LIFEKV: FULL-LIFECYCLE KV CACHE RETRIEVAL FOR STREAMING VIDEO QUESTION-ANSWERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Streaming VideoQA suffers from increasing GPU memory consumption and inference latency as the visual context grows. Existing KV retrieval methods reduce this overhead, but typically target only specific inference stages and lack a unified mechanism for controlling memory and latency throughout streaming inference. We propose LifeKV, a training-free full-lifecycle KV retrieval framework that keeps historical visual KV states revisitable during visual prefilling, question prefilling, and autoregressive decoding. The complete historical visual KV memory is stored in RAM, while only a compact query-relevant context is maintained on the GPU. LifeKV further introduces layer-ahead asynchronous KV prefetching to overlap descriptor-based retrieval and RAM-to-GPU KV recall with ongoing Transformer computation. Experiments show that LifeKV maintains competitive VideoQA accuracy while substantially reducing the growth of GPU memory usage and inference latency as the visual context length increases, providing an efficient and scalable solution for streaming video understanding over existing VideoQA models. Our code is available at this link.

1 INTRODUCTION

Multimodal large language models (MLLMs), building upon the rapid advances of large language models (Yiyang et al., 2024), have demonstrated strong capabilities in video question answering (VideoQA) (Bai et al., 2025; Maaz et al., 2024; Wang et al., 2025). However, video inputs are inherently token-intensive, with each frame typically producing hundreds of visual tokens. As a video stream grows, the accumulated visual tokens continuously increase key-value (KV) cache and attention computation, leading to rising GPU memory consumption and inference latency and eventually exceeding the model's context or memory capacity (Kim et al., 2026; Yang et al., 2026). This scalability challenge is particularly severe in streaming VideoQA, where incoming frames must be processed continuously while user queries are answered with low latency.

Existing approaches alleviate this growing context through either visual token compression or externalized memory with online retrieval. Token compression methods prune, merge, or summarize redundant visual tokens to reduce the effective context length and inference cost (Tao et al., 2025; Fu et al., 2025b; Xie et al., 2026; Chen et al., 2026b). However, such compression inevitably discards or alters part of the historical representation, which can be problematic in streaming VideoQA because future queries are unknown when the video is processed. We therefore focus on a complementary paradigm that preserves historical information and retrieves relevant evidence on demand. Existing systems store past video information as textual descriptions (Zhang et al., 2024a; Chatterjee et al., 2025), visual embeddings (Chatterjee et al., 2025; Zhang et al., 2024b), or KV caches (Ning et al., 2025; Kim et al., 2026). Compared with higher-level textual or embedding-based memories, KV cache retrieval preserves the model's native historical visual states with finer-grained information, making it particularly attractive when future queries may depend on details that are difficult to anticipate during streaming. It also enables training-free integration with pretrained MLLMs by directly reusing previously computed model states (Di et al., 2025; Ning et al., 2025; Xiao et al., 2026).

Nevertheless, existing streaming KV retrieval methods typically activate retrieval only at selected inference stages, which limits their effectiveness throughout the full streaming lifecycle.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

When retrieval is absent during visual prefilling, historical KV states can still accumulate on the GPU as the video grows. When the retrieved context is fixed after question prefilling, the model cannot adaptively refresh historical evidence as the decoding state evolves (Di et al., 2025; Xiao et al., 2026). Extending retrieval across the full lifecycle, however, introduces a new memory-latency dilemma. Keeping the complete historical visual KV cache on the GPU is not scalable, while offloading it to RAM requires query-relevant KV blocks to be selected, gathered, and transferred back to the GPU. Under synchronous execution, these operations lie directly on the inference critical path and can turn memory savings into additional response latency (Liu et al., 2026a; Zhang et al., 2025). These limitations motivate a full-lifecycle KV retrieval framework that keeps historical visual information revisitable while controlling GPU residency and retrieval overhead.

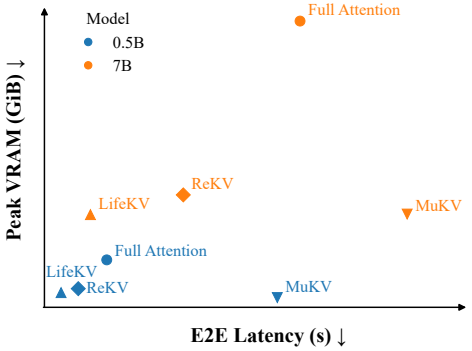


Figure 1: Latency-memory trade-off at 1024-frame context length. Lower-left indicates a more favorable trade-off.

To this end, we propose **LifeKV** (Full-**Life**cycle **KV** Cache Retrieval), a training-free framework for scalable streaming VideoQA. LifeKV extends historical KV retrieval across visual prefilling, question prefilling, and autoregressive decoding, maintaining the complete historical visual KV memory in RAM while recalling only query-relevant KV blocks to the GPU. To reduce the latency of frequent RAM-to-GPU recall, LifeKV exploits the strong retrieval similarity across adjacent Transformer layers (Zuo et al., 2025; Zhuang et al., 2026) and performs retrieval once for a group of consecutive layers. In addition, we further introduce layer-ahead asynchronous KV prefetching, which initiates retrieval and recall for an upcoming layer group while the current group is still being computed, thereby overlapping this pipeline with Transformer computation. Extensive experiments on streaming VideoQA benchmarks show that LifeKV maintains competitive answer accuracy while substantially reducing GPU memory usage and end-to-end (E2E) inference latency. As illustrated in Figure 1, LifeKV achieves a favorable latency-memory trade-off at long visual contexts across different model scales.

In summary, our main contributions are as follows:

- We introduce full-lifecycle KV retrieval for streaming VideoQA, making historical visual KV states revisitable across visual prefilling, question prefilling, and autoregressive decoding, while keeping only query-relevant historical KV blocks on the GPU to mitigate GPU memory growth as the video context accumulates.
- We develop group-wise retrieval with layer-ahead asynchronous KV prefetching, exploiting cross-layer retrieval similarity to amortize retrieval across adjacent layers while overlapping retrieval and RAM-to-GPU KV recall with Transformer computation, thereby reducing exposed retrieval latency with little additional GPU memory.
- Extensive experiments demonstrate that LifeKV maintains competitive VideoQA accuracy while achieving a favorable memory-latency trade-off, with increasingly pronounced efficiency advantages under long visual contexts.

