QUART: Query-Aware Context Routing for Streaming Video Understanding

Anonymous ACL submission

Abstract

Streaming video understanding requires a Video-LLM to watch an incoming video stream and answer user queries on the fly. Existing pipelines either pass every accumulated frame to the downstream Video-LLM or store them in a compressed memory. However, both do so without considering the query, so most of what the Video-LLM receives is unrelated to it. We propose QUART, a query-aware context routing framework that decides both when to respond and which frames to pass. A small router classifies each moment into SILENCE, SHORT, or LONG. For SHORT, only recent frames are passed. For LONG, segment-level captions built asynchronously in the background are used to retrieve query-relevant past segments. Since the router only selects which visual context to pass on, QUART can be applied to existing Video-LLMs without further training. QUART outperforms prior open-source streaming Video-LLMs on various streaming video understanding benchmarks, while passing 77% fewer frames to the downstream Video-LLM on average. The added latency is small enough not to be a bottleneck for real-time execution.

1 Introduction

Streaming video understanding (Chen et al., 2024a; Qian et al., 2025; Yao et al., 2025; Liu et al., 2024c) requires Video-LLMs to watch an incoming video stream and answer user queries on the fly. A streaming model needs not only to answer a query as soon as it arrives, but also to decide when to answer, since the right moment often comes only after the relevant event has been seen. Several works (Chen et al., 2024a; Li et al., 2025; Wang et al., 2025a; Xia et al., 2025) address this by having the Video-LLM predict special tokens, so that response timing is learned implicitly during generation. A related line of work instead adds a separate activation module that operates as a lightweight front-end (Qian et al., 2025; Wang et al., 2026). It predicts an activation signal at each frame to find the moment to respond, and a downstream Video-LLM is called only at that moment, consuming the accumulated frame cache to produce an answer.

In existing streaming pipelines, once the activation is decided, the Video-LLM receives every frame accumulated up to that point, regardless of what the user asked. However, most of these frames are not related to the question and act as noise, wasting the Video-LLM's limited context (Liu et al., 2024d; Sharma et al., 2024). We argue that the downstream Video-LLM should receive only the frames relevant to the query, not the full accumulated history.

Some streaming models address this by compressing the accumulated frames into a fixed-size memory (Zhang et al., 2025; Song et al., 2024; Yang et al., 2025c), rather than passing all raw frames to the downstream Video-LLM. The compression, however, is performed without considering the query, so the resulting memory still mixes the few frames that carry the answer with many that do not. The downstream Video-LLM therefore still receives a large amount of query-irrelevant visual information at every response.

In a streaming setting, this problem takes on a specific shape. Streaming queries are more often about the present than queries in general video understanding (Yang et al., 2026). For a question like *"What type of clothing is he wearing?"*, a few frames just before the activation are enough, but existing pipelines still pass the entire history. Even when the answer lies in the past, it lies in only part of the history. Long video understanding has addressed irrelevant-frame removal through frame retrieval and selection, where the full video is indexed in advance and only the query-relevant frames are kept. However, this assumption does not hold in streaming, because frames arrive over time and converting them into a retrievable form (e.g.,

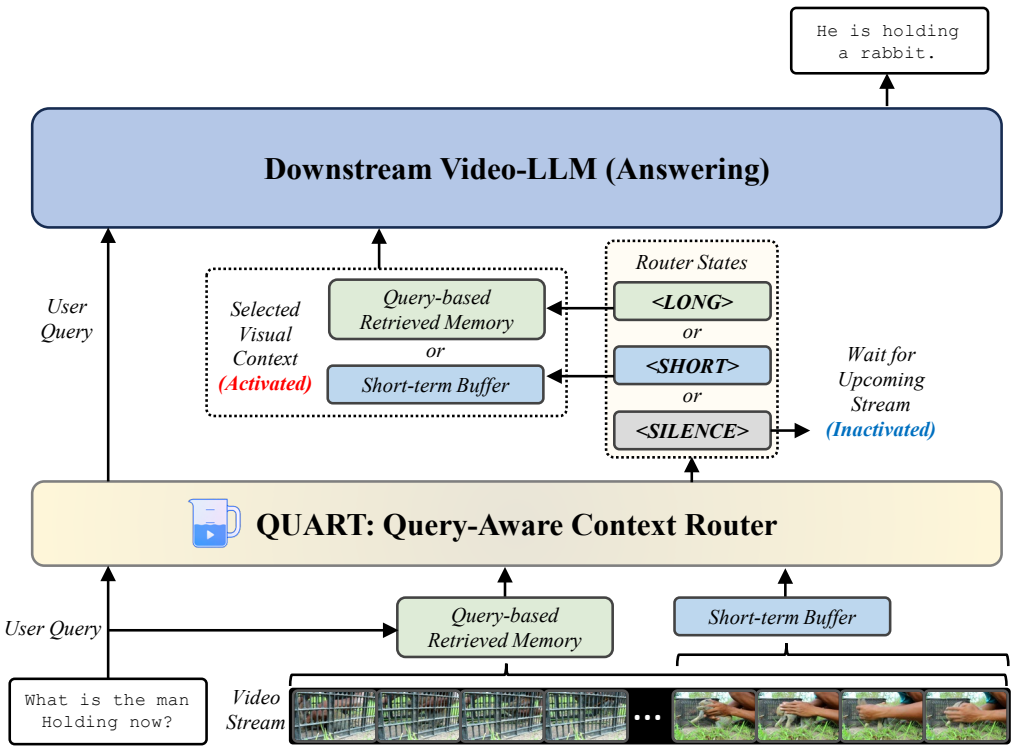

Figure 1: Overview of QUART framework.

text (Wang et al., 2025b) or embeddings (Tang et al., 2025)) is not instantaneous. For the past frames, the conversion can be done asynchronously in the background as the video plays, but the recent frames leave no time. A streaming framework that aims to remove irrelevant frames must therefore handle both cases.

Based on this observation, we propose a small router model that classifies each moment into one of three states, *SILENCE*, *SHORT*, and *LONG*. For *SHORT*, where the recent frames carry the answer, only those frames are passed to the downstream Video-LLM to remove unrelated past frames. Inspired by previous work (Zhang et al., 2026), we run the downstream Video-LLM in the background to generate a textual caption for each completed past segment, which serves as the retrieval index. At the response time, we use this index to select query-relevant past segments and pass their frames to the downstream Video-LLM.

Since the router only selects which visual context to pass on, our framework can be applied to existing Video-LLMs without further training. With this design, our method passes on average 77% fewer frames to the downstream Video-LLM than an offline setting, and successfully adapts the offline Video-LLM to a streaming paradigm, outperforming both prior streaming methods and its own offline baseline on streaming video understanding benchmarks (Table 1, 2). Our contributions are summarized as follows:

- We propose **QUART**, a **QU**ery-**A**ware context **R**ou**T**ing framework for streaming video understanding that helps the downstream Video-LLM focus on the visual context needed to answer the query. The router predicts one of three states, *SILENCE*, *SHORT* and *LONG*, to decide both when to respond and which temporal context to use.

