] and temporal reasoning capabilities of off-the-shelf MLLMs for Video Temporal Grounding [11, 16, 27, 32, 34] many existing methods remain limited by various constraints.

Two emerging paradigms in this domain are *visual prompting* and *iterative refinement*. Approaches such as NumPro [27] and TAG [11] reformulate grounding as a visual reasoning task, overlaying numeric markers or progress bars directly onto video frames.

Simultaneously, methods like HawkEye [25] adopt a recursive coarse-to-fine strategy to narrow down temporal windows. However this recursive strategy is optimized to converge on a single dominant moment, making it not suited for multiple moment retrieval scenarios (e.g., QVHighlights [12]). In contrast, our approach generates and evaluates multiple candidate temporal windows and employs a verification-based scoring pipeline to select the most plausible moments, enabling more flexible retrieval without requiring recursive inference.

The TimeChat [20] model introduces a timestamp-aware frame encoder that explicitly associates visual content with frame timestamps and a sliding video Q-Former that compresses video frames into a variable-length token sequence to handle videos of different durations. However, the authors note that the generated outputs still lack richness and fine-grained detail in video summarization. By iteratively generating potential moments and validating them using multiple signals, our method encourages more precise and detailed grounding of events, enabling the model to capture richer temporal information without relying solely on compressed video representations.

VTimeCoT [32] introduces a visual time chain-of-thought representation using a video progress bar that is overlaid onto video frames. However, the authors report that standard Chain-of-Thought (CoT) reasoning alone performs poorly for temporal grounding, leading to significant performance degradation suggesting that purely text-based step-by-step reasoning is insufficient for reliable temporal localization. To mitigate hallucinations in open-ended text generation, recent works propose structured consistency checks over simple forward passes. Dhuliawala et al. [3] introduced the *Chain-of-Verification (CoVe)* framework, which moves beyond intermediate reasoning steps (CoT) to a systematic deliberation phase where the model generates self-correction questions. While these methods enhance logical robustness in the textual domain, they lack mechanisms to evaluate temporal visual evidence. We bridge this gap by reformulating CoVe for VTG: our approach grounds verification steps in candidate-windows evaluation after initially drafting them.

In general LLMs also exhibit a *lost-in-the-middle* phenomenon [15], where reasoning degrades for mid-sequence input tokens. MLLMs inherit this performance, creating a critical bot-

tleneck for VTG where target events are uniformly distributed across video inputs. To counteract this, we employ *sandwich prompting*. By bracketing the visual sequence with pre- and post-prompts, we leverage the model’s inherent recency bias, forcing the attention mechanism to re-process the query constraints immediately after the multi-dimensional tokens. This ensures the task definition to remain prominent.

To address all of the limits of the TF approaches, we build upon *Chrono* [16], which provides a simple yet effective blueprint for video temporal representations. Chrono demonstrates that simply interleaving discrete integer timestamps with visual tokens enables image-text pre-trained MLLMs to grasp temporal boundaries. We adopt the Chrono blueprint because it offers a robust and flexible *representation* of time for MLLMs without the overhead of external tools. However, while Chrono effectively maps visual content to timestamps, the nature of LLMs decoding can still lead to hallucinations of the visual context [9]. We bridge this gap by augmenting the Chrono baseline with a systematic candidate generator for inference-time verification. This enables the reliable detection of multiple disjoint events within a strictly TF framework.

3. Method

Our method builds on Multimodal Large Language Models (MLLMs) in a Training-Free (TF) setting and relies on carefully designed prompts to maintain temporal alignment across visual sequences. Therefore, we first outline the prompt-design principles used throughout our approach. Based on the explorative insights from Sec. A.1, we adopt the *Short + Sandwich* configuration as the prompt template for all approaches in this section. In the following subsections, we formalize the general task of VTG (Sec. 3.1), detail the baseline we build upon (Sec. 3.2), and introduce our verification process and overall framework (Secs. 3.3 and 3.4). Finally, we describe the specific verification techniques implemented within our framework (Sec. 3.5).

3.1. Training-Free Video Temporal Grounding

Given a video V and a natural-language query q describing an event or action, Video Temporal Grounding (VTG) identifies the set of temporal intervals $[t_{\text{start}}, t_{\text{end}}]$ in V that match the query. The video V is represented as a sequence of frames $\{f_1, f_2, \dots, f_T\}$ sampled at a constant frame rate. In this work, we use a strictly TF setting: the underlying MLLM remains frozen, with no training or fine-tuning.

3.2. Chrono Baseline

We use Chrono [16] as our *baseline* for TF VTG. It introduces temporal awareness into MLLMs by interleaving absolute integer timestamps with sampled video frames directly in the input sequence. This simple blueprint provides an effective temporal encoding mechanism without requiring any additional training [33].

Finding 1: Encoding timestamps as interleaved textual prefix is the essential for VTG.

3.3. JointCoVeVTG

To improve the reliability of VTG predictions in MLLMs, we use a *Chain-of-Verification (CoVe) method* [3] as a guiding mechanism. CoVe shows that MLLMs often hallucinate facts, but can