2 Related Work

Streaming VideoQA with MLLMs. Recent MLLMs have extended VideoQA from short clips to long videos with complex temporal structure (Wu et al., 2024; Fu et al., 2025a; Niu et al., 2025; Lin et al., 2026). Dense visual tokens substantially increase both attention computation and KV cache memory, making direct video processing expensive (Zhang et al., 2026). This cost is further amplified in streaming settings, where the model must retain access to past evidence while answering online queries. Memory-based (Chen et al., 2026a; Zeng et al., 2026b) and agentic (Yin et al., 2026; Li et al., 2026) approaches address this challenge by storing external descriptions, embeddings, or intermediate states for retrieval, thereby reducing the need to keep all visual tokens active. However, such representations may lose fine-grained visual details and require additional grounding computation (Di et al., 2025), limiting their suitability for accurate and low-latency streaming VideoQA.

KV Cache Retrieval. As model-native states, KV cache enables direct reuse of historical content without requiring a separate visual memory module (Zheng et al., 2023). Prior work reduces KV cache overhead through compression (Wan et al., 2024; Zeng et al., 2026a), eviction (Li et al., 2025; Gu et al., 2026), quantization (Fan & Chen, 2026), and retrieval (Di et al., 2025). Compression and quantization may reduce representation fidelity, whereas eviction permanently removes part of the history. In contrast, retrieval can preserve the original information by retaining complete historical KV entries in external memory, making it particularly attractive for streaming VideoQA. However, existing methods do not fully exploit this fidelity advantage throughout the inference lifecycle. Many perform retrieval only during question answering or decoding, leaving visual prefilling dependent on lossy memory management. For example, ReKV (Di et al., 2025) adopts a sliding window during video processing, which may discard early evidence before a relevant question arrives. Retrieval quality also depends on the fidelity of the stored memory. MuKV (Xiao et al., 2026) retrieves from compressed multi-granularity KV memories, thereby weakening the information-preservation advantage of retrieval. The central challenge is therefore to preserve full-fidelity historical visual KV cache while enabling retrieval across all retrieval stages of streaming inference.

RAM-GPU KV Transfer in LLM Inference. Offloading the complete KV cache to RAM alleviates GPU memory pressure, but KV selection, packing, and transfer to the GPU can introduce substantial latency on the inference critical path (Liu et al., 2026b). Selective recall is practical only when a small fraction of memory pages is transferred (Chen et al., 2024). Existing LLM systems address this bottleneck in several ways. ShadowKV (Sun et al., 2024) reduces transfer volume by retaining compressed keys on the GPU, while ParisKV (Qi et al., 2026) accelerates retrieval computation. Heterogeneous inference systems overlap memory movement with model execution (Sheng et al., 2023; Aminabadi et al., 2022). InfiniGen predicts and prefetches KV entries for subsequent layers (Lee et al., 2024), whereas recent methods use speculative or asynchronous recall to hide both selection and transfer latency (Liu et al., 2026b; Wu et al., 2026). These approaches primarily target long-text inference by exploiting locality across layers or adjacent decoding steps. Nevertheless, software-based scheduling that coordinates RAM-GPU KV transfer across visual prefilling, question prefilling, and answer decoding remains underexplored.

3 METHOD

3.1 PROBLEM FORMULATION AND METHOD OVERVIEW

Given a streaming video $\mathcal{V} = \{f_t\}_{t=1}^{T}$ arriving sequentially and a sequence of questions $\mathcal{Q} = \{q_r\}_{r=1}^{M}$ issued at timestamps $\{t_r\}_{r=1}^{M}$, streaming VideoQA aims to answer each question q_r using the video content observed up to t_r. Following the standard Video-LLM inference pipeline, each stream proceeds through encoding, prefilling, and decoding (Zhang et al., 2026). KV retrieval methods leave visual encoding unchanged and perform retrieval during visual prefilling, question prefilling, and answer decoding. Each incoming frame is encoded and projected into m visual tokens, which are incrementally prefilled into the LLM to construct the corresponding KV cache. When a question arrives, its textual tokens are prefilled before autoregressive answer generation.

For layer $l \in \{1, \dots, L\}$, where L is the number of Transformer layers, we group the KV states of the i-th frame into a visual KV block

$$B_{l,i}^{\text{vis}} = \left(K_{l,i}^{\text{vis}}, V_{l,i}^{\text{vis}}\right), \tag{1}$$

where $K_{l,i}^{\text{vis}}$ and $V_{l,i}^{\text{vis}}$ contain the KV states of the m visual tokens from frame f_i. Similarly, the KV states associated with question q_r are denoted by

$$B_{l,r}^{\text{txt}} = \left(K_{l,r}^{\text{txt}}, V_{l,r}^{\text{txt}}\right). \tag{2}$$

We omit the attention-head dimension for clarity, since LifeKV operates on visual KV blocks.

After N frames have been observed, the complete historical visual KV memory at layer l is

$$\mathcal{H}_l^{\text{vis}}(N) = \left(B_{l,1}^{\text{vis}}, \dots, B_{l,N}^{\text{vis}}\right), \tag{3}$$

and the historical visual KV memory across L Transformer layers is

$$\mathcal{H}_{\text{KV}}^{\text{vis}}(N) = \left(\mathcal{H}_1^{\text{vis}}(N), \dots, \mathcal{H}_L^{\text{vis}}(N)\right). \tag{4}$$