- For the *LONG* state, we propose a retrieval-guided context selection mechanism that uses segment-level captions accumulated during playback as a retrieval index and passes visual context from query-relevant past segments to the downstream Video-LLM.

- We validate QUART through broad experiments on diverse streaming video understanding benchmarks, showing that it improves streaming video understanding in a plug-and-play manner without further training the downstream Video-LLM.

2 Related Work

2.1 Streaming Video Understanding

Streaming video understanding requires the model to respond while the video is still being received. Early work (Chen et al., 2024a) introduces the streaming EOS objective, where the Video-LLM stays silent at frames irrelevant to the query and produces a response only when the right moment arrives. Subsequent work has explored several directions, such as separating the decision of *when* to respond from the model that produces the response, and managing the visual memory under real-time constraints.

2.1.1 Proactive Decision Making

While early works (Chen et al., 2024a; Li et al., 2025; Wang et al., 2025a; Xia et al., 2025) train LLMs to predict special tokens to maintain silence, recent studies decouple the decision making process from downstream Video-LLMs. Compact activation models (Qian et al., 2025; Wang et al., 2026) are introduced to avoid optimization conflict between reasoning and decision making in a single Video-LLM. Some research instead builds the activation module from only the shallow Transformer layers of the Video-LLM itself (Ding et al., 2025). However, these modules only decide when to respond, so the downstream Video-LLM still receives many frames that the query does not need.

2.1.2 Visual Memory Compression for Streaming

Redundancy in neighboring video frames can lead to memory and computational overhead, which can hinder Video-LLMs from processing the video in real-time. Although early work actively uses a KV cache during streaming video processing, using all past frames still remains a bottleneck. Building on long video understanding methods (Jin et al., 2023; Shen et al., 2025; Yang et al., 2025b), one line of research compresses long-term visual memory by pruning similar and repetitive visual tokens (Yao et al., 2025). Inspired by the memory model of human cognition (Atkinson and Shiffrin, 1968), some methods not only compress long-term memory but also use high-resolution short-term memory (Song et al., 2024; Zhang et al., 2025; Huang et al., 2025; Xiong et al., 2025) to track ongoing events. KV cache optimizing methods (Yang et al., 2025c; Xu et al., 2026) for each Transformer decoder layer are also helpful to save memory and reduce computational overhead. Such memories, however, compress the visual stream itself, and do so without considering the query.

2.2 Context Retrieval and Selection

Another line of work reduces irrelevant visual context through retrieval or frame selection. In long video understanding, the full video is available before inference, so models can index the video in advance and select query-relevant frames or segments before answer generation (Wang et al., 2025b; Tang et al., 2025). This assumption does not directly hold in streaming video understanding, where future frames are unavailable and recently observed frames leave little time for indexing. Nevertheless, past observations can be converted into searchable representations while the video is being received. QUART uses this property by storing segment-level captions as a retrieval index for past visual context. This differs from visual memory compression methods in that the retrieved context is selected according to the query.

3 Methods

3.1 Problem Definition

We consider the streaming video question answering setting, where a model continuously receives video frames and answers user queries issued during the stream. Let $\mathcal{V} = \{v_1, v_2, \ldots\}$ denote a streaming video, where v_t is the incoming frame at time step t. Suppose that a user query q is issued at time step T_q. At any later time step $T \geq T_q$, the model has access only to the observed prefix $\mathcal{V}_{\leq T} = \{v_1, \ldots, v_T\}$. Unlike offline Video-LLMs that process the entire video at once, a streaming model must operate under partial observability.

Streaming video question answering also requires a proactive decision at every time step. After receiving q, the model repeatedly decides whether the current prefix contains sufficient evidence to answer the query. We denote this activation decision as

$$d_T = D(q, \mathcal{V}_{\leq T}) \in \{0, 1\},$$

where $d_T = 0$ means that the model remains silent, and $d_T = 1$ means that the model produces a response. The response time is the first activated time step,

$$T^\star = \min\{T \geq T_q : d_T = 1\}.$$

At $T^\star$, a conventional streaming pipeline generates a response using the observed prefix $\mathcal{V}_{\leq T^\star}$. How-

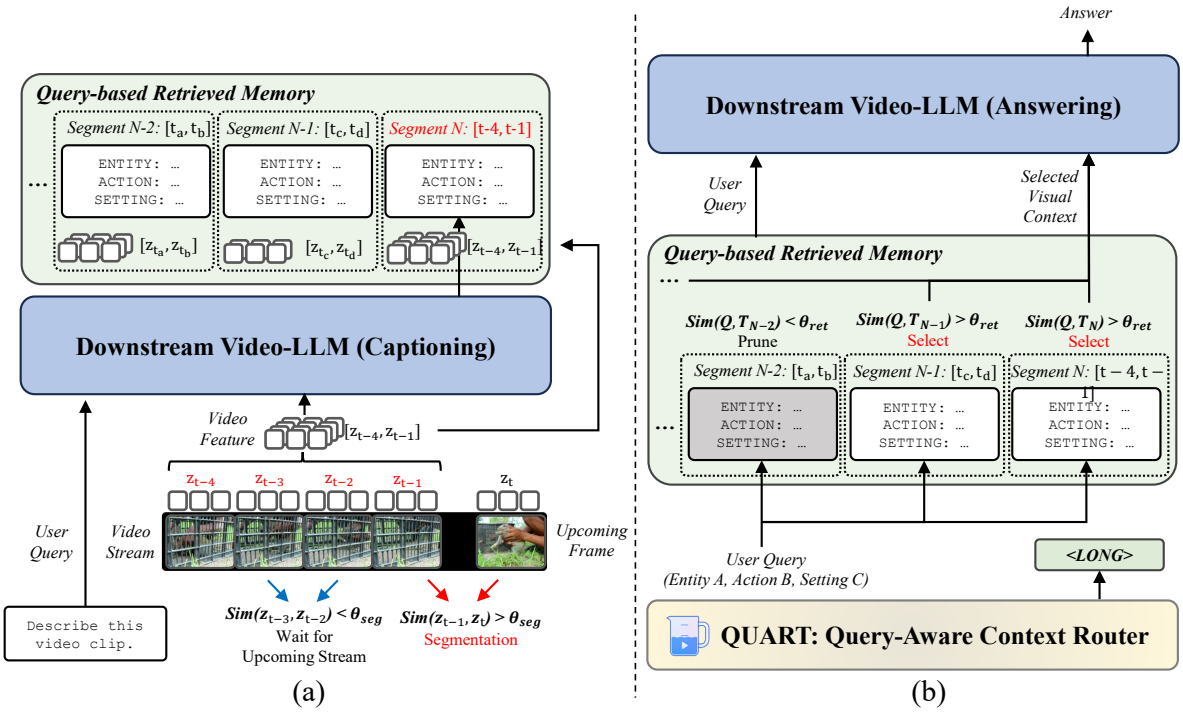

Figure 2: (a) shows how QUART merges video segments by calculating the similarity between adjacent video features. (b) shows how each segment is selected as visual context.

ever, the full prefix may contain query-irrelevant visual context that can distract the downstream Video-LLM. This motivates our problem of selecting the visual context to expose to the downstream Video-LLM at the response time.