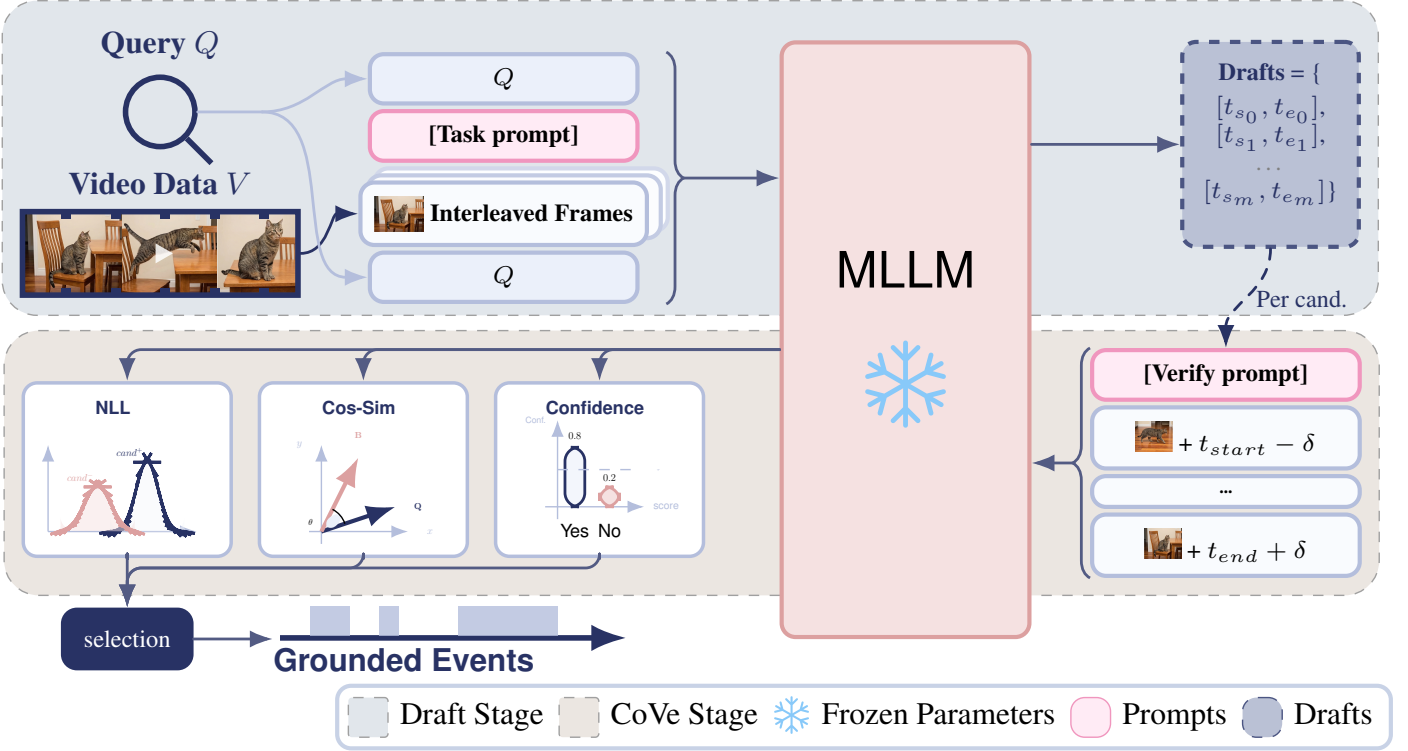


Figure 2. **TF-TimeCoVe pipeline overview.** Draft Stage: We feed the video and query to a frozen MLLM, which generates initial temporal proposals. CoVe Stage: We evaluate each candidate using multiple signals, extracting confidence scores and binary answers, measuring semantic alignment via cosine similarity between the query and generated reasoning, and computing textual and visual likelihood ratios against negative baselines. We then use these scores to select the final prediction(s).

correct themselves when required to verify their initial assumptions. This insight motivates us to integrate an internal verification. We refer to this single-pass verification setup as *JointCoVeVTG*. In *JointCoVeVTG*, we ask the model to propose candidate moments and verify them within the same prompt. The model must immediately justify its prediction by answering verification questions about its proposals. We obtain these questions through two approaches. The underlying MLLM generates them directly within the same prompt, or we pre-compute them once using GPT-4o [17]. Example questions are in the appendix in Fig. 11 as well as the prompts (Figs. 9 and 10). For the pre-computed variant, we use three verification questions that check temporal consistency before, during, and after the queried action. Within the same prompt, we instruct the model to select the most appropriate windows based on its own verification answers. Despite promising motivation behind CoVe, *JointCoVeVTG* overloads the model. The query itself, interleaved frames, the moment proposal, verification questions, and reasoning guidelines all appear in a single prompt. This joint CoVe approach is similar to Chain-of-Thought methods. Thus the design does not mitigate existing biases or shortcut reasoning in MLLMs [21, 24].

Finding 2: JointCoVeVTG remains susceptible to biases, hallucinations, and shortcut-reasoning errors because it relies on a single-pass generation scheme.

3.4. TF-TimeCoVe Framework

To solve the limitations of single-pass *JointCoVeVTG*, we split candidate generation and verification into two passes [2, 14]:

drafting and then verifying. We refer to them as *Draft Stage* and *CoVe Stage* respectively. Building on this two-stage design, we introduce the **Training Free Time Chain-of-Verification (TF-TimeCoVe)** framework (Fig. 2).

Draft Stage Using the *Chrono baseline* (Sec. 3.2) and *prompt template* (6), we prompt the model to generate a high-recall set of temporal windows that likely contain the queried event.

Finding 3: Candidate Drafting Stage combines timestamp interleaving with sandwich prompting to produce high-recall temporal proposals.

CoVe Stage We now verify each drafted candidate window $x_w = [t_{\text{start}}, t_{\text{end}}]$. By expanding the video segment with a buffer, we provide temporal context to the model. We set the buffer size relative to the window length $L_w = t_{\text{end}} - t_{\text{start}}$, constructing

$$\tilde{x}_w = [t_{\text{start}} - \alpha L_w, t_{\text{end}} + \alpha L_w], \quad (1)$$

where α controls the buffer size. Following the *Chrono baseline* (Sec. 3.2), we sample frames from $\tilde{x}_w$ and interleave them with timestamps. We include the expanded window duration in the prompt and instruct the model to generate verification signals for the original window x_w . Using these signals, we compute verification scores as described in Sec. 3.5. Afterwards we rank candidate windows to determine the final grounded moments. For single-moment retrieval, we select the highest-scoring window; for multi-moment retrieval, we retain all windows exceeding a threshold.

Priority	Pattern ($c_1, c_2, c_3 > 50$)	Description
3	(✓, ✓, ✓)	Complete verification (Start, Action, End)
2	(✓, ✓, ×) or (×, ✓, ✓)	Core action verified with one boundary
1	(✓, ×, ✓) or (×, ×, ×)	Boundary-only or action-only verification

Table 1. **Confidence-Based Self-Verification: Pattern Priority Hierarchy for single moment retrieval.** Binary patterns indicate whether c_i yielded a positive response with confidence exceeding the 50% threshold.

3.5. Verification Techniques

We evaluate each drafted candidate window using multiple verification techniques that capture temporal correctness, semantic similarity, and text likelihood, forming a solid basis for ranking and selecting final moments.

Confidence-Based Self-Verification. We use the model’s self-reported confidence as a verification signal. For each candidate window x_w and query q , we ask the model three binary (yes/no) temporal questions and request a confidence score $conf_i \in [0, 100]$ for each response (Fig. 7). We mark a window as *verified* if

$$verified \iff (conf_2 > 50) \vee (conf_1 > 50 \wedge conf_3 > 50), \quad (2)$$

where $conf_1$, $conf_2$, and $conf_3$ correspond to confidence scores for the following query-conditioned temporal questions (Fig. 11):

1. Start check ($conf_1$): Does q begin at t_{start} ?
2. Core action ($conf_2$): Is q occurring within $[t_{start}, t_{end}]$?
3. End check ($conf_3$): Is q completed by t_{end} ?

For single-moment retrieval, we rank verified candidates using the *Pattern Priority Hierarchy* (Tab. 1) and break ties with a weighted mean confidence score:

$$\bar{c} = 0.25 conf_1 + 0.50 conf_2 + 0.25 conf_3, \quad (3)$$

giving more weight to the core action check. If a tie remains, we select the longest window. For multi-moment retrieval, we retain all verified windows.

Similarity-Based Verification. We verify each candidate window by measuring the semantic alignment between the query and the model’s explanation for the core-action verification question described in Section 3.5. Specifically, we embed both the query and the generated reasoning text using a sentence-embedding model [19] and compute their cosine similarity, producing a similarity score for each window. Based on these scores, we select the window with the highest value for single-moment retrieval. For multi-moment retrieval, we keep all windows above a predefined threshold $\mathcal{T}$.

Confidence-Similarity Fusion Technique. We combine self-reported confidence and semantic similarity into a single score. For each candidate window, we compute:

$$S_\lambda(x_w) = \lambda \tilde{C}(x_w) + (1 - \lambda) \text{sim}(q, y_w), \quad (4)$$

where $\tilde{C}(x_w) = \frac{C(x_w)}{100}$ is the normalized confidence for the core-action question, $\text{sim}(q, y_w)$ is the cosine similarity between the

query q and the model’s reasoning y_w , and $\lambda \in [0, 1]$ balances the two signals. We perform final prediction as described in Sec. 3.5.

Likelihood Based Verification. We verify candidate windows by checking how consistently the MLLM supports its own generated text across different visual contexts. For each window x_w , we prompt the model to describe it (Fig. 8) and compute the likelihood of this description under multiple visual inputs. This contrastive signal provides two verification scores: Textual Negative Log-Likelihood (TNLL) and Visual Negative Likelihood (VNL). Again, we rank and select as previously described in Sec. 3.5.