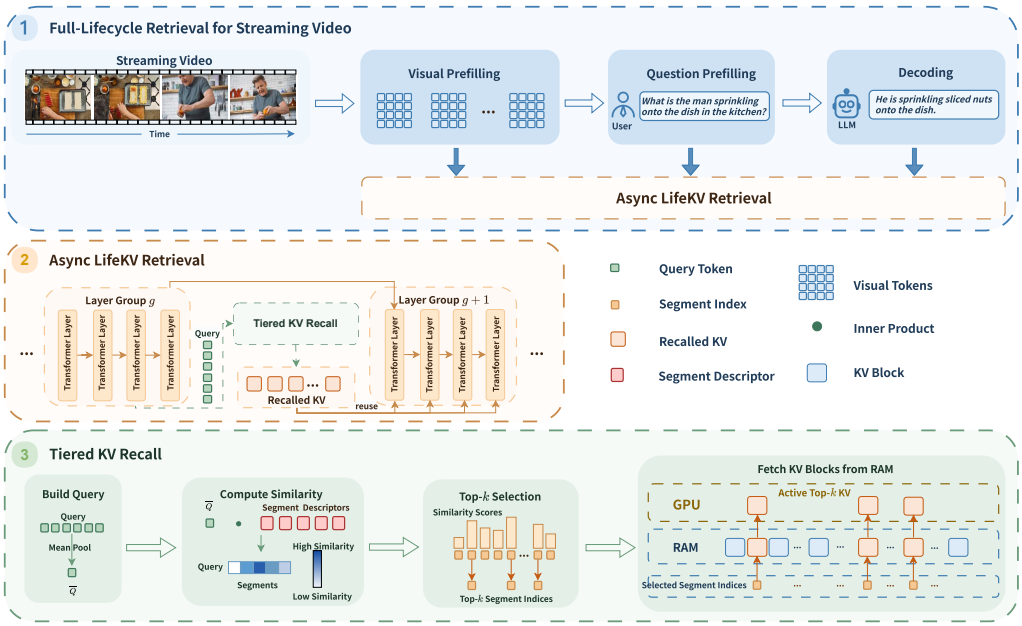


Figure 2: Overview of LifeKV. (1) Full-lifecycle retrieval revisits historical visual KV states during visual prefilling, question prefilling, and decoding. (2) Layer-ahead asynchronous KV prefetching overlaps retrieval and recall for an upcoming layer group with ongoing Transformer computation. (3) Tiered KV Recall retrieves top-k historical segments using compact descriptors and fetches the corresponding KV blocks from RAM.

This memory is defined independently of question arrival and therefore also covers visual prefilling before any question is issued.

When question q_r arrives after N_r frames, the corresponding context reference at layer l is

$$\begin{aligned} K_l^{(r)} &= [K_{l,1}^{\text{vis}}; \ldots; K_{l,N_r}^{\text{vis}}; K_{l,r}^{\text{txt}}], \\ V_l^{(r)} &= [V_{l,1}^{\text{vis}}; \ldots; V_{l,N_r}^{\text{vis}}; V_{l,r}^{\text{txt}}], \end{aligned} \tag{5}$$

LifeKV offloads only the historical visual KV memory, while the current textual context remains outside the historical visual store.

As the video stream grows, the accumulated historical visual KV cache continuously increases GPU memory and attention cost. Offloading historical visual KV cache to RAM alleviates GPU memory pressure but introduces additional retrieval and RAM-to-GPU communication overhead.

To address these challenges, we propose **LifeKV**, a training-free and model-agnostic revisitable KV memory framework for streaming VideoQA. LifeKV combines full-lifecycle KV retrieval, which selectively retrieves historical visual KV blocks during visual prefilling, question prefilling, and answer decoding, with layer-ahead asynchronous KV prefetching, which initiates retrieval and recall for upcoming layer groups while overlapping them with ongoing Transformer computation. The overview of LifeKV is shown in Figure 2.

3.2 Full-Lifecycle KV Retrieval

LifeKV extends KV retrieval across visual prefilling, question prefilling, and answer decoding. Across all three stages, stage-specific query states retrieve relevant segments from the same historical visual KV memory. We denote the retrieval stage by

$$s \in \{\text{VP}, \text{QP}, \text{DEC}\}, \tag{6}$$

where VP, QP, and DEC denote visual prefilling, question prefilling, and decoding, respectively.

Lifecycle-Wide Segment Retrieval. LifeKV adopts a visual segment as the basic retrieval unit. Let $\mathcal{S}_j$ denote the j-th segment containing c consecutive frames. The special case $c = 1$ corresponds to frame-level retrieval, which is used by default in our experiments. We construct the segment-level visual KV block at the l-th Transformer layer as

$$B_{l,j}^{\mathrm{vis,seg}} = \left(K_{l,j}^{\mathrm{vis,seg}}, V_{l,j}^{\mathrm{vis,seg}}\right), \tag{7}$$

where $K_{l,j}^{\mathrm{vis,seg}}$ and $V_{l,j}^{\mathrm{vis,seg}}$ are obtained by sequentially concatenating the frame-level visual KV states. Each segment is further represented by a compact layer-wise descriptor $D_{l,j} \in \mathbb{R}^d$, where d denotes the feature dimension. The descriptor is obtained by mean pooling its visual key states.

At a given retrieval event in stage s, let $Q_l^{(s)} \in \mathbb{R}^{n_s \times d}$ denote the query states involved in the current retrieval, where n_s is the number of query tokens. We mean-pool these query states into a single representation $\overline{Q}_l^{(s)} \in \mathbb{R}^d$ and compute the relevance between the current inference state and each historical segment as

$$\alpha_{l,j}^{(s)} = \frac{\left\langle \overline{Q}_l^{(s)}, D_{l,j} \right\rangle}{\sqrt{d}}. \tag{8}$$

Given a retrieval budget k, we select the k highest-scoring historical segments:

$$\mathcal{I}_l^{(s)} = \mathrm{TopK}\left(\{\alpha_{l,j}^{(s)}\}_{j=1}^{J^{(s)}}, k\right), \tag{9}$$

where $J^{(s)}$ denotes the number of historical segments available at the current retrieval event in stage s, and TopK returns the indices of the k largest scores.

The retrieval target remains the same evolving historical visual KV memory across the three stages, while the query states are derived from different inference contexts. During visual prefilling, queries from newly arriving visual segments retrieve relevant non-local history, allowing earlier visual events to interact with the current input before any question is issued. During question prefilling, question-derived queries retrieve question-relevant visual evidence from the accumulated video history. During decoding, query states evolve with autoregressive generation, allowing the active visual evidence to be refreshed according to the current generation state rather than remaining fixed to the selection made during question prefilling.