To formalize this, we denote the latent visual evidence needed to answer q as $\mathcal{V}^*_{q,T^\star} \subseteq \mathcal{V}_{\leq T^\star}$. Frames outside $\mathcal{V}^*_{q,T^\star}$ are not directly required for answering q and may act as query-irrelevant visual context. Thus, passing the full observed prefix is not always desirable, since the relevant evidence can be mixed with query-irrelevant observations.

We formulate the problem as proactive visual context selection for streaming video question answering. At the response time $T^\star$, the system selects a context $C_{T^\star}$ from the observed prefix and generates an answer using a downstream Video-LLM M:

$$\hat{a} = M(q, C_{T^\star}), \qquad C_{T^\star} \subseteq \mathcal{V}_{\leq T^\star}.$$

The goal is to respond when the observed prefix contains sufficient evidence and to select $C_{T^\star}$ so that it approximates the latent evidence, $C_{T^\star} \approx \mathcal{V}^*_{q,T^\star}$, rather than simply using the full prefix $\mathcal{V}_{\leq T^\star}$. The following sections describe how QUART reduces query-irrelevant visual context while preserving the evidence needed to answer the query.

3.2 Query-aware Context Router

QUART separates the activation process from the subsequent answer decision. In this design, a lightweight router model first decides whether the downstream Video-LLM should be called, and the downstream Video-LLM generates the final answer only after activation. We extend this binary activation decision into a triplet routing decision. The original binary activation decision can be recovered from the router output by treating *SHORT* and *LONG* as activation states. Instead of only deciding whether to respond, the router also determines which temporal context should be used for answer generation. Let $\mathcal{W}_T = \mathcal{V}_{T-W:T}$ denote the recent visual window at time step T. Given the query q and $\mathcal{W}_T$, the router predicts

$$r_T = R(q, \mathcal{W}_T).$$

The output r_T is one of three states: *SILENCE*, *SHORT*, and *LONG*.

Silence. The *SILENCE* state means that the current prefix does not yet provide enough evidence to answer the query. The router therefore keeps the system waiting for future frames without calling the downstream Video-LLM.

Short. The *SHORT* state indicates that the query can be answered from the recent visual window. At

the response time $T^\star$, we set the selected context to the short window $C_{T^\star} = \mathcal{V}_{T^\star - w_s : T^\star}$, where w_s is the window length. This state covers present-oriented questions and questions whose answer becomes visible near the response time.

Long. The *LONG* state indicates that the model should respond, but the recent visual window does not contain the required evidence. Instead, the answer depends on an earlier part of the observed prefix. At the response time $T^\star$, we construct the selected context as

$$C_{T^\star} = \pi_{\text{long}}(q, \mathcal{I}_{\leq T^\star}, \mathcal{V}_{\leq T^\star}),$$

where $\mathcal{I}_{\leq T^\star}$ is the textual index accumulated during streaming. The policy π_{long} selects past frames using the query and this accumulated textual index. This state covers questions whose answer lies in the distant past rather than near the response time.

This formulation lets the router decide both whether to answer and where to look for the evidence. Equivalently, $d_T = 1$ when r_T is either *SHORT* or *LONG*, and $d_T = 0$ when r_T is *SILENCE*. Unlike other activation modules that mainly decide whether the downstream Video-LLM should be called, QUART also determines which temporal context should be used for answer generation.

3.3 Long Context Selection with Caption-Indexed Retrieval

As the video stream arrives, we incrementally divide the observed frames into temporal segments. For each incoming frame v_t, we compute its visual feature z_t by using the visual encoder of downstream Video-LLM and compare it with the feature of the previous frame z_{t-1}. A segment boundary is detected when the cosine similarity between consecutive frame features falls below a threshold:

$$\text{sim}(z_t, z_{t-1}) = \frac{z_t^\top z_{t-1}}{\|z_t\| \, \|z_{t-1}\|} < \theta_{\text{seg}},$$

where θ_{seg} is the segmentation threshold. To avoid overly long segments, we impose a maximum segment length $L_{\max}$. If no boundary is detected before the current segment reaches $L_{\max}$ frames, the segment is closed at $L_{\max}$.

Let $\mathcal{S}_{\leq T} = \{s_i\}_{i=1}^{N_T}$ denote the set of completed segments up to time T, where each segment s_i corresponds to a frame interval $\mathcal{V}_{[a_i, b_i]} \subseteq \mathcal{V}_{\leq T}$. Only completed segments are included in this set, so each segment satisfies $b_i \leq T$. Once a segment is completed, the downstream Video-LLM generates a textual description c_i for that segment and stores it together with its temporal span $[a_i, b_i]$. This gives the accumulated textual index

$$\mathcal{I}_{\leq T} = \{(c_i, [a_i, b_i])\}_{i=1}^{N_T}.$$

When the router outputs *LONG* at the response time $T^\star$, we use the query q to search the accumulated textual index $\mathcal{I}_{\leq T^\star}$. For each indexed segment, we compute the cosine similarity between the query and the segment description $\alpha_i = \text{sim}\big(E(q), E(c_i)\big)$, where $E(\cdot)$ is a text encoder. Segments with similarity above a threshold are selected as relevant past segments:

$$\mathcal{R}_{T^\star} = \{s_i \in \mathcal{S}_{\leq T^\star} : \alpha_i \geq \theta_{\text{ret}}\},$$

where θ_{ret} is the retrieval threshold. This lets the system find relevant past segments using compact textual descriptions, rather than searching over all past frames directly.

After retrieving the relevant segments, π_{long} gathers their original frames from the observed prefix. The selected visual context is written as

$$C_{T^\star} = \pi_{\text{long}}(q, \mathcal{I}_{\leq T^\star}, \mathcal{V}_{\leq T^\star}), \qquad C_{T^\star} \subseteq \mathcal{V}_{\leq T^\star}.$$

The captions are used only to locate relevant past segments. They are not passed to the downstream Video-LLM as answer-time textual hints. The downstream Video-LLM receives the query and the selected original frames:

$$\hat{a} = M(q, C_{T^\star}).$$

All indexing steps are performed asynchronously while the video stream is received, and this process does not require additional training of the downstream Video-LLM.

3.4 Training Objective

We train the context router with token-level supervision. For each labeled temporal position, the router is trained to predict one of the three router tokens corresponding to *SILENCE*, *SHORT*, and *LONG*. The loss is applied only to these router-token positions, while other text, visual, and padding tokens are masked out.

Since most streaming positions do not contain enough evidence to answer the query, the labels are dominated by *SILENCE*. We therefore use a

focal cross-entropy loss (Lin et al., 2017) to prevent the objective from being dominated by silence positions:

$$\mathcal{L}_{\text{router}} = \frac{1}{|\Omega|} \sum_{i \in \Omega} \alpha_{c_i} (1 - p_i)^{\gamma} \left[-\log p_i \right],$$

where Ω is the set of router-token positions, c_i is the target router token at position i, p_i is the probability assigned to c_i, γ is the focal loss exponent, and α_{c_i} is a class balancing weight.