Textual Negative Log-Likelihood Based Verification: Given a textual description y generated by the underlying MLLM for a candidate window x_w , we compute the token-averaged negative log-likelihood (NLL) of y when conditioned on the candidate window x_w and on a negative window x_n , which serves as a distractor context for contrastive evaluation. Using teacher forcing with the underlying MLLM, we compute the TNLL based verification ratio as:

$$S_{\text{TNLL}} = \frac{1}{|y|} [\log p(y | x_w) - \log p(y | x_n)], \quad (5)$$

This score quantifies the model’s relative support for the candidate window. A higher positive value indicates that the model finds its own description higher compatible with the candidate window than with the distractor context. When comparing multiple candidate windows, higher scores indicate stronger alignment with the queried event. In this work, we consider two types of negative windows x_n : (1) non-candidate segments from the same video and (2) black frames, both having the same length and using the same sampling strategy as x_w . Non-candidate segments act as soft negatives that share visual context with the candidate window, while black frames provide context-free hard negatives. We *hypothesize* that black frames serve as a stricter visual distractor. Based on these settings, we define two TNLL-based verification variants: $\text{TNLL}_{\text{Same}}$ and $\text{TNLL}_{\text{Black}}$.

Visual Negative Likelihood Based Verification: Inspired by TNLL based verification, we evaluate candidate windows in the visual domain. Here we measure the alignment between the generated textual description and visual representations of the candidate window. Given a description y generated for a candidate window x_w , we compute a VNL-based verification score by comparing the cosine similarity between the text embedding $e(y)$ and the visual embeddings of the candidate window $e(x_w)$ and a negative window $e(x_n)$:

$$S_{\text{VNL}} = \cos(e(y), e(x_w)) - \cos(e(y), e(x_n)), \quad (6)$$

where $\cos$ denotes cosine similarity, and all embeddings $e(y)$, $e(x_w)$, and $e(x_n)$ are obtained from the same underlying MLLM. A higher positive score indicates stronger visual-semantic alignment between the generated description and the candidate window relative to the negative window. When ranking multiple candidate windows, higher scores reflect greater visual consistency with the queried event. Analogue to TNLL, we define two VNL-based verification variants: VNL_{Same} and $\text{VNL}_{\text{Black}}$.

4. Experiments

We conduct a series of experiments to evaluate the effectiveness of our Training-Free (TF) TF-TimeCoVe framework. First, we

Method	QVHighlights		Charades-STA		ActivityNet	
	R1@.5 $\uparrow$	R1@.7 $\uparrow$	R1@.5 $\uparrow$	R1@.7 $\uparrow$	R1@.5 $\uparrow$	R1@.7 $\uparrow$
<i>Fine-Tuned</i>						
VTimeLLM-13B [7]	–	–	34.30	14.70	29.50	14.20
TimeLens-7B [33]	63.60	43.70	39.80	14.50	35.20	19.70
Chrono-BLIP [16]	74.77	60.51	69.31	49.29	53.92	35.55
<i>Zero-Shot</i>						
HawkEye [25]	–	–	31.40	14.50	29.30	10.70
TimeChat [20]	–	–	32.20	13.40	–	–
NumPro (Qwen2-VL-72B) [27]	–	–	36.80	15.90	26.40	14.40
VTimeCoT (GPT4o) [32]	59.74	33.81	<u>51.02</u>	22.45	–	–
TFVTG [34]	–	–	49.97	24.32	27.02	13.39
TAG [11]	–	–	48.58	<u>26.67</u>	28.91	15.07
Chrono-GPT [†] [16]	<u>61.68</u>	<u>41.80</u>	28.76	10.99	<u>31.06</u>	<u>17.51</u>
Qwen3-VL-8B (frames only) [†]	4.58	1.55	22.31	6.48	8.94	3.45
Chrono-Qwen3-VL-8B [†]	45.68	31.48	32.74	13.58	30.34	17.86
TF-TimeCoVe-8B[†]	65.94 +4.26	50.06 +8.26	56.16 +5.14	29.38 +2.71	34.26 +3.20	19.64 +2.13
Qwen3-VL-32B (frames only) [†]	4.13	1.29	9.89	3.23	8.37	3.45
Chrono-Qwen3-VL-32B [†]	59.42	42.13	26.02	10.08	30.70	18.47
TF-TimeCoVe-32B[†]	72.58 +10.90	57.10 +15.30	58.55 +7.53	30.46 +3.79	40.24 +9.18	24.27 +6.76

Table 2. **Comparison to state-of-the-art (SOTA) methods in Moment Retrieval on QVHighlights, Charades-STA, and ActivityNet Captions.** Prior Training-Free (TF) SOTA is underlined, new TF SOTA is bold. Gains for TF-TimeCoVe are in comparison to prior TF SOTA. [†] indicates QVHighlights results are reported on the validation set.

describe the experimental setup (Sec. 4.1). Next, in Sec. 4.2 we present quantitative results comparing our method against prior zero-shot approaches and ablate key components of our two-stage verification pipeline. Finally, we provide a qualitative analysis in Sec. 4.3 to illustrate the model’s strengths.

4.1. Experimental Setup

To evaluate our TF VTG approach, we utilize three standard benchmarks for VTG. Charades-STA from 2017 [5] contains 9,848 videos (avg. duration 30.6s) and 16,128 annotations. Because we are TF, we evaluate directly on the standard test split (3,720 annotations). QVHighlights [12](QVH) published in 2021, features 10,148 videos each with a duration of 150s. The queries often map to multiple moments. We report performance on the publicly available val split (1,550 videos), as the private test split has a limited number of submissions. ActivityNet Captions [10], released in 2017, consists of 20,000 videos (avg. duration 2 mins) and 72,000 annotated segments. Following standard practice [16, 22, 28, 31], we perform our zero-shot inference on the val_2 split (17,031 segments).

To evaluate our approach, we adopt standard metrics for both single- and multi-moment retrieval tasks. On single-moment datasets, we measure performance using Recall@ τ_r ($R@ \tau_r$) at Intersection over Union (IoU) thresholds $\tau_r \in \{0.5, 0.7\}$. For the multi-moment QVH dataset, we supplement this by reporting the mean Average Precision (mAP) and mAP@ τ_{map} and the IoU thresholds $\tau_{map} \in \{0.5, 0.75\}$.

The backbone of our approach is Qwen3-VL [29] in both 8B and 32B parameter variants. We benchmark our results against Chrono-GPT [16] as the primary baseline. Similar to the baseline we do not perform any fine-tuning or parameter updates. Instead, we optimize performance strictly through inference-time

prompting and the introduction of a novel verification and scoring procedure. For consistency, we adopt the sampling strategy of Chrono-GPT [16] but introduce these modifications: (a) We use a temperature of 0.7 for candidate generation. (b) In the dedicated CoVe Stage, we sample frames at 2 frames per second (FPS), capped by the dataset’s frame limit during drafting. (c) We enrich local context by expanding each candidate segment with a temporal buffer as defined in Eq. (1) using $\alpha = 0.2$.

4.2. Quantitative Results and Analysis

We evaluate TF-TimeCoVe on three VTG benchmarks and analyze the contribution of each pipeline component. First, we compare our method with prior zero-shot TF approaches and establish new state-of-the-art results (see Sec. 4.2.1). Afterwards we show that moving from single-pass generation to our two-stage CoVe framework improves performance. Highlighting the benefit of decoupled verification (Sec. 4.2.2). Finally, we ablate the two-pass verification techniques in Sec. 4.2.4.