Tiered RAM-GPU KV Recall. LifeKV organizes historical KV memory into two storage tiers: lightweight segment descriptors and active recalled KV blocks reside on the GPU, while the complete historical visual KV blocks are stored in pinned RAM. Once $\mathcal{I}_l^{(s)}$ is obtained, the corresponding segment-level KV blocks are recalled from RAM. Let $j_1 < \cdots < j_k$ denote the selected indices after restoring their temporal order. The recalled visual KV states are

$$\widetilde{K}_l^{(s)} = [K_{l,j_1}^{\mathrm{vis,seg}}; \ldots; K_{l,j_k}^{\mathrm{vis,seg}}], \qquad \widetilde{V}_l^{(s)} = [V_{l,j_1}^{\mathrm{vis,seg}}; \ldots; V_{l,j_k}^{\mathrm{vis,seg}}], \tag{10}$$

and are transferred to the GPU as the active historical visual context. The KV states within each segment are stored contiguously in RAM, enabling block-wise gathering and RAM-to-GPU transfer without fine-grained token-level memory accesses. In this way, the full visual history remains revisitable without keeping all historical KV states resident on the GPU.

Group-Wise Multi-Layer Retrieval. Performing retrieval independently at every Transformer layer provides layer-specific retrieval results but introduces repeated similarity search and KV transfer. Conversely, sharing a single retrieval result across the entire LLM may fail to capture the evolution of representations across layers. LifeKV therefore performs retrieval once for every group of G consecutive Transformer layers and reuses the selected segment indices within the group. Let $g \in \{1, \ldots, \lceil L/G \rceil\}$ index the retrieval groups and $l_g = 1 + (g-1)G$ denote the first layer of group g. The retrieved segment indices are shared by

$$\mathcal{I}_l^{(s)} = \mathcal{I}_{l_g}^{(s)}, \qquad l = l_g, \ldots, \min(l_g + G - 1, L). \tag{11}$$

Thus, the retrieved visual context can evolve across layer groups while redundant retrieval operations among neighboring layers are avoided.

3.3 LAYER-AHEAD ASYNCHRONOUS KV PREFETCHING

Offloading historical visual KV states to RAM substantially reduces GPU memory consumption, but introduces additional retrieval and RAM-to-GPU transfer latency compared with full-attention inference. To mitigate this overhead, LifeKV exploits retrieval similarity among neighboring layers and initiates the retrieval-and-recall pipeline for the next layer group in advance, overlapping it with ongoing Transformer computation. For the first layer group of each retrieval stage, no preceding layer computation is available for overlap; retrieval and recall are therefore performed synchronously. For a subsequent group starting at layer l_g, LifeKV evaluates retrieval scores at the earlier source layer $l_g - \delta$ and uses the resulting segment indices to prefetch the KV blocks required by the target group:

$$\mathcal{I}_{l_g}^{(s)} = \mathrm{TopK}\left(\{\alpha_{l_g-\delta,j}^{(s)}\}_{j=1}^{J^{(s)}}, k\right), \tag{12}$$

where $\delta \in \{0, 1, \ldots, G\}$ denotes the prefetch distance. The special case $\delta = 0$ corresponds to synchronous retrieval at the target layer group. For $\delta > 0$, once the top-k segment indices are determined, the corresponding KV blocks are immediately gathered from RAM and asynchronously transferred to the GPU. This retrieval-and-recall pipeline proceeds concurrently with the forward computation of the intervening Transformer layers. Ideally, the prefetched KV blocks are fully available on the GPU before layer l_g starts; otherwise, only the unfinished portion of the pipeline remains exposed as a stall on the critical path.

The choice of δ therefore controls a trade-off between retrieval freshness and latency hiding. A larger δ initiates KV prefetching earlier and provides a longer overlap window, increasing the likelihood that the required KV blocks are ready before the target layer group begins. The latency benefit saturates once the retrieval-and-recall pipeline is completely hidden by Transformer computation. However, increasing δ also enlarges the layer distance between the query representation used for retrieval and the target group, potentially reducing retrieval consistency as hidden representations evolve across layers. Conversely, a smaller δ uses a fresher query representation but provides a shorter overlap window. When this window is insufficient, the target layer group must wait for the unfinished recall pipeline. Thus, δ controls the trade-off between retrieval freshness and latency hiding through asynchronous execution.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate LifeKV on two streaming VideoQA benchmarks, OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026). For the main cross-backbone comparison, we report the real-time tasks of both benchmarks, while full OVO-Bench results are used for selected ablations. All evaluations follow official protocols. Benchmark details are provided in the appendix.

Models. We evaluate LifeKV with two Video-LLM families, LLaVA-OneVision (LLaVA-OV) (Li et al., 2024) and Qwen3-VL (Bai et al., 2025). Specifically, the main experiments include LLaVA-OV-0.5B, LLaVA-OV-7B, Qwen3-VL-2B-Instruct, and Qwen3-VL-8B-Instruct, covering different model families and scales.

Baselines and Controlled Variants. We compare LifeKV against full-attention inference (Full Attention) and existing streaming KV retrieval methods, including ReKV (Di et al., 2025) and MuKV (Xiao et al., 2026). We further construct two controlled variants to isolate the effects of tiered KV memory and asynchronous prefetching. LifeKV-GPU keeps the complete historical visual KV cache on the GPU while following the same group-wise full-lifecycle retrieval workflow. LifeKV-Sync stores the historical visual KV blocks in RAM but performs retrieval and KV recall synchronously on the forward critical path. In contrast, LifeKV overlaps retrieval and RAM-to-GPU recall with Transformer computation through layer-ahead asynchronous KV prefetching. These variants separate the effects of KV offloading from those of asynchronous scheduling.

Metrics. We evaluate both task quality and system efficiency. For task quality, we report accuracy following the official metrics of each benchmark. For system efficiency, we primarily report end-to-

Table 1: Results on the real-time subsets of OVO-Bench and StreamingBench. For OVO-Bench, we report six real-time tasks: OCR, ACR, ATR, STU, FPD, and OJR, together with their average. For StreamingBench, we report ten real-time categories: OP, CR, CS, ATP, EU, TR, PR, SU, ACP, and CT. Best results within each backbone are in bold. All results are reproduced on our machine.