3.5 Inference Setting

Before a user query arrives, the system processes the incoming video stream incrementally. As each frame arrives, visual features are computed and cached. Before the query is given, the observed stream is continuously segmented, and the downstream Video-LLM generates captions to prepare for a possible *LONG* state. These captions are stored in the textual index and later used for retrieval.

After the user query arrives, the router is evaluated at each time step and predicts one of *SILENCE*, *SHORT*, and *LONG*. Since the training labels are dominated by *SILENCE*, direct argmax decoding often leads the router to predict *SILENCE*. We therefore introduce an activation threshold τ. Let $p_0 = p_T(\textit{SILENCE})$, $p_1 = p_T(\textit{SHORT})$, and $p_2 = p_T(\textit{LONG})$. The decoded state is defined as follows:

$$r_T = \begin{cases} \textit{SHORT} & \text{if } p_1 \geq \max(p_2, \tau) \\ \textit{LONG} & \text{if } p_2 > \max(p_1, \tau) \\ \textit{SILENCE} & \text{otherwise.} \end{cases}$$

If either *SHORT* or *LONG* has probability above τ, we suppress the *SILENCE* logit and choose the more likely state between *SHORT* and *LONG*.

If the router predicts the *SILENCE* state, video streaming continues, while segmentation continues and the downstream Video-LLM generates captions whenever a segment is completed. If the router predicts the *SHORT* state, the short context of length w_s is selected and passed to the downstream Video-LLM together with the query to generate the answer. If the router predicts the *LONG* state, retrieval is performed over the captions of the visual segments observed up to that time. The long context is selected from the retrieved segments, and its original frames are passed to the downstream Video-LLM.

4 Experiments

4.1 Training

Dataset. For router training, we curate a diverse collection of temporally annotated video datasets covering temporally localized event descriptions (Krishna et al., 2017; Regneri et al., 2013; Liu et al., 2024e), action and activity annotations (Gao et al., 2017; Sigurdsson et al., 2018), instructional procedures (Zhou et al., 2018; Tang et al., 2019), and moment-level grounding or video QA (Anne Hendricks et al., 2017; Lei et al., 2018). In total, the resulting training set contains 111,822 rows from 26,000 videos and 53,583 source events or questions. The detailed description of the dataset is in Appendix A.1.

Implementation Details. We initialize the router from Qwen3-VL-2B-Instruct (Bai et al., 2025) and add three router tokens, $\langle$router_silence$\rangle$, $\langle$router_short$\rangle$, and $\langle$router_long$\rangle$. Instead of adding a separate classifier head, we train the original LM head to output one of these router tokens. Each training row corresponds to a streaming state rather than the full video. Each row specifies a stream end time and a training window, and the model input includes only the frames within that window sampled at 1 FPS. By varying the stream end time and the window length, the router observes different streaming states, including windows where the target evidence is visible, windows where it has moved into the past, and windows where no activation should occur. These deactivated windows help improve activation precision, instead of encouraging the router to fire whenever the query is related to the video. We use up to 256 frames per row and set both the minimum and maximum visual pixels to 50,176. Since Qwen3-VL merges every two adjacent frames during video tokenization, we convert frame-level labels into group-level labels accordingly. For each merged group, we use the label of the last frame as the group label. We train the model for one epoch on 111,822 streaming video-query rows. We use AdamW with a learning rate of 1×10^{-5}, cosine decay, 5% warmup, zero weight decay, gradient clipping at 1.0, and bf16 precision. Training is performed on 8 NVIDIA RTX 3090 GPUs with per-device batch size 1 and gradient accumulation of 8, giving an effective batch size of 64.

Table 1: Evaluation results on OVO-Bench (Niu et al., 2025). ‡ denotes the downstream Video-LLM of QUART, and the benchmark was performed by ourselves.

Method	# of Frames	Real-Time Visual Perception							Backward Tracing				Forward Act. Responding				Overall
		OCR	ACR	ATR	STU	FPD	OJR	**Avg.**	EPM	ASI	HLD	**Avg.**	REC	SSR	CRR	**Avg.**	**Avg.**
Human																	
Human	-	93.96	92.57	94.83	92.70	91.09	94.02	93.20	92.59	93.02	91.37	92.33	95.48	89.67	93.56	92.90	92.81
Open-source Models (Offline)																	
LLaVA-OneVision-7B (Li et al., 2024)	64	66.44	57.80	73.28	53.37	71.29	61.96	64.02	54.21	55.41	21.51	43.71	25.64	67.09	58.75	50.50	52.74
LLaVA-Next-Video-7B (Liu et al., 2024a)	64	69.80	59.60	66.40	50.60	72.30	61.40	63.30	51.20	64.20	9.70	41.70	34.10	67.60	60.80	54.20	53.10
Qwen2-VL-7B (Wang et al., 2024)	64	60.40	50.46	56.03	47.19	66.34	55.43	55.98	47.81	35.48	56.08	46.46	31.66	65.82	48.75	48.74	50.39
Qwen3-VL-8B‡ (Bai et al., 2025)	1 FPS	77.18	60.55	77.59	58.43	70.30	65.22	68.21	53.20	58.11	27.96	46.42	24.95	51.34	44.58	40.29	51.64
InternVL-V2-8B (Chen et al., 2024b)	64	67.11	60.55	63.79	46.07	68.32	56.52	60.39	48.15	57.43	24.73	43.44	26.50	59.14	54.14	46.60	50.15
LongVU-7B (Shen et al., 2025)	1 FPS	55.70	49.50	59.50	48.30	68.30	63.00	57.40	43.10	66.20	9.10	39.50	16.60	69.00	60.00	48.50	48.50
Open-Source Models (Streaming)																	
Flash-VStream (Zhang et al., 2025)	1 FPS	24.16	29.36	28.45	33.71	25.74	28.80	28.37	39.06	37.16	5.91	27.38	8.02	67.25	60.00	45.09	33.61
VideoLLM-Online (Chen et al., 2024a)	2 FPS	8.05	23.85	12.07	14.04	45.54	21.20	20.79	22.22	18.80	12.18	17.73	-	-	-	-	-
Dispider (Qian et al., 2025)	1 FPS	57.72	49.54	62.07	44.94	61.39	51.63	54.55	48.48	55.41	4.30	36.06	18.05	37.36	48.75	34.72	41.78
TimeChat-Online (Yao et al., 2025)	1 FPS	69.80	48.60	64.70	44.90	68.30	55.40	58.60	53.90	62.80	9.10	42.00	32.50	36.50	40.00	36.40	45.60
StreamAgent-7B (Yang et al., 2025a)	1 FPS	71.20	53.20	63.60	53.90	67.30	58.70	61.30	54.80	58.10	25.80	41.70	35.90	48.40	52.00	45.40	49.40
StreamBridge (Wang et al., 2026)	1 FPS	84.56	71.56	74.14	49.44	75.25	72.83	71.30	-	-	-	-	-	-	-	-	-
QUART (Ours)	1 FPS	89.26	74.31	78.45	66.29	70.30	72.83	**75.24**	54.88	71.62	33.33	**53.28**	45.55	60.09	56.25	**53.96**	**60.83**

Table 2: Evaluation results on StreamingBench (Lin et al., 2026). ‡ denotes the downstream Video-LLM of QUART, and the benchmark was performed by ourselves. The overall average score is the average of real-time visual understanding score and contextual understanding score.