4.2.1. State-of-the-Art Zero-Shot Comparison

We benchmark *TF-TimeCoVe* against current state-of-the-art zero-shot TF methods. As detailed in Tab. 2, *TF-TimeCoVe-32B* sets a new state-of-the-art for TF VTG. It achieves an R1@0.5 of 72.58% on QVH [12], 58.55% on Charades-STA [5], and 40.25% on ActivityNet Captions [10], showing absolute improvements of 10.90%, 7.53%, and 6.76% over existing baselines. Furthermore, while recent TF methods like VTimeCoT [32] and Chrono-GPT [16] are dependent on proprietary models (e.g., GPT-4o), we achieve stronger results using fully open-source ones. Notably, even the smaller *TF-TimeCoVe-8B* model surpasses all previously reported zero-shot methods, demonstrating the efficacy and scalability of our framework at lower parameter counts.

Method	# Stages	QVHighlights [†]				
		R1@.5	R1@.7	mAP	mAP@.5	mAP@.75
Chrono-Qwen	1	45.68	31.48	25.70	42.07	25.40
JointCoVeVTG	1	53.29	39.42	31.61	51.02	32.50
TF-TimeCoVe	2	65.94	50.06	39.69	58.59	41.82

Table 3. **Ablation of Chain-of-Verification (CoVe) strategies on QVHighlights using Qwen3-VL-8B.** The table compares Chrono-Qwen against single-pass and two-stage CoVe approaches.

While our focus is on TF methods, we note that TF-TimeCoVe approaches the performance of fine-tuned baselines. For instance, *TF-TimeCoVe-32B* achieves 72.58% R1@0.5 on QVH, narrowing the gap to Chrono-BLIP [16] (74.77%), a fully fine-tuned model. These results show that our method pushes open-source models to achieve competitive VTG performance, while remaining TF.

4.2.2. From Frames-Only to Chrono-Qwen to TF-TimeCoVe

As shown in Tab. 2, performance improves progressively from frames-only inputs to Chrono-Qwen and finally to TF-TimeCoVe. Frames-only models lack temporal structure, causing temporal drift and low recall. To address this, we apply the blueprint of Meinardus et al. [16] to Qwen3-VL. By interleaving timestamps with frames, we provide a discrete temporal axis that improves alignment between the query, frames, and predicted timestamps. Interestingly, the 8B variants of both the frames-only configuration and Chrono-Qwen outperform their 32B counterparts on Charades. This observation aligns with TimeLens [33], which shows that larger models are often more sensitive to noisy inputs. We hypothesize that Chrono’s reasoning-heavy prompting exacerbates this effect. Because VTG is primarily a temporal localization task rather than a complex reasoning problem, forcing explicit reasoning over noisy frames can distract the model and degrade localization accuracy, particularly in the 32B variant. By replacing Chrono’s reasoning-heavy prompting with a two-stage verification approach, TF-TimeCoVe-32B avoids this failure mode and achieves superior zero-shot performance across all three datasets.

4.2.3. From Single-Pass Generation to Two-Stage CoVe

We observe clear performance improvements when we replace the single-pass generation with our two-stage CoVe framework. As shown in Tab. 3, Chrono-Qwen achieves an R1@0.5 of 45.68% on QVH. Adding in-prompt verification in the single-pass *JointCoVeVTG* variant increases performance to 53.29% R1@0.5. The largest improvements are achieved once we decouple verification from candidate generation: *TF-TimeCoVe-8B* reaches 65.94% on R1@0.5, resulting in absolute improvements of **+12.65** over JointCoVeVTG-8B and **+20.26** over Chrono-Qwen-8B. These results show that JointCoVeVTG improves VTG, and separating drafting from verification further reduces confirmation bias and stabilizes temporal reasoning.

Finding 4: Applying CoVe even in a single-pass setting already provides a noticeable performance improvement.

4.2.4. Ablation of Verification-Stage Scoring Techniques

As we observe in Tab. 4, the lightweight signals *Confidence*, *Similarity*, and *their fusion* provide only limited verifying power. For

CoVe-Strategy	$\mathcal{T}$	QVHighlights [†]				
		R1@.5	R1@.7	mAP	mAP@.5	mAP@.75
1. Draft						
All Candidates	–	71.94	56.19	44.82	65.72	46.35
Oracle	–	<u>77.87</u>	<u>61.61</u>	<u>46.24</u>	<u>68.06</u>	<u>47.80</u>
2. Verify						
Confidence	0.5	70.84	54.77	43.38	63.72	45.25
Similarity	0.38	71.1	54.84	43.97	65.09	45.79
Confidence & Similarity	0	68.97	53.35	43.52	64.30	45.28
TNLL _{All}	0	71.94	56.19	44.82	65.72	46.35
VNL _{Same}	0	69.35	53.94	42.27	61.57	43.69
VNL _{Black}	0.13	72.58	57.10	44.35	64.16	46.14

Table 4. **Ablation of TF-TimeCoVe-32B verification strategies on QVHighlights.** *All Candidates* uses drafted windows without verification. *Oracle* shows the R1@0.5-based upper bound from all GT-aligned proposals. $\mathcal{T}$ is threshold. *All* means same results for both *Same* and *Black* types. Underlined: oracle best; bold: best verification. [†] indicates results are reported on the validation set.

all verification techniques we report the best-performing thresholds; however, a threshold of 0 is generally sufficient because negative scores are naturally filtered out. *Confidence based* verification is noisy because the model reports confidence based on language priors rather than visual temporal evidence, causing it to underperform the unverified baseline (*All Candidates*). *Similarity based verification* delivers comparatively better results, but it still fails to surpass the unverified baseline since cosine similarity in the text domain captures semantic overlap rather than VTG correctness. When we combine both signals, the noise from *confidence* overwhelms the weak corrective signal from *similarity*, resulting in the lowest R1@0.5 and R1@0.7 among the three.

The likelihood based signals provide a stronger contrastive mechanism overall. However, both *TNLL_{All}* and *VNL_{Same}* still fall below the drafted baseline, where *All* means same results for both *Same* and *Black* types. In case of *VNL_{Same}*, same-video negatives often contain visually similar content, producing too weak contrast. *TNLL_{All}* fails for a different reason. Textual likelihoods are dominated by language priors, so replacing the visual context, even with black frames, barely affects the model’s predicted text probabilities, which severely limits its discriminative power.

Only *VNL_{Black}* produces a clear improvement. By using black-frame negatives within the same embedding domain, we remove all visual structure from the comparison window and force the model to rely on genuine visual evidence from the candidate window. As a result, *VNL_{Black}* becomes the only verification strategy that outperforms the unverified baseline on R1@0.5 and R1@0.7. Although *VNL_{Black}* improves performance, it still lags well behind the *oracle upper bound*, with gaps of 5.29% at R1@0.5 and 4.51% at R1@0.7, indicating significant potential for further improvement.