Method	FPS	OVO-Bench Real-Time							StreamingBench Real-Time										
		OCR	ACR	ATR	STU	FPD	OJR	Avg.	OP	CR	CS	ATP	EU	TR	PR	SU	ACP	CT	Avg.
LLaVA-OV-0.5B																			
Full Attention	0.5 fps	61.07	51.38	50.00	39.33	**58.42**	47.28	51.25	66.12	**71.09**	56.47	**66.99**	61.15	49.53	**51.85**	39.84	58.81	**20.21**	**55.11**
ReKV	0.5 fps	55.03	47.71	**52.59**	37.08	56.44	46.74	49.26	64.50	**71.09**	**56.78**	66.02	61.15	49.22	**51.85**	40.24	58.52	**20.21**	54.75
MuKV	0.5 fps	27.52	24.77	28.45	34.27	30.69	29.89	29.26	54.47	64.06	51.42	59.55	57.96	44.24	49.07	36.99	53.98	16.49	49.22
LifeKV-GPU	0.5 fps	**63.76**	**55.05**	50.86	42.13	54.46	**56.52**	**53.80**	62.33	65.62	51.42	62.46	59.24	48.60	49.07	43.09	57.10	18.09	52.63
LifeKV-Sync	0.5 fps	59.73	50.46	49.14	**42.70**	53.47	53.80	51.55	**66.40**	64.84	53.63	**66.99**	62.42	48.91	50.93	42.28	**59.94**	18.09	54.67
LifeKV	0.5 fps	**63.76**	**55.05**	50.86	42.13	54.46	**56.52**	**53.80**	65.30	63.28	52.22	66.56	**70.15**	**51.16**	48.15	**44.08**	59.77	20.00	55.03
LLaVA-OV-7B																			
Full Attention	0.5 fps	71.81	60.55	**76.72**	50.56	**75.25**	65.76	66.78	**77.24**	**78.13**	83.91	**79.61**	65.61	**70.72**	**73.15**	62.20	65.63	**42.02**	70.90
ReKV	0.5 fps	71.14	55.96	73.28	51.12	74.26	63.04	64.80	76.15	77.34	83.91	78.32	65.61	**70.72**	**73.15**	61.38	65.34	**42.02**	70.42
MuKV	0.5 fps	44.97	51.38	62.07	38.20	61.39	45.65	50.61	64.50	75.00	76.03	71.52	**70.06**	60.12	62.96	58.13	58.81	32.45	63.25
LifeKV-GPU	0.5 fps	76.51	71.56	72.41	51.69	73.27	**72.28**	69.62	76.96	75.78	83.60	78.96	69.43	66.98	72.22	61.79	**69.03**	**42.02**	70.78
LifeKV-Sync	0.5 fps	76.51	**74.31**	72.41	**52.81**	**75.25**	70.11	**70.23**	73.44	75.00	82.65	78.32	**70.06**	67.29	**73.15**	**63.41**	68.18	**42.02**	70.18
LifeKV	0.5 fps	**77.18**	69.72	**76.72**	51.69	73.27	70.11	69.78	76.15	77.34	**85.80**	78.32	68.15	67.91	72.22	63.01	**69.03**	41.49	**71.06**
Qwen3-VL-2B-Instruct																			
Full Attention	0.5 fps	79.87	64.22	**75.00**	51.12	72.28	64.67	67.86	77.51	**71.88**	87.07	**81.88**	70.70	71.65	**72.22**	65.04	**63.92**	37.77	**71.42**
ReKV	0.5 fps	50.34	44.95	45.69	31.46	54.46	36.96	43.98	**77.78**	71.09	86.44	80.91	71.34	70.09	**72.22**	64.63	63.07	**38.83**	70.98
MuKV	0.5 fps	32.89	17.43	22.41	31.46	40.59	28.26	28.84	43.09	49.22	58.68	42.39	57.32	43.30	49.07	45.93	51.70	21.28	46.33
LifeKV-GPU	0.5 fps	84.56	62.39	65.52	56.74	73.27	65.76	68.04	75.34	67.97	83.60	77.02	70.06	71.03	68.52	65.85	63.35	35.64	69.42
LifeKV-Sync	0.5 fps	86.58	66.06	70.69	52.25	72.28	71.20	69.84	76.96	67.97	84.23	77.35	70.06	71.34	69.44	66.67	61.93	36.70	69.82
LifeKV	0.5 fps	**87.25**	**69.72**	73.28	**58.99**	**75.25**	**71.74**	**72.70**	76.42	66.41	**87.70**	77.02	**72.61**	**73.21**	67.59	**67.89**	62.78	36.17	70.58
Qwen3-VL-8B-Instruct																			
Full Attention	0.5 fps	83.89	70.64	73.28	62.36	71.29	67.93	71.57	79.95	77.34	90.54	82.52	75.80	76.32	**86.11**	68.70	67.05	**44.68**	75.43
ReKV	0.5 fps	60.40	54.13	50.86	44.94	64.36	47.28	53.66	78.59	76.56	90.54	80.26	75.80	76.32	**86.11**	66.26	67.05	**44.68**	74.67
MuKV	0.5 fps	30.20	26.61	29.31	33.15	30.69	32.61	30.43	50.95	60.94	61.51	50.16	57.96	51.40	50.93	43.09	51.14	15.43	49.78
LifeKV-GPU	0.5 fps	89.93	73.39	73.28	61.24	73.27	69.57	73.45	76.96	78.12	89.91	83.50	74.52	**76.95**	81.48	67.48	68.47	42.55	74.79
LifeKV-Sync	0.5 fps	90.60	**80.73**	76.72	**65.73**	73.27	**77.17**	77.37	80.22	**78.91**	**92.11**	83.17	78.34	76.01	83.33	**69.92**	**71.88**	**44.68**	76.63
LifeKV	0.5 fps	**91.28**	79.82	**78.45**	**65.73**	**74.26**	**77.17**	**77.78**	**80.76**	78.12	**92.11**	**85.44**	**78.98**	75.70	84.26	**69.92**	71.59	42.55	**76.79**

end (E2E) inference latency and peak GPU memory (Peak VRAM). Additional stage-wise latency, recall latency, and retrieval statistics are reported where needed for detailed system analysis.