Method	# of Frames	Real-Time Visual Understanding											Contextual Understanding					Overall
		OP	CR	CS	ATP	EU	TR	PR	SU	ACP	CT	**Avg.**	ACU	MCU	SQA	PO	**Avg.**	**Avg.**
Human																		
Human	-	89.47	92.00	93.60	91.47	95.65	92.52	88.00	88.75	89.74	91.30	91.46	88.80	90.40	95.00	100	93.55	92.51
Open-source Models (Offline)																		
LLaVA-OneVision-7B (Li et al., 2024)	32	80.38	74.22	76.03	80.72	72.67	71.65	67.59	65.45	65.72	45.08	71.12	35.60	36.00	27.27	29.55	32.74	51.93
Qwen3-VL-8B‡ (Bai et al., 2025)	1 FPS	78.32	75.00	78.55	84.62	70.44	80.25	83.33	71.54	70.45	51.06	75.66	36.00	39.20	47.20	22.80	39.68	57.67
Qwen2-VL-7B (Wang et al., 2024)	1 FPS	75.20	82.81	73.19	77.45	68.32	71.03	72.22	61.19	61.47	46.11	69.04	31.20	26.00	39.60	22.73	31.66	50.35
MiniCPM-V 2.6 8B (Yao et al., 2024)	32	71.93	71.09	77.92	75.82	64.60	65.73	70.37	56.10	62.32	53.37	67.44	34.00	31.60	41.92	22.22	34.97	51.21
InternVL-V2-8B (Chen et al., 2024b)	16	68.12	60.94	69.40	77.12	67.70	62.93	59.26	53.25	54.96	56.48	63.72	32.00	31.20	32.32	40.91	32.42	48.07
Kangaroo-7B (Liu et al., 2024b)	64	71.12	84.38	70.66	73.20	67.08	61.68	56.48	55.69	62.04	38.86	64.60	32.80	26.40	33.84	16.00	30.06	47.33
LongVA-7B (Zhang et al., 2024)	128	70.03	63.28	61.20	70.92	62.73	59.50	61.11	53.66	54.67	34.72	59.96	32.80	29.60	30.30	15.91	29.95	44.96
VILA-1.5-8B (Lin et al., 2024)	14	53.68	49.22	70.98	56.86	53.42	53.89	54.63	48.78	50.14	17.62	52.32	26.80	34.00	23.23	17.65	27.35	39.84
Video-LLaMA2-7B (Cheng et al., 2024)	32	55.86	55.47	57.41	58.17	52.80	43.61	39.81	42.68	45.61	35.23	49.52	24.80	26.80	18.67	0.00	21.93	35.73
Open-source Models (Streaming)																		
Flash-VStream-7B (Zhang et al., 2025)	1 FPS	25.89	43.57	24.91	23.87	27.33	13.08	18.52	25.20	23.87	48.70	23.23	24.80	25.20	26.80	1.96	24.12	23.68
VideoLLM-Online-8B (Chen et al., 2024a)	2 FPS	39.07	40.06	34.49	31.05	45.54	32.40	31.48	34.16	42.49	27.89	35.99	24.19	29.20	30.80	3.92	26.55	31.27
Dispider (Qian et al., 2025)	1 FPS	74.92	75.53	74.10	73.08	74.44	59.92	76.14	62.91	62.16	45.80	67.63	39.62	27.65	34.80	25.34	33.61	50.62
TimeChat-Online-7B (Yao et al., 2025)	1 FPS	80.80	79.70	80.80	83.30	74.80	78.80	78.70	64.20	68.80	58.00	75.28	-	-	-	-	-	-
StreamAgent-7B (Yang et al., 2025a)	1 FPS	79.63	78.31	79.28	75.87	74.74	76.92	82.94	66.31	73.69	55.40	74.31	39.72	30.25	39.60	28.90	34.62	54.47
StreamBridge (Wang et al., 2026)	1 FPS	84.74	82.68	88.92	89.77	77.36	85.36	84.26	69.92	71.67	35.75	77.04	-	-	-	-	-	-
QUART (Ours)	1 FPS	77.78	73.44	87.07	88.78	72.96	84.42	78.70	72.36	74.43	53.19	**77.43**	34.80	36.80	43.60	46.00	**38.88**	**58.15**

4.2 Evaluation Benchmarks

We evaluate our framework on OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026) as timestamped streaming video question-answering benchmarks. Since the omni-source understanding task of StreamingBench requires audio input, we exclude the task to compare only with Video-LLMs. In both benchmarks, each query is associated with a time point, and the model is evaluated using only the video prefix observed up to that time. We use these benchmarks to evaluate whether selecting different visual contexts for different query states improves answer accuracy under causal prefix constraints.

Table 3: Evaluation results on OVO-Bench and StreamingBench across different subsets. Removing the context router reverts the framework to an offline setting, which cannot respond proactively.

Context Router	Segment Retrieval	OVO-Bench				StreamingBench		
		RT	BW	FW	Avg.	RT	CTX	Avg.
✗	✗	68.21	46.42	40.29	51.64	75.66	39.68	57.67
✓	✗	74.56	46.46	48.35	56.46	77.14	38.75	57.95
✓	✓	74.76	52.11	55.21	**60.69**	77.43	38.88	**58.15**

We use the same evaluation setup for both benchmarks. For each query, the router first predicts one of *SILENCE*, *SHORT*, and *LONG* from an 8-second recent window sampled at 1 FPS. If it predicts *SHORT*, the downstream model receives the recent 4 frames. If it predicts *LONG*, the caption-indexed branch combines 4 recent frames with up to 60 frames sampled from retrieved past segments. Frames are sampled from the retrieved segments in proportion to their retrieval similarity, so higher-scoring segments contribute more frames. This results in at most 64 image frames for the downstream model.

5 Results

5.1 Main Results

To evaluate the effectiveness of our proposed method in modeling streaming video understanding, we compare QUART against various streaming Video-LLM baselines. Table 1 and 2 present results on OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026), respectively. As a result, QUART outperforms prior open-source streaming Video-LLMs, confirming that routing

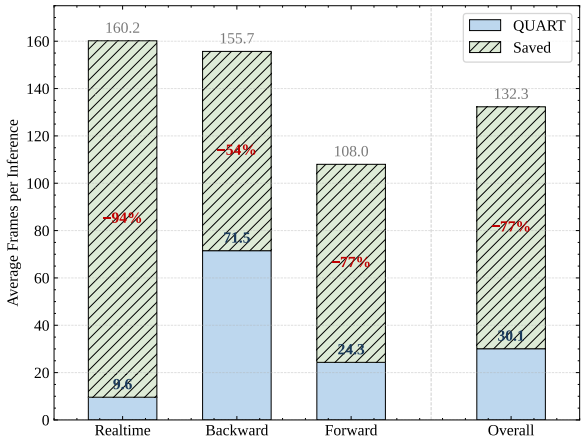

Figure 3: Average number of frames passed to the downstream Video-LLM per inference on OVO-Bench, including calls used for background segment captioning. QUART removes around 77% of the frames that would otherwise act as noise.