4.3. Qualitative Analysis

In Fig. 3, we show quantitative results and demonstrate our TF-TimeCoVe’s ability to precisely align with ground truth (GT) boundaries (1), predict and verify multiple discontinuous windows (2), and avoid mislocalizations compared to zero-shot Chrono-GPT [16] (3). In (1), we use the query “Two men are looking at tools in Home Depot.” While Chrono-GPT predicts only a brief fraction of the event, our TF-TimeCoVe draft retrieves a broad candidate region that we successfully verify to match the GT. Here, we observe that our sampling shifts the pre-

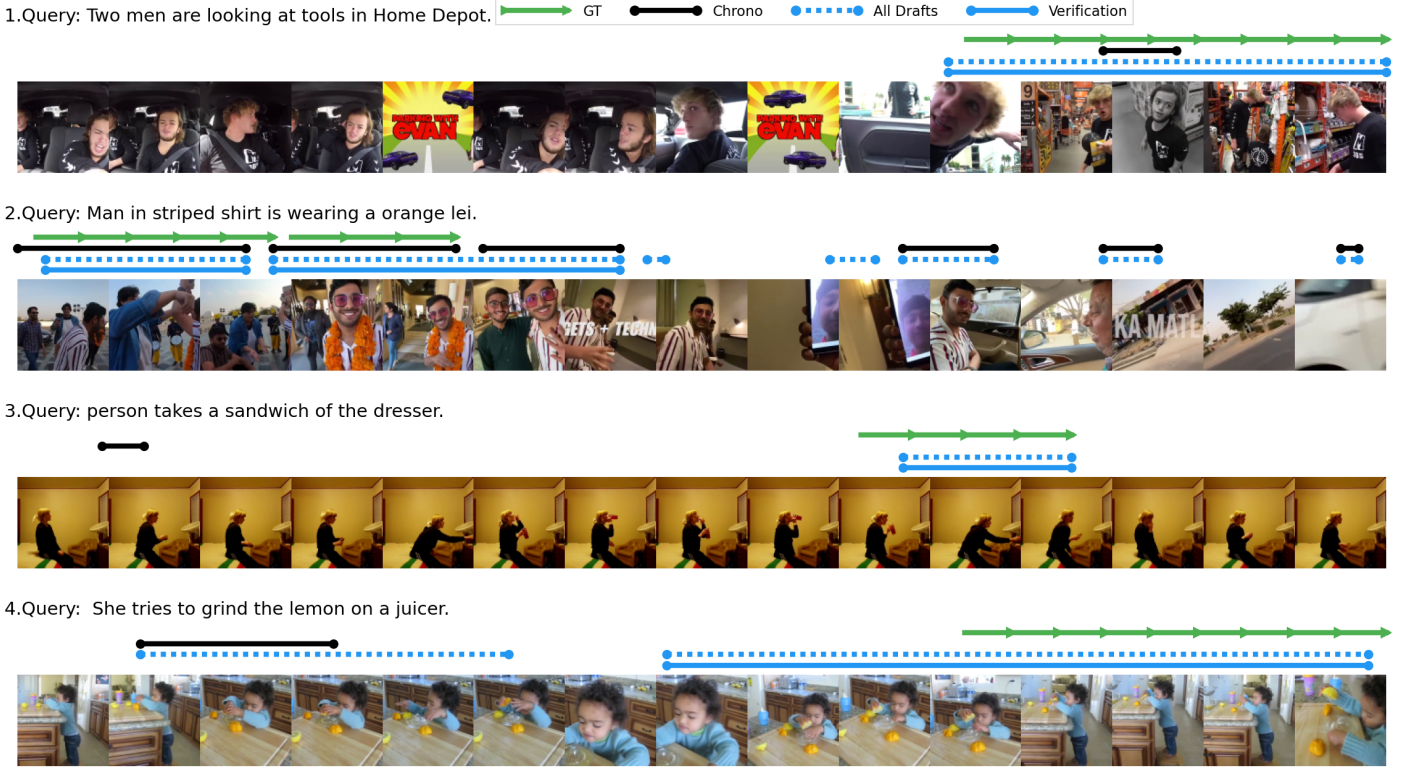


Figure 3. (1,3,4) correct single-window moment retrieval for TF-TimeCoVe. (2) correct multi-window moment retrieval for TF-TimeCoVe. (1,2) are from QVHighlights, (3) is from Charades-STA, (4) is from ActivityNet.

cise boundaries, but we improve the overall intersection significantly. Furthermore, in (3), we illustrate a scenario where the baseline fails completely. It predicts an incorrect segment at the start of the video, whereas our method correctly localizes the relevant action of the person taking a sandwich out of a dresser. We consider the frames of the video in Fig. 3 (3) to be very ambiguous, providing a difficult video for VTG. In (4), we show a scenario where our drafting step initially produces one accurate region and one incorrect region (according to GT), but through our verification process, we refine the final output to align with the GT.

5. Conclusions and Limitations

We present TF-TimeCoVe, the first framework to introduce Chain-of-Verification into zero-shot Video Temporal Grounding. Our novel approach reformulates localization as a draft-then-verify pipeline. By explicitly separating candidate generation from verification, we address two fundamental limitations of prior Training-Free methods: weak temporal awareness and reliance on unverified single-pass predictions. Our approach integrates timestamp-interleaved frames, sandwich prompting to mitigate long-context forgetting, and a Chain-of-Verification stage that evaluates candidate windows using confidence-based checks, semantic consistency, and contrastive likelihood ratios.

Across Charades-STA, ActivityNet Captions, and QVHighlights, we achieve new state-of-the-art performance among all Training-Free approaches, improving $R1@0.5$ by (+7.5%), (+9.2%) and (+10.9%), respectively. Additionally we close the performance gap to fine-tuned methods. Our *TF-TimeCoVe-32B* reaches 72.58% $R1@0.5$ on QVHighlights, approaching Chrono-

BLIP (74.77%). These results demonstrate that inference-time verification substantially improves zero-shot temporal grounding without fine-tuning, directly achieving our research objective.

Despite these improvements, two stage verification process increases inference cost. Furthermore, performance remains constrained by the reasoning capacity of frozen MLLMs. Verification thresholds and strategies also require further investigation and refinement. Identifying verification techniques and thresholding rules that generalize reliably across datasets remains an open challenge.

Overall, TF-TimeCoVe demonstrates that structured inference-time verification is a powerful and underexplored tool for improving zero-shot Video Temporal Grounding. We believe this work provides both practical improvements and conceptual guidance for future research on zero-shot Video Temporal Grounding.

As future work, we plan to fine-tune a lightweight verifier for the verification stage to improve efficiency and robustness while preserving the training-free nature of the overall pipeline. Additionally, we are curious how fine-tuning an open-weights model using our approach affects the results. Future work should also focus on moment retrieval for longer videos, a setting that remains largely unsolved. The central challenge lies in representing a very large number of frames within a single sequence without overwhelming the model’s context window or causing severe forgetting. This raises important questions about how to design scalable frame sampling strategies and how to preserve temporal coherence over long horizons.