Implementation Details. Unless otherwise specified, retrieval operates on segment-level visual KV units with a fixed top-k budget, where each segment contains c consecutive frames and the retrieved segment indices are reused within a group of Transformer layers. By default, the retrieval budget, segment size, layer-group size, and prefetch distance are set to $k = 8$, $c = 1$, $G = 4$, and $\delta = 2$, respectively. For the main benchmark experiments, we follow the benchmark-specific frame sampling and generation settings used in the corresponding evaluation protocols.

4.2 Main Results

Streaming VideoQA Accuracy. Table 1 compares LifeKV with Full Attention and existing streaming KV cache retrieval methods on the real-time tasks of OVO-Bench and StreamingBench across multiple model families and scales. Overall, LifeKV retains accuracy close to the corresponding full-attention backbone and matches or surpasses it in several settings. On the OVO-Bench real-time tasks, LifeKV achieves higher average accuracy than Full Attention across all four backbones, while on StreamingBench it remains comparable. This suggests that selective retrieval can retain task-relevant evidence without consistently degrading answer accuracy. ReKV remains competitive on LLaVA-OV but degrades substantially on Qwen3-VL in our evaluation, while MuKV shows larger accuracy drops, which is consistent with the additional approximation introduced by its compressed historical KV representation. In contrast, LifeKV preserves full-fidelity historical visual KV states and exhibits more consistent accuracy retention across the evaluated model families and scales, with relatively stable performance across different backbone architectures and model sizes.

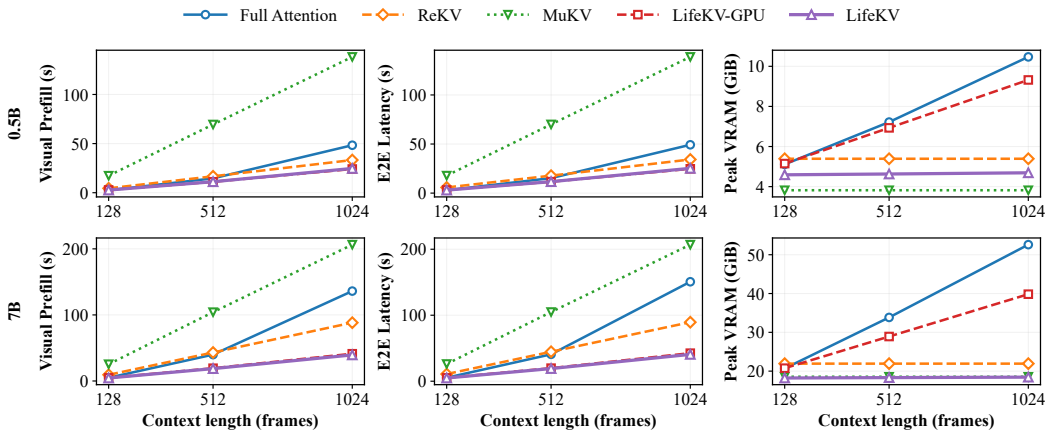


Figure 3: Controlled long-context system scaling on a single A800 GPU. Results are averaged over 5 videos at 1.0 fps under sequential inference. We compare Full Attention, ReKV, MuKV, LifeKV-GPU, and LifeKV on LLaVA-OV-0.5B and LLaVA-OV-7B as the accumulated visual context increases from 128 to 1,024 frames. From left to right, the three columns report visual-prefill latency, end-to-end (E2E) latency, and peak GPU memory, respectively. Visual-prefill latency measures streaming video processing through the language model.

System Efficiency. We further evaluate long-context system scaling under controlled sequential inference on a single A800 GPU. As shown in Figure 3, ReKV, MuKV, and LifeKV substantially flatten Peak VRAM growth compared with Full Attention by avoiding a fully GPU-resident historical KV context. MuKV achieves the lowest memory footprint in several settings through additional KV compression, but this stronger reduction is accompanied by the larger accuracy degradation observed above. The latency trends differ markedly: MuKV can become slower than Full Attention because compression and retrieval remain on the inference critical path, whereas LifeKV-GPU and LifeKV exhibit much slower latency growth by reducing the active KV context used in Transformer attention. This benefit is more pronounced on the 7B model, where reducing attention over historical KV yields larger computational savings. Notably, LifeKV remains close in latency to the GPU-resident LifeKV-GPU despite requiring RAM-to-GPU recall, indicating that layer-ahead asynchronous prefetching hides most of the additional communication overhead.

4.3 Ablation Studies

Full-Lifecycle Retrieval Ablation. We study the contribution of visual-prefill and decoding retrieval while keeping question-prefill retrieval enabled in all configurations. All other memory and retrieval configurations are kept identical. We evaluate the five longest videos in OVO-Bench on a single NVIDIA A800 GPU, where long visual contexts induce the highest memory pressure. As shown in Table 2, enabling visual-prefill retrieval consistently reduces Peak VRAM from about 8.9 GiB to 4.7 GiB (∼47%), while the associated latency increase depends on whether decoding-stage retrieval is also enabled. This result identifies visual-prefill retrieval as the main component for controlling GPU memory growth as the video stream expands. By contrast, decoding-stage retrieval provides no clear efficiency benefit in this setting, largely because OVO-Bench produces short answers. We nevertheless retain decoding retrieval as part of the unified full-lifecycle interface to support workloads with longer autoregressive generation.

Table 2: Ablation of retrieval stages on the five longest OVO-Bench videos using a single NVIDIA A800 GPU. Experiments are conducted with LLaVA-OV-0.5B.