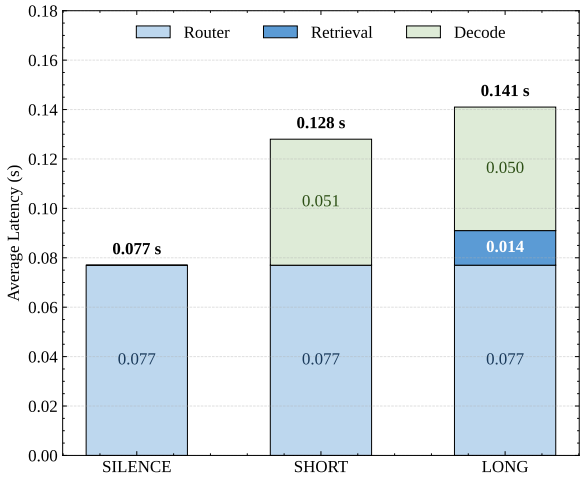

Figure 4: Inference time for each router state. The router and retrieval are not a bottleneck for real-time use.

only the query-relevant context to the downstream Video-LLM improves streaming video understanding. QUART also outperforms its own downstream Video-LLM, Qwen3-VL-8B on average, supporting prior findings (Wang et al., 2025b; Tang et al., 2025) that selecting query-relevant frames can improve an existing Video-LLM without further training.

5.2 Ablation Test

To verify that each component of QUART is necessary, we ablate the router and the retrieval module independently. Table 3 reports how OVO-Bench and StreamingBench performance changes when each is removed. Applying the router improves real-time performance on both benchmarks. The router turns the offline Video-LLM into a streaming one, and it also correctly separates the moments that need only recent frames from those that need past context. Applying retrieval then improves performance across the board. This shows that removing query-irrelevant frames helps the Video-LLM on its own, and this gain is separate from the router's contribution.

5.3 Inference Efficiency

To evaluate frame efficiency of downstream Video-LLM, we compare QUART and the offline setting in terms of the average number of frames the downstream Video-LLM has to process on OVO-Bench. Figure 3 shows that QUART removes around 77% of the input frames on average, even as real-time performance improves. This is because real-time queries can usually be answered from the few frames just before the activation, exactly what the *SHORT* branch passes.

We also measure the average inference time of QUART on OVO-Bench, including the time for each component. Segment captions and the KV cache are computed in the background while the video plays, and we assume they are ready by the time a response is needed. Figure 4 shows that the time from query arrival to answer generation is 0.077 seconds for *SILENCE*, 0.128 seconds for *SHORT*, and 0.141 seconds for *LONG*. The router takes 0.077 seconds in every branch, and retrieval adds 0.014 seconds in *LONG*. Both are small enough not to be a bottleneck for real-time use.

6 Conclusion

Streaming Video-LLMs have learned to decide *when* to respond, but the context they receive at that moment is still shaped by the stream rather than by the query. We present **QUART**, which makes this context selection query-aware under streaming constraints: a single router emits *SILENCE*, *SHORT*, or *LONG* to decide both when to respond and which frames to expose, with the *LONG* case resolved through captions built asynchronously during playback. Our experiments suggest that query-aware context selection, long studied in the offline setting, can be brought into the streaming regime without sacrificing real-time response, exposing the downstream Video-LLM to a smaller and more focused set of frames.

Limitations

QUART has several limitations that we leave to future work. First, the *LONG* branch relies on segment-level captions as the retrieval index, so visual details that the captioner omits, such as fine-grained appearance or short-lived motion, cannot be recovered by text-based retrieval; a hybrid index that combines captions with visual embeddings is a natural next step. Second, our pipeline assumes that captions for completed segments are ready by response time. This holds in our experiments because captioning runs asynchronously during playback, but the assumption may break under heavy load or on devices where the captioner cannot keep up with the stream. Third, our retrieval-based *LONG* branch concentrates frames on segments most similar to the query, which suits questions anchored to specific past moments but underperforms on reasoning queries that need uniform coverage of the stream, such as causal or prospective reasoning. Although uniform sampling is in principle compatible with our policy, similarity scores in practice cluster around a few segments. Adding router states tailored to such queries or replacing flat captions with chain-of-thought captions that encode cross-segment reasoning are promising directions.

References

Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. 2017. Localizing moments in video with natural language. In *Proceedings of the IEEE international conference on computer vision*, pages 5803–5812.

Richard C Atkinson and Richard M Shiffrin. 1968. Human memory: A proposed system and its control processes. In *Psychology of learning and motivation*, volume 2, pages 89–195. Elsevier.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, and 1 others. 2025. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*.

Joya Chen, Zhaoyang Lv, Shiwei Wu, Kevin Qinghong Lin, Chenan Song, Difei Gao, Jia-Wei Liu, Ziteng Gao, Dongxing Mao, and Mike Zheng Shou. 2024a. Videollm-online: Online video large language model for streaming video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18407–18418.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, and 1 others. 2024b. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 24185–24198.

Zesen Cheng, Sicong Leng, Hang Zhang, Yifei Xin, Xin Li, Guanzheng Chen, Yongxin Zhu, Wenqi Zhang, Ziyang Luo, Deli Zhao, and Lidong Bing. 2024. Videollama 2: Advancing spatial-temporal modeling and audio understanding in video-llms. *arXiv preprint arXiv:2406.07476*.

Xin Ding, Hao Wu, Yifan Yang, Shiqi Jiang, Qianxi Zhang, Donglin Bai, Zhibo Chen, and Ting Cao. 2025. Streammind: Unlocking full frame rate streaming video dialogue through event-gated cognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13448–13459.

Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. 2017. Tall: Temporal activity localization via language query. In *Proceedings of the IEEE international conference on computer vision*, pages 5267–5275.

Zhenpeng Huang, Xinhao Li, Jiaqi Li, Jing Wang, Xiangyu Zeng, Cheng Liang, Tao Wu, Xi Chen, Liang Li, and Limin Wang. 2025. Online video understanding: Ovbench and videochat-online. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 3328–3338.

Peng Jin, Ryuichi Takanobu, Caiwan Zhang, Xiaochun Cao, and Li Yuan. 2023. Chat-univi: Unified visual representation empowers large language models with image and video understanding. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13700–13710.

Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-captioning events in videos. In *Proceedings of the IEEE international conference on computer vision*, pages 706–715.

Jie Lei, Licheng Yu, Mohit Bansal, and Tamara Berg. 2018. Tvqa: Localized, compositional video question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 1369–1379.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and 1 others. 2024. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Wei Li, Bing Hu, Rui Shao, Leyang Shen, and Liqiang Nie. 2025. Lion-fs: Fast & slow video-language thinker as online video assistant. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3240–3251.

Ji Lin, Hongxu Yin, Wei Ping, Pavlo Molchanov, Mohammad Shoeybi, and Song Han. 2024. Vila: On

pre-training for visual language models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 26689–26699.

Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. 2026. Streamingbench: Assessing the gap for mllms to achieve streaming video understanding. In *ICASSP 2026-2026 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 12147–12151. IEEE.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2017. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. 2024a. Llava-next: Improved reasoning, ocr, and world knowledge.

Jiajun Liu, Yibing Wang, Hanghang Ma, Xiaoping Wu, Xiaoqi Ma, xiaoming Wei, Jianbin Jiao, Enhua Wu, and Jie Hu. 2024b. Kangaroo: A powerful video-language model supporting long-context video input. *arXiv preprint arXiv:2408.15542*.

Jihao Liu, Zhiding Yu, Shiyi Lan, Shihao Wang, Rongyao Fang, Jan Kautz, Hongsheng Li, and Jose M Alvare. 2024c. Streamchat: Chatting with streaming video. *arXiv preprint arXiv:2412.08646*.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024d. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173.

Ye Liu, Zongyang Ma, Zhongang Qi, Yang Wu, Ying Shan, and Chang W Chen. 2024e. Et bench: Towards open-ended event-level video-language understanding. *Advances in Neural Information Processing Systems*, 37:32076–32110.

Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, and 1 others. 2025. Ovo-bench: How far is your video-llms from real-world online video understanding? In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 18902–18913.

Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. 2025. Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 24045–24055.

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

Aditya Sharma, Michael Saxon, and William Yang Wang. 2024. Losing visual needles in image haystacks: Vision language models are easily distracted in short and long contexts. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 5429–5451.

Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, Zhuang Liu, Hu Xu, Hyunwoo J. Kim, Bilge Soran, Raghuraman Krishnamoorthi, Mohamed Elhoseiny, and Vikas Chandra. 2025. LongVU: Spatiotemporal adaptive compression for long video-language understanding. In *Forty-second International Conference on Machine Learning*.

Gunnar A Sigurdsson, Abhinav Gupta, Cordelia Schmid, Ali Farhadi, and Karteek Alahari. 2018. Actor and observer: Joint modeling of first and third-person videos. In *proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7396–7404.

Enxin Song, Wenhao Chai, Guanhong Wang, Yucheng Zhang, Haoyang Zhou, Feiyang Wu, Haozhe Chi, Xun Guo, Tian Ye, Yanting Zhang, and 1 others. 2024. Moviechat: From dense token to sparse memory for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18221–18232.

Xi Tang, Jihao Qiu, Lingxi Xie, Yunjie Tian, Jianbin Jiao, and Qixiang Ye. 2025. Adaptive keyframe sampling for long video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 29118–29128.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. 2019. Coin: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1207–1216.

Haibo Wang, Bo Feng, Zhengfeng Lai, Mingze Xu, Shiyu Li, Weifeng Ge, Afshin Dehghan, Meng Cao, and Ping Huang. 2026. Streambridge: Turning your offline video large language model into a proactive streaming assistant. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, and 1 others. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Yueqian Wang, Xiaojun Meng, Yuxuan Wang, Jianxin Liang, Jiansheng Wei, Huishuai Zhang, and Dongyan Zhao. 2025a. VideoLLM knows when to speak: Enhancing time-sensitive video comprehension with video-text duet interaction format. In *Findings of the*

Association for Computational Linguistics: EMNLP 2025, pages 6338–6359, Suzhou, China. Association for Computational Linguistics.

Ziyang Wang, Shoubin Yu, Elias Stengel-Eskin, Jaehong Yoon, Feng Cheng, Gedas Bertasius, and Mohit Bansal. 2025b. Videotree: Adaptive tree-based video representation for llm reasoning on long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pages 3272–3283.

Jiaer Xia, Peixian Chen, Mengdan Zhang, Xing Sun, and Kaiyang Zhou. 2025. Streaming video instruction tuning. *arXiv preprint arXiv:2512.21334*.

Haomiao Xiong, Zongxin Yang, Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Jiawen Zhu, and Huchuan Lu. 2025. Streaming video understanding and multi-round interaction with memory-enhanced knowledge. In *The Thirteenth International Conference on Learning Representations*.

Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. 2026. StreamingVLM: Real-time understanding for infinite video streams. In *The Fourteenth International Conference on Learning Representations*.

Haolin Yang, Feilong Tang, Lingxiao Zhao, Xiang An, Ming Hu, Huifa Li, Xinlin Zhuang, Yifan Lu, Xiaofeng Zhang, Abdalla Swikir, and 1 others. 2025a. Streamagent: Towards anticipatory agents for streaming video understanding. *arXiv preprint arXiv:2508.01875*.

Senqiao Yang, Yukang Chen, Zhuotao Tian, Chengyao Wang, Jingyao Li, Bei Yu, and Jiaya Jia. 2025b. Visionzip: Longer is better but not necessary in vision language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19792–19802.

Yanlai Yang, Zhuokai Zhao, Satya Narayan Shukla, Aashu Singh, Shlok Kumar Mishra, Lizhu Zhang, and Mengye Ren. 2025c. Streammem: Query-agnostic kv cache memory for streaming video understanding. *arXiv preprint arXiv:2508.15717*.

Zhenyu Yang, Kairui Zhang, Yuhang Hu, Bing Wang, Shengsheng Qian, Bin Wen, Fan Yang, Tingting Gao, Weiming Dong, and Changsheng Xu. 2026. Livestar: Live streaming assistant for real-world online video understanding. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

Linli Yao, Yicheng Li, Yuancheng Wei, Lei Li, Shuhuai Ren, Yuanxin Liu, Kun Ouyang, Lean Wang, Shicheng Li, Sida Li, and 1 others. 2025. Timechat-online: 80% visual tokens are naturally redundant in streaming videos. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 10807–10816.

Yuan Yao, Tianyu Yu, Ao Zhang, Chongyi Wang, Junbo Cui, Hongji Zhu, Tianchi Cai, Haoyu Li, Weilin Zhao, Zhihui He, and 1 others. 2024. Minicpm-v: A gpt-4v level mllm on your phone. *arXiv preprint arXiv:2408.01800*.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, and Xiaojie Jin. 2025. Flash-vstream: Efficient real-time understanding for long video streams. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 21059–21069.

Jialiang Zhang, Junlong Tong, Junyan Lin, Hao Wu, Yirong Sun, Yunpu Ma, and Xiaoyu Shen. 2026. Think-as-you-see: Streaming chain-of-thought reasoning for large vision-language models. *arXiv preprint arXiv:2603.02872*.

Peiyuan Zhang, Kaichen Zhang, Bo Li, Guangtao Zeng, Jingkang Yang, Yuanhan Zhang, Ziyue Wang, Haoran Tan, Chunyuan Li, and Ziwei Liu. 2024. Long context transfer from language to vision. *arXiv preprint arXiv:2406.16852*.