References

- [1] Zechen Bai, Pichao Wang, Tianjun Xiao, Tong He, Zongbo Han, Zheng Zhang, and Mike Zheng Shou. Hallucination of Multimodal Large Language Models: A survey. In *International Conference on Learning Representations (ICLR)*, 2025. 2
- [2] Zhaowei Cai and Nuno Vasconcelos. Cascade R-CNN: Delving into high quality object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6154–6162, 2018. 4
- [3] Shehzaad Dhuliawala, Mojtaba Komeili, Jing Xu, Roberta Raileanu, Xian Li, Asli Celikyilmaz, and Jason Weston. Chain-of-Verification reduces hallucination in large language models. In *Findings of the association for computational linguistics: ACL 2024*, pages 3563–3578, 2024. 2, 3
- [4] Takumi Fukuzawa, Kensho Hara, Hirokatsu Kataoka, and Toru Tamaki. Can masking background and object reduce static bias for zero-shot action recognition? In *International Conference on Multimedia Modeling*, pages 366–379. Springer, 2025. 2
- [5] 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*, pages 5267–5275, 2017. 2, 6
- [6] Gemini Team and Google DeepMind. Introducing Gemini-3 pro. Model card, Google DeepMind, 2025. 1
- [7] Bin Huang, Xin Wang, Hong Chen, Zihan Song, and Wenwu Zhu. VTimeLLM: Empower llm to grasp video moments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14271–14280, 2024. 2, 6
- [8] De-An Huang, Shijia Liao, Subhashree Radhakrishnan, Hongxu Yin, Pavlo Molchanov, Zhiding Yu, and Jan Kautz. LITA: Language instructed temporal-localization assistant. In *Proceedings of the European conference on computer vision (ECCV)*, pages 202–218. Springer, 2024. 1
- [9] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55, 2025. 3
- [10] 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*, pages 706–715, 2017. 2, 6
- [11] Jin-Seop Lee, SungJoon Lee, Jaehan Ahn, YunSeok Choi, and Jee-Hyong Lee. TAG: A simple yet effective temporal-aware approach for zero-shot video temporal grounding. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2025. 2, 3, 6, 17
- [12] 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*, pages 11846–11858, 2021. 2, 3, 6
- [13] KunChang Li, Yanan He, Yi Wang, Yizhuo Li, Wenhui Wang, Ping Luo, Yali Wang, Limin Wang, and Yu Qiao. VideoChat: Chat-centric video understanding. *Science China Information Sciences*, 68(10):200102, 2025. 1
- [14] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. BSN: Boundary sensitive network for temporal action proposal generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 4
- [15] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranajpe, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024. 3, 11
- [16] Boris Meinardus, Hector G Rodriguez, Anil Batra, Anna Rohrbach, and Marcus Rohrbach. Chrono: A simple blueprint for representing time in MLLMs. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 4092–4097, 2025. 1, 2, 3, 6, 7, 11, 17
- [17] OpenAI. Gpt-4o system card. Technical report, OpenAI, 2024. 4
- [18] OpenAI. Introducing GPT-5, 2025. Accessed: 11.03.2026. 1
- [19] Nils Reimers and Iryna Gurevych. all-mpnet-base-v2: A sentence-transformers model. Technical report, UKP Lab, Technical University of Darmstadt, 2021. 5
- [20] Shuhuai Ren, Linli Yao, Shicheng Li, Xu Sun, and Lu 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*, pages 14313–14323, 2024. 2, 3, 6
- [21] Xiaowen Ruan, Zhaobo Qi, Yuanrong Xu, and Weigang Zhang. Dual-guided multi-modal bias removal strategy for temporal sentence grounding in video. *Multimedia Systems*, 31(2):108, 2025. 2, 4
- [22] Mattia Soldan, Mengmeng Xu, Sisi Qu, Jesper Tegner, and Bernard Ghanem. Vlg-net: Video-language graph matching network for video grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3224–3234, 2021. 6
- [23] Maya Varma, Jean-Benoit Delbrouck, Sophie Ostmeier, Akshay S Chaudhari, and Curtis Langlotz. Trove: Discovering error-inducing static feature biases in temporal Vision-Language Models. In *Advances in Neural Information Processing Systems*, 2025. 2
- [24] Qiyi Wang, Senda Chen, and Ying Shen. Causalvtg: Towards robust video temporal grounding via causal inference. In *Advances in Neural Information Processing Systems*, 2025. 2, 4
- [25] Yueqian Wang, Xiaojuan Meng, Jianxin Liang, Yuxuan Wang, Qun Liu, and Dongyan Zhao. Hawkeye: Training video-text LLMs for grounding text in videos. *arXiv preprint arXiv:2403.10228*, 2024. 2, 3, 6
- [26] Jianlong Wu, Wei Liu, Ye Liu, Meng Liu, Liqiang Nie, Zhouchen Lin, and Chang Wen Chen. A survey on video temporal grounding with Multimodal Large Language Model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025. 2
- [27] Yongliang Wu, Xinting Hu, Yuyang Sun, Yizhou Zhou, Wenbo Zhu, Fengyun Rao, Bernt Schiele, and Xu Yang. Number it: Temporal grounding videos like flipping manga. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13754–13765, 2025. 2, 3, 6, 11
- [28] Shen Yan, Xuehan Xiong, Arsha Nagrani, Anurag Arnab, Zhonghao Wang, Weina Ge, David Ross, and Cordelia Schmid. Unloc: A unified framework for video localization tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13623–13633, 2023. 6
- [29] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 1, 2, 6
- [30] Shoubin Yu, Jaemin Cho, Prateek Yadav, and Mohit Bansal. Self-chained image-language model for video localization and question answering. In *Advances in Neural Information Processing Systems*, pages 76749–76771, 2023. 2
- [31] Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Minghui Tan, and Chuang Gan. Dense regression network for video grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10287–10296, 2020. 6
- [32] Jinglei Zhang, Yuanfan Guo, Rolandos Alexandros Potamias, Jiankang Deng, Hang Xu, and Chao Ma. VTimeCot: Thinking by drawing for video temporal grounding and reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 24203–24213, 2025. 1, 2, 3, 6, 17

- [33] Jun Zhang, Teng Wang, Yuying Ge, Yixiao Ge, Xinhao Li, Ying Shan, and Limin Wang. TimeLens: Rethinking video temporal grounding with multimodal LLMs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026. [1](#), [2](#), [3](#), [6](#), [7](#), [11](#)
- [34] Minghang Zheng, Xinhao Cai, Qingchao Chen, Yuxin Peng, and Yang Liu. Training-free video temporal grounding using large-scale pre-trained models. In *Proceedings of the European conference on computer vision (ECCV)*, pages 20–37. Springer, 2024. [2](#), [3](#), [6](#)

TF-TimeCoVe: Zero-Shot Video Temporal Grounding via Chain of Verification

Supplementary Material

A. Additional Content

A.1. Explorative Approaches

Prompt	CoT	Charades-STA	
		R1@0.5	R1@0.7
Short	✗	46.83	21.08
Short	✓	29.27	13.09
Long	✗	46.72	20.99
Long	✓	27.85	12.04

Table 5. Ablation of Chain-of-Thought (CoT) on (✓)/off(✗) and Short vs Long. Qwen3-VL-8B is the backbone.

To better understand the source of improvements achieved by *TF-TimeCoVe*, we analyze how performance evolves from the original Chrono [16] baseline to our final *TF-TimeCoVe* approach.

A.1.1. Evolution of the Baseline: Prompting Foundations

Prompt Length. We first compare *Short* and *Long* prompt templates using Qwen3-VL-8B. Both depicted in Fig. 4 and Fig. 5 respectively. As shown in Table 5, shorter and more direct prompts outperform longer variants, suggesting that concise instructions help the model focus on temporal localization rather than processing unnecessary context, not relevant for the target.

Explorative Finding 1: Short prompt outperforms Long variants.

The CoT Penalty. Enabling Chain-of-Thought (CoT) reasoning is detrimental in this zero-shot setting. On Charades-STA, adding the instruction *Think step by step* causes R1@0.5 to drop from 46.83% to 29.27% as seen in Table 5. This indicates that reasoning step distracts the model from generating precise timestamps. This behavior is expected, as VTG is primarily a perception task [33] rather than a reasoning one.

Explorative Finding 2: CoT consistently harms VTG performance.

A.1.2. The Impact of Sandwiching and Visual Temporal Cues

Building on the prompting foundations discussed above, we further refine the prompt structure by evaluating two strategies. First, we explore *Sandwiching* [15], which first presents the query at the beginning and repeats it at after the interleaved frames of the prompt. This sandwiching helps to reinforce the query instruction. Second, we investigate *Visual Temporal Cues* (inspired by NumPro [27]), where frame indices are rendered directly onto the images to provide explicit temporal cues.

Sandwich Prompting. This modification proves to be a key improvement. By repeating the query at the end of the long video context, we mitigate the effect of long-context amnesia, where the model may lose focus on the initial query. As shown in Table 6,

this simple restructuring significantly improves temporal localization performance, boosting R1@0.5 on QVHighlights from 52.60% to 62.58%.