Retrieval Stage			Efficiency	
VP	QP	DEC	E2E (s) ↓	Peak VRAM (GiB) ↓
×	✓	×	26.68	8.89
✓	✓	×	27.99	4.70
×	✓	✓	26.55	8.90
✓	✓	✓	31.53	4.71

Group-Wise Multi-Layer Retrieval. Figure 4 studies the effect of retrieval group size G on OVO-Bench using LLaVA-OV-0.5B. For valid group sizes $G \geq 1$, model accuracy changes only mildly with retrieval frequency and does not improve monotonically with denser retrieval, indicating that

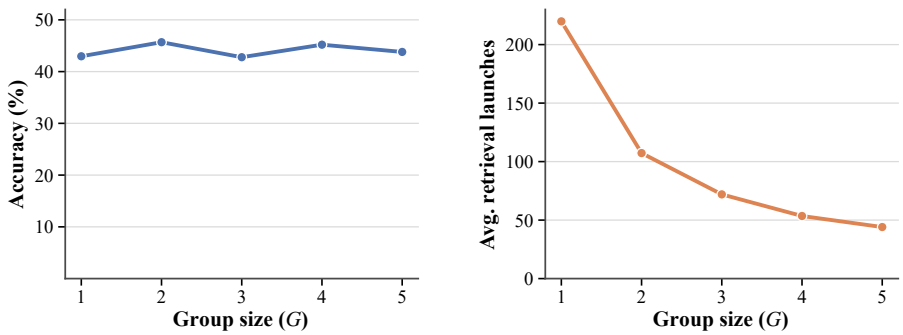


Figure 4: Effect of group-wise multi-layer retrieval on OVO-Bench with LLaVA-OV-0.5B. G denotes the number of consecutive Transformer layers sharing one retrieval result. The left plot reports VideoQA accuracy under different group sizes, while the right plot reports the average number of retrieval launches.

Table 3: Ablation of prefetch distance δ on OVO-Bench using LLaVA-OV-0.5B with $G = 4$. $\delta = 0$ denotes synchronous recall, while $\delta > 0$ launches retrieval ahead of the target layer group. Top-k Overlap measures the fraction of retrieved segments shared between the prefetched result and the synchronous retrieval result at the target consuming layer. Recall Latency reports the implementation-measured runtime attributed to KV recall in this ablation.

δ	Overall ↑	Backward	Realtime	Forward	E2E Latency (s) ↓	Recall Latency (s) ↓	Top-k Overlap ↑
0	45.20	39.02	54.73	41.85	7.964	0.601	1.000
1	43.45	36.17	53.31	40.86	8.632	0.621	0.710
2	43.58	36.06	53.80	40.90	6.991	0.233	0.732
4	44.24	35.77	56.34	40.61	6.935	0.220	0.634

retrieval at every Transformer layer is unnecessary in this setting. Increasing G substantially reduces the number of retrieval launches while largely preserving accuracy. These results suggest that a moderate retrieval frequency provides a favorable quality-efficiency trade-off, and we therefore adopt $G = 4$ as the default setting.

Layer-Ahead Asynchronous Prefetching. We study the prefetch distance δ, which determines how early retrieval and recall for an upcoming layer group are launched. As shown in Table 3, a sufficiently large look-ahead window reduces E2E latency: $\delta = 2$ and $\delta = 4$ both outperform synchronous recall ($\delta = 0$), whereas $\delta = 1$ does not provide enough overlap to reduce latency in this setting. The retrieved segments remain substantially consistent with the synchronous selection at the target layer, with Top-k Overlap ranging from 0.634 to 0.732 for the asynchronous settings. The overlap exhibits an overall decline as the prefetch distance increases, although the trend is not strictly monotonic. VideoQA accuracy also remains relatively stable across different prefetch distances, indicating that moderate retrieval staleness has limited impact on model performance. This observation is consistent with the main results in Table 1, where asynchronous LifeKV achieves accuracy close to its synchronous counterpart, LifeKV-Sync.

Additional sensitivity analyses on the segment size c and retrieval budget k are provided in Appendix B.2 and Appendix B.3, respectively.

5 Conclusion

This paper presents LifeKV, a training-free full-lifecycle KV retrieval framework for efficient streaming VideoQA. By keeping historical visual KV states revisitable across visual prefilling, question prefilling, and decoding, and by introducing layer-ahead asynchronous KV prefetching across Transformer layer groups, LifeKV mitigates GPU-memory growth and exposed retrieval latency as the visual context scales. Experiments across multiple Video-LLM backbones show that LifeKV maintains competitive VideoQA accuracy while substantially improving system scalability as the visual context length increases. These results demonstrate the potential of lifecycle-aware KV memory management for efficient multimodal understanding over streaming contexts.

AI USE STATEMENT

In this work, we use generative AI tools for writing refinement and code editing assistance. The core research ideas, methodology design, implementation, and experimental analysis are conducted by the authors. All AI-assisted outputs are manually reviewed and verified for correctness and consistency before use. We take full responsibility for the final content of this work, including all claims and artifacts.

REPRODUCIBILITY STATEMENT

We have made every effort to ensure the reproducibility of our work. All experimental settings are included in Appendix A.3. The datasets used in our experiments are publicly available through OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026).

REFERENCES

Reza Yazdani Aminabadi, Samyam Rajbhandari, Ammar Ahmad Awan, Cheng Li, Du Li, Elton Zheng, Olatunji Ruwase, Shaden Smith, Minjia Zhang, Jeff Rasley, et al. Deepspeed-inference: enabling efficient inference of transformer models at unprecedented scale. In *SC22: International Conference for High Performance Computing, Networking, Storage and Analysis*, pp. 1–15. IEEE, 2022.

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

Dibyadip Chatterjee, Edoardo Remelli, Yale Song, Bugra Tekin, Abhay Mittal, Bharat Bhatnagar, Necati Cihan CamgÃķz, Shreyas Hampali, Eric Sauser, Shugao Ma, et al. Memory-efficient streaming videollms for real-time procedural video understanding. *arXiv preprint arXiv:2504.13915*, 2025.

Renze Chen, Zhuofeng Wang, Beiquan Cao, Tong Wu, Size Zheng, Xiuhong Li, Xuechao Wei, Shengen Yan, Meng Li, and Yun Liang. Arkvale: Efficient generative llm inference with recallable key-value eviction. *Advances in Neural Information Processing Systems*, 37:113134–113155, 2024.

Tao Chen, Kun Zhang, Qiong Wu, Xiao Chen, Chao Chang, Xiaoshuai Sun, Yiyi Zhou, and Rongrong Ji. Scaling the long video understanding of multimodal large language models via visual memory mechanism. *arXiv preprint arXiv:2603.29252*, 2026a.

Xueyi Chen, Keda Tao, Kele Shao, and Huan Wang. Streamingtom: Streaming token compression for efficient video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24675–24685, 2026b.

Shangzhe Di, Zhelun Yu, Guanghao Zhang, Haoyuan Li, Hao Cheng, Bolin Li, Wanggui He, Fangxun Shu, and Hao Jiang. Streaming video question-answering with in-context video kv-cache retrieval. In *International Conference on Learning Representations*, volume 2025, pp. 42115–42127, 2025.