Luowei Zhou, Chenliang Xu, and Jason Corso. 2018. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

A Details of Query-aware Context Router

A.1 Training Data

We curate router training data from nine publicly available video datasets that provide temporally localized supervision: ActivityNet Captions (Krishna et al., 2017), Charades-STA (Gao et al., 2017), Charades-Ego (Sigurdsson et al., 2018), COIN (Tang et al., 2019), DiDeMo (Anne Hendricks et al., 2017), ET-Instruct (Liu et al., 2024e), TACoS (Regneri et al., 2013), TVQA (Lei et al., 2018), and YouCook2 (Zhou et al., 2018). These sources cover diverse supervision types — dense event captions, action and activity intervals, instructional procedures, and moment-level video QA.

For TVQA, we use the original natural-language questions as router queries. For the other eight sources, queries are generated from temporal annotations using a fixed set of templates, described in Section A.2. To prevent any single source from dominating, we draw approximately 12,500 training rows from each of the eight templated sources. The final training set contains 111,822 rows generated from 26,000 unique videos and 53,583 unique source events or questions. Table 4 reports per-source counts and per-class breakdowns.

A.2 Router Label Construction

Per-frame label rule. Each training row corresponds to one event in a source video with start time t_s and end time t_e. Given a stream end time T and a

Source	#Videos	#Events	#Rows
ActivityNet Captions (Krishna et al., 2017)	4,229	7,440	12,348
Charades-STA (Gao et al., 2017)	4,219	6,137	12,500
Charades-Ego (Sigurdsson et al., 2018)	5,038	11,474	12,500
COIN (Tang et al., 2019)	3,999	7,334	12,500
DiDeMo (Anne Hendricks et al., 2017)	1,207	2,921	12,449
ET-Instruct (Liu et al., 2024e)	1,729	5,502	12,500
TACoS (Regneri et al., 2013)	185	3,663	12,500
TVQA (Lei et al., 2018)	5,005	6,203	12,747
YouCook2 (Zhou et al., 2018)	389	2,909	11,778
Total	**26,000**	**53,583**	**111,822**

Table 4: Composition of the router training data. #Videos and #Events count unique videos and source-level events/questions, respectively. #Rows denotes the number of streaming training samples after generating multiple views per event.

window length W, the router observes the window $\mathcal{V}_{T-W:T}$ at 1 FPS. For each frame $v_t \in \mathcal{V}_{T-W:T}$, we assign a label as

$$y_t = \begin{cases} \textit{SHORT} & \text{if } q_{\text{type}}\text{=short, } t \in \mathcal{T}^+ \\ \textit{LONG} & \text{if } q_{\text{type}}\text{=long, } t \in \mathcal{T}^+ \\ \textit{SHORT} & \text{if } q_{\text{type}}\text{=future, } t \in \mathcal{T}^- \\ \textit{SILENCE} & \text{otherwise,} \end{cases}$$

where $\mathcal{T}^+ = [t_s, t_e + K_{\text{post}}]$ and $\mathcal{T}^- = [t_s - K_{\text{pre}}, t_s)$ are the post and pre-trigger windows, with $K_{\text{post}} = K_{\text{pre}} = 5$ seconds. The tag q_{type} comes from the query itself. Present-tense questions about objects or states are tagged *SHORT*, and past-tense or recall questions are tagged *LONG*. The post-trigger window $\mathcal{T}^+$ is the same for both *SHORT* and *LONG* rows, so the router cannot tell them apart from the trigger position alone. It has to read both the query and the visual content.

Multiple views per event. From each event, we generate several training rows by sampling (T, W) in different ways. A silence view sets $T < t_s$, so the trigger has not happened yet and every label is *SILENCE*. A trigger view sets T inside or just after the event, so the window covers the trigger and the labels become *SHORT* or *LONG*. A random view samples T uniformly over the video, so the trigger may or may not be visible. This prevents the router from tying its prediction to where T sits in the video. The window length W is sampled uniformly from $[8, 256]$. Together, these views show the router many different streaming states for each event, not just one.

Surface-form debiasing. Since q_{type} comes from the query template, the router could rely on template surface form instead of the video. We add two augmentations to block this. First, we add verbless queries about objects, states, or attributes under both q_{type} = short and q_{type} = long, so that a missing verb does not signal either label. Second, we add present-tense queries with *LONG* labels (present-tense questions whose answer lies in the distant past), so that present-tense phrasing does not signal *SHORT*. Both augmentations follow the same label rule above and only change the mix of (query, label) pairs.

A.3 License Discussion

All datasets used in this work (ActivityNet Captions, Charades-STA, Charades-Ego, COIN, DiDeMo, ET-Instruct, TACoS, TVQA, YouCook2) are publicly released under licenses permitting academic research use. Our use of all listed datasets is consistent with their intended research purposes.

Algorithm 1: Frame sampling for the *LONG* branch (π_{long})

Input: query q, retrieved segments $\mathcal{R}_{T^\star}$ with similarities $\{\alpha_i\}$, observed prefix $\mathcal{V}_{\leq T^\star}$, recent window length w_s, past frame budget N

$Z \leftarrow \sum_{s_i \in \mathcal{R}_{T^\star}} \alpha_i$;
$P \leftarrow \emptyset$;
for $s_i \in \mathcal{R}_{T^\star}$ **do**
  $n_i \leftarrow \lfloor N \cdot \alpha_i / Z \rfloor$;
  $P_i \leftarrow \text{Sample}(\mathcal{V}_{[a_i,b_i]}, n_i)$;
  $P \leftarrow P \cup P_i$;
$C_{T^\star} \leftarrow P \cup \mathcal{V}_{T^\star - w_s:T^\star}$;
return $C_{T^\star}$

B Details of Segment Retrieval

When the router predicts *LONG*, the policy π_{long} assembles the visual context $C_{T^\star}$ from two sources. It always includes the recent window $\mathcal{V}_{T^\star - w_s:T^\star}$, the same short window used in the *SHORT* branch. On top of that, it adds up to N frames drawn from the retrieved past segments $\mathcal{R}_{T^\star}$.

The past frames are distributed across segments in proportion to their retrieval similarity. A segment s_i with higher α_i contributes more frames than a less relevant one. Concretely, s_i contributes $n_i = \lfloor N \cdot \alpha_i / \sum_j \alpha_j \rfloor$ frames, and these frames are sampled uniformly from its temporal span $\mathcal{V}_{[a_i,b_i]}$. The final context $C_{T^\star}$ is the union of the sampled past frames and the recent window, ordered by

User Prompt

```
Describe this short video clip in three labeled fields, separated by ' | '.
ENTITIES: comma-separated visible objects and people (concrete nouns only, no vague words).
ACTION: a concrete verb phrase describing what is happening.
SETTING: location, environment, or scene.
Example: ENTITIES: knife, cutting board, red tomatoes, hands | ACTION: chopping tomatoes into
small pieces | SETTING: kitchen counter with overhead lighting.
Output exactly one line in this format. Do not add any preamble.
```

Figure 5: User prompt given to the downstream Video-LLM for segment captioning in the *LONG* branch.

time. Algorithm 1 summarizes the procedure, and the prompt to extract features denoted on Figure 5.

C Potential Risks

QUART processes continuous video streams, which may raise privacy concerns in deployment settings. Practitioners should ensure appropriate consent and data handling when applying QUART to user-facing applications.