Explorative Finding 3: Sandwich prompting is essential for mitigating long-context forgetting in VTG.

Visual Temporal Cues. Inspired by Wu et al. [27], we examine imprinting frame numbers directly onto video frames. However, combining these visual numbers with interleaved textual timestamps introduces conflicting temporal signals and ultimately confuses the model rather than improving VTG. The mismatch arises because textual timestamps encode time, while imprinted numbers encode frame order. As shown in Table 6, configurations with imprinting consistently underperform their counterparts without imprinting.

Explorative Finding 4: Adding visual temporal cues on top of interleaved textual timestamps decreases the model’s temporal understanding.

B. An overview of the different prompts and modifications

User

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
 [Frame at t_F seconds: f_F]
 The video lasts $duration$ seconds.
 Query: $query$.
 Given the video and the query, find the relevant window.
 Relevant window: Answer only with the window in format [start, end], for example [5, 12]. Do not add any other text.

Assitant

Figure 4. **Short** prompt provided to Qwen3VL for single-window moment localization. Blue text represents variables.

Sandwich	Imprinting	QVHighlights [†]					Charades-STA		ActivityNet	
		R1@.5	R1@.7	mAP	mAP@.5	mAP@.75	R1@.5	R1@.7	R1@.5	R1@.7
X	X	52.60	36.60	29.90	47.80	30.20	46.83	21.08	31.21	19.29
X	✓	50.71	35.48	28.72	45.52	28.43	42.06	18.17	30.64	18.82
✓	X	62.58	46.97	37.44	57.06	38.87	48.63	20.05	33.52	20.30
✓	✓	53.48	37.55	31.73	50.08	31.52	36.88	14.09	27.51	15.95

(b) Sandwiching & Imprinting

Table 6. Ablation of Sandwich and Imprinting of visual timestamps on the frames are explored. ✓ highlights when the technique is used, while X shows the opposite. Qwen3-VL-8B is the backbone.

User

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
[Frame at t_F seconds: f_F]

The video lasts duration seconds.

Query: query .

Given the video and the query, find the relevant window.
The frames were sampled uniformly from the video.

If you think the event does not take place in the video, give your best guess, as otherwise the evaluation will be marked as incorrect. Never provide an empty list for <your answer>. The descriptions of moments are sometimes imprecise, so retrieve the closest moment. If you don't see an event remotely similar to the description, guess what is the most likely moment given the context. For instance, for cutting onion this could be between the time we see that the scene takes place in the kitchen and the time we see the onions being boiled in the pan.

The answer should be in the format of a list indicating the start and end of a window of moment, [start_window, end_window], for instance [0, 1]. If you detect multiple windows for the same moment, choose the most relevant one. It's important your final answer only contains one window. It's important that your answer is in this format, otherwise the evaluation will fail.

Output 'ANSWER: <your answer>' in the format specified above. For instance, for cutting onion this could be between the time we see that the scene takes place in the kitchen and the time we see the onions being boiled in the pan. It is very important that the answer is in this format, otherwise the evaluation will fail.

Assitant

User

Task: Given the video and query, find all relevant windows.

Query: " query ".

Video Stream (Duration: durations):

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
[Frame at t_F seconds: f_F]

You are given a video with multiple frames. The numbers after each video frame indicate its sampling timestamp (in seconds).

Please analyze the provided frames to find ALL occurrences of the query.

Query: " query ".

Prioritize Recall: It is better to include 2 extra seconds of context than to cut the action short(before and after the action, not just before).

If the action seems to start at 10s, start your window at 8s to catch the preparation.

The video might be noisy. If you see the action at 10s, 12s, and 15s, assume it is ONE continuous event from 10s to 15s. Do not break it into fragments.

If you find a small window (e.g., [7, 8]) that is already inside a larger window (e.g., [1, 33]), DISCARD the small one. Keep only the widest, most complete window.

Return ONLY the raw list of windows in the format: [[start, end], [start, end], ...].

Example: [[5, 15], [40, 65]]

Assitant

Figure 5. **Long** prompt provided to Qwen3VL for single-window moment localization. Blue text indicates dynamic variables injected at run-time.

Figure 6. TF-TimeCoVe **drafting** prompt with **sandwiching**, focusing on recall, provided to Qwen3VL for multi-window moment localization. Blue text indicates variables.

User

Evidence-Based Scoring Guidelines (0-100)

Use these three zones to determine your score. You may choose any integer within the appropriate range.

- Zone 1: NEGATIVE / CONTRADICTED (Score 0 - 49)
 - Definition: The visual evidence either contradicts the question (proving it false) or shows the subject is absent.
 - Guidance:
 - § 21-49 (Probably No): The condition appears absent, but there is a small element of confusion (e.g., A similar object is present, but likely not the correct one).
- Threshold: UNCERTAIN / INSUFFICIENT (Score 50)
 - Definition: The question cannot be confidently answered with "Yes" or "No" due to lack of evidence. The view might be fully occluded, the lighting too dark, or the camera cuts away at the exact moment of interest. It is a "toss-up."
- Zone 2: POSITIVE / SUPPORTED (Score 51 - 100)
 - Definition: The visual evidence supports the condition described in the question.
 - Guidance:
 - § 51-75 (Likely Yes): The condition is likely true, but the evidence is imperfect (e.g., distant view, brief glimpse, or partial occlusion).
 - § 76-100 (Definitely Yes): The condition is clearly visible and unambiguous. The visual evidence leaves no doubt.

Instruction: Evaluate the strength of the visual evidence. First, decide if the evidence leans Positive (>50) or Negative (<50). Then, assign a precise integer to reflect your certainty.

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...] [Frame at t_N seconds: f_N]

TASK: Action Verification

You are verifying a specific candidate window: [start to end]

Query: "query"

I have provided frames specifically from start to end below:

Answer these specific questions based on the visual evidence:

1. At start, question_1?
 - Visual Evidence: [Describe what is actually visible in the frames]
 - Answer: [Yes/No]
 - Confidence: [0-100]
2. Between start and end, question_2?
 - Visual Evidence: [Describe what is actually visible in the frames]
 - Answer: [Yes/No]
 - Confidence: [0-100]
3. At end, question_3?
 - Visual Evidence: [Describe what is actually visible in the frames]
 - Answer: [Yes/No]
 - Confidence: [0-100]

After answering, for each answer, provide the scores and confidences in a list.

- Format: [Answer number, Confidence score]
- Example: If Q1 is Yes (90%) and Q3 is No (20%), output: [[1, 90], [3, 20]]
- Constraint: At the end output the list.

Assitant

Figure 7. **Confidence-based verification** prompt provided to Qwen3VL during TF-TimeCoVe Confidence and Similarity verification. Blue text indicates variables, including window boundaries and context-specific verification questions. **Note: Timestamps, start, and end are relative to the original video length.**

User

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
[Frame at t_N seconds: f_N]
In five sentences, describe what happens in this video segment.

Assitant

Figure 8. Description prompt provided to Qwen3VL. This is then used for **Likelihood** methods. The model is asked to unconditionally describe the visual content of a drafted segment. Blue text indicates variables. **Note: Interleaved timestamps are relative to the original video length**

User

Task: Given the video and query, find the relevant window.

Query: "query".

Video Stream (Duration: durations):

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
[Frame at t_N seconds: f_N]

Instructions:

Based on the video above, find the start and end timestamps for ALL occurrences of the action of: "query".

Strictly check each candidate window against these Verification Questions: "questions".

You must catch ALL relevant moments. If the action happens multiple times, list every single occurrence.

Adjust the start/end seconds based on your answers to the questions above to maximize precision.

Output only the windows in format [[start, end], [start, end]], for example [[5, 12], [120, 125]].

Assitant

Figure 10. JointCoVe with pre-generated questions prompt for multi-window moment localization. Blue text indicates variables.

User

Task: Given the video and query, find the relevant window.

Query: "query".

Video Stream (Duration: durations):

[Frame at t_1 seconds: f_1] [Frame at t_2 seconds: f_2] [...]
[Frame at t_N seconds: f_N]

Instructions:

Based on the video above, find the candidate start and end timestamps for the action of "query".

Strictly check your candidate window against these Verification Questions: "questions".

Adjust the start/end seconds based on your answers to the questions above to maximize precision.

Output only one window in format [start, end], for example [5, 12].

Assitant

Figure 9. JointCoVeVTG with pre-generated questions prompt for single-window moment localization. Blue text indicates variables.

User

Query: A girl and her mother cook while talking over FaceTime.

Start check: "At start second, is the girl beginning to cook while initiating a FaceTime call with her mother?",

Core action check: "Between start and end seconds, are the girl and her mother actively cooking and conversing via FaceTime?",

End check: "At end second, have both the cooking activity and FaceTime conversation concluded?"

Assitant

Figure 11. Example of pregenerated questions derived from a query in QVHighlights. Blue text (Blue) indicates variable placeholders.

C. Additional Ablations

Table 7. Two-Stage CoVe ablation for Qwen3-VL-8B and 32B. All Candidates uses drafted windows directly; oracle selects multiple candidates based on GT. $\mathcal{T}$ is threshold.

CoVe-Strategy	Model	$\mathcal{T}$	QVHighlights [†]				
			R1@.5	R1@.7	mAP	mAP@.5	mAP@.75
1. Draft							
All Candidates	8B	–	66.39	50.45	40.33	59.92	42.26
	32B	–	71.94	56.19	44.82	65.72	46.35
Oracle	8B	–	71.48	54.84	41.56	61.98	43.52
	32B	–	77.87	61.61	46.24	68.06	47.8
2. Verify							
Confidence	8B	0.5	65.87	49.48	39.33	58.57	41.39
	32B	0.5	70.84	54.77	43.38	63.72	45.25
Similarity	8B	0	66.39	50.45	40.33	59.92	42.26
	32B	0.38	71.1	54.84	43.97	65.09	45.79
Confidence & Similarity	8B	0	65.29	49.61	39.74	59.56	41.73
	32B	0	68.97	53.35	43.52	64.30	45.28
Textual-LR (all types)	8B	0	66.39	50.45	40.33	59.92	42.26
	32B	0	71.94	56.19	44.82	65.72	46.35
Visual-LR (same)	8B	0	65.94	50.06	39.69	58.59	41.82
	32B	0	69.35	53.94	42.27	61.57	43.69
Visual-LR (black)	8B	0	65.94	50.06	39.69	58.59	41.82
	32B	0.13	72.58	57.10	44.35	64.16	46.14

Table 8. Two-Stage CoVe ablation for Qwen3-VL-8B and 32B on Charades-STA. All Candidates uses drafted windows directly; Oracle selects the most fitting moment with prior knowledge of GT.

CoVe-Strategy	Model	Charades-STA	
		R1@.5	R1@.7
1. Draft			
All Candidates	8B	56.61	29.95
	32B	58.87	30.19
Oracle	8B	57.15	30.11
	32B	59.62	30.40
2. Verify Stage			
Confidence	8B	56.16	29.38
	32B	58.55	30.46
Similarity	8B	54.97	28.79
	32B	57.69	29.87
Confidence & Similarity	8B	55.32	28.84
	32B	58.01	30.16
Textual-LR (same)	8B	55.4	29.46
	32B	57.58	29.54
Textual-LR (black)	8B	55.13	29.35
	32B	57.74	29.54
Visual-LR (same)	8B	55.32	29.33
	32B	57.45	29.41
Visual-LR (black)	8B	55.05	29.25
	32B	57.37	29.33

Table 9. Two-Stage CoVe ablation for Qwen3-VL-8B and 32B on ActivityNet. All Candidates uses drafted windows directly; Oracle selects the most fitting moment with prior knowledge of GT.

CoVe-Strategy	Model	ActivityNet	
		R1@.3	R1@.5
1. Draft Stage			
All Candidates	8B	34.98	20.84
	32B	40.72	24.68
Oracle	8B	37.38	21.71
	32B	43.84	25.85
2. Verify Stage			
Confidence	8B	34.26	18.47
	32B	40.24	24.27
Similarity	8B	31.95	19.15
	32B	37.70	22.90
Confidence & Similarity	8B	32.66	19.49
	32B	38.29	23.16
Textual-LR (same)	8B	32.75	19.64
	32B	38.18	23.20
Textual-LR (black)	8B	32.75	19.64
	32B	38.18	23.20
Visual-LR (same)	8B	32.26	19.44
	32B	38.01	23.24
Visual-LR (black)	8B	32.35	19.46
	32B	39.62	23.92

Method	QVHighlights		Charades-STA		ActivityNet	
	R1@.5 $\uparrow$	R1@.7 $\uparrow$	R1@.5 $\uparrow$	R1@.7 $\uparrow$	R1@.5 $\uparrow$	R1@.7 $\uparrow$
<i>Fine-Tuned</i>						
Chrono-BLIP [16]	75	61	69	49	54	36
<i>Zero-Shot</i>						
VTimeCoT (GPT4o) [32]	60	34	<u>51</u>	22	–	–
TAG [11]	–	–	49	<u>27</u>	29	15
Chrono-GPT [†] [16]	<u>62</u>	<u>42</u>	29	11	<u>31</u>	<u>18</u>
Qwen3-VL-32B (frames only) [†]	4	1	10	3	8	3
Chrono-Qwen3-VL-32B [†]	59	42	26	10	31	18
TF-TimeCoVe-32B [†]	74 ₊₁₂	57 ₊₁₅	59 ₊₈	30 ₊₃	40 ₊₉	24 ₊₆

Table 10. **Comparison to state-of-the-art (SOTA) methods in Moment Retrieval on QVHighlights, Charades-STA, and ActivityNet Captions.** Prior Training-Free (TF) SOTA is underlined, new TF SOTA is bold. Gains for TF-TimeCoVe are in comparison to prior TF SOTA. [†] indicates QVHighlights results are reported on the validation set.

Method	QVHighlights	Charades-STA	ActivityNet
	R1@.5 $\uparrow$	R1@.5 $\uparrow$	R1@.5 $\uparrow$
<i>Fine-Tuned</i>			
Chrono-BLIP [16]	75	69	54
<i>Zero-Shot</i>			
VTimeCoT (GPT4o) [32]	60	<u>51</u>	–
TAG [11]	–	49	29
Chrono-GPT [†] [16]	<u>62</u>	29	<u>31</u>
Qwen3-VL-32B (frames only) [†]	4	10	8
Chrono-Qwen3-VL-32B [†]	59	26	31
TF-TimeCoVe-32B [†]	74 ₊₁₂	59 ₊₈	40 ₊₉

Table 11. **Comparison to state-of-the-art (SOTA) methods in Moment Retrieval (R1@.5) on QVHighlights, Charades-STA, and ActivityNet Captions.** Prior Training-Free (TF) SOTA is underlined, new TF SOTA is bold. Gains for TF-TimeCoVe are in comparison to prior TF SOTA. [†] indicates QVHighlights results are reported on the validation set.