Jiahao Fan and Chien-Ming Chen. Efficient multimodal large language model via dynamic kv cache quantization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 20994–21001, 2026.

Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24108–24118. IEEE, 2025a.

Tianyu Fu, Tengxuan Liu, Qinghao Han, Guohao Dai, Shengen Yan, Huazhong Yang, Xuefei Ning, and Yu Wang. Framefusion: Combining similarity and importance for video token reduction on large vision language models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 22654–22663. IEEE, 2025b.

Yifeng Gu, Jianxiu Jin, Kailing Guo, and Xiangmin Xu. David: Dual-stage adaptive vision-text integrated decoupling for multimodal kv cache eviction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 21387–21395, 2026.

Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. Infinipot-v: Memory-constrained kv cache compression for streaming video understanding. *Advances in Neural Information Processing Systems*, 38:138983–139013, 2026.

Wonbeom Lee, Jungi Lee, Junghwan Seo, and Jaewoong Sim. {InfiniGen}: Efficient generative inference of large language models with dynamic {KV} cache management. In *18th USENIX symposium on operating systems design and implementation (OSDI 24)*, pp. 155–172, 2024.

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

Keliang Li, Yansong Li, Hongze Shen, Mengdi Liu, Hong Chang, and Shiguang Shan. Lenswalk: Agentic video understanding by planning how you see in videos. *arXiv preprint arXiv:2603.24558*, 2026.

Kunxi Li, Zhonghua Jiang, Zhouzhou Shen, Zhaode Wang, Chengfei Lv, Shengyu Zhang, Fan Wu, and Fei Wu. Madakv: Adaptive modality-perception kv cache eviction for efficient multimodal long-context inference. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13306–13318, 2025.

Junming Lin, Zheng Fang, Chi Chen, Haoxuan Cheng, Zihao Wan, Fuwen Luo, Ziyue Wang, Peng Li, Yang Liu, and Maosong Sun. 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)*, pp. 12147–12151. IEEE, 2026.

Di Liu, Meng Chen, Baotong Lu, Huiqiang Jiang, Zhenhua Han, Qianxi Zhang, Qi Chen, Chengruidong Zhang, Bailu Ding, Kai Zhang, et al. Retrievalattention: Accelerating long-context llm inference via vector retrieval. *Advances in Neural Information Processing Systems*, 38:54358–54385, 2026a.

Guangda Liu, Chengwei Li, Zhenyu Ning, Jing Lin, Yiwu Yao, Danning Ke, Minyi Guo, and Jieru Zhao. Freekv: Boosting kv cache retrieval for efficient llm inference. In *International Conference on Learning Representations*, volume 2026, pp. 49150–49173, 2026b.

Muhammad Maaz, Hanoona Rasheed, Salman Khan, and Fahad Khan. Video-chatgpt: Towards detailed video understanding via large vision and language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12585–12602, 2024.

Zhenyu Ning, Guangda Liu, Qihao Jin, Chengwei Li, Wenchao Ding, Minyi Guo, and Jieru Zhao. Livevlm: Efficient online video understanding via streaming-oriented kv cache and retrieval. *arXiv preprint arXiv:2505.15269*, 2025.

Junbo Niu, Yifei Li, Ziyang Miao, Chunjiang Ge, Yuanhang Zhou, Qihao He, Xiaoyi Dong, Haodong Duan, Shuangrui Ding, Rui Qian, et al. Ovo-bench: How far is your video-llms from real-world online video understanding? In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18902–18913. IEEE, 2025.

Yanlin Qi, Xinhang Chen, Huiqiang Jiang, Qitong Wang, Botao Peng, and Themis Palpanas. Pariskv: Fast and drift-robust kv-cache retrieval for long-context llms. *arXiv preprint arXiv:2602.07721*, 2026.

Ying Sheng, Lianmin Zheng, Binhang Yuan, Zhuohan Li, Max Ryabinin, Beidi Chen, Percy Liang, Christopher Ré, Ion Stoica, and Ce Zhang. Flexgen: High-throughput generative inference of large language models with a single gpu. In *International Conference on Machine Learning*, pp. 31094–31116. PMLR, 2023.

Hanshi Sun, Li-Wen Chang, Wenlei Bao, Size Zheng, Ningxin Zheng, Xin Liu, Harry Dong, Yuejie Chi, and Beidi Chen. Shadowkv: Kv cache in shadows for high-throughput long-context llm inference. *arXiv preprint arXiv:2410.21465*, 2024.

Keda Tao, Can Qin, Haoxuan You, Yang Sui, and Huan Wang. Dycoke: Dynamic compression of tokens for fast video large language models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18992–19001. IEEE, 2025.

Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. Look-m: Look-once optimization in kv cache for efficient multimodal long-context inference. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 4065–4078, 2024.

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

Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

Wenbo Wu, Qingyi Si, Xiurui Pan, Ye Wang, and Jie Zhang. Louiskv: Efficient kv cache retrieval for long input-output sequences. In *International Conference on Learning Representations*, volume 2026, pp. 65401–65421, 2026.

Junbin Xiao, Jiajun Chen, Tianxiang Sun, Xun Yang, and Angela Yao. Mukv: Multi-grained kv cache compression for long streaming video question-answering. *arXiv preprint arXiv:2605.22269*, 2026.

Yiweng Xie, Bo He, Junke Wang, Xiangyu Zheng, Ziyi Ye, and Zuxuan Wu. Fluxmem: Adaptive hierarchical memory for streaming video understanding. *arXiv preprint arXiv:2603.02096*, 2026.

Yaoxin Yang, Peng Ye, Xudong Tan, Chongjun Tu, Maosen Zhao, Jia Hao, and Tao Chen. Revisiting multimodal kv cache compression: A frequency-domain-guided outlier-kv-aware approach. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 39550–39560, 2026.

Yufei Yin, Qianke Meng, Minghao Chen, Jiajun Ding, Zhenwei Shao, and Zhou Yu. Videoarm: Agentic reasoning over hierarchical memory for long-form video understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24042–24051, 2026.

MU Yiyang, CHEN Hanxiao, and LI Hongwei. Advances in security and privacy- preserving techniques for large language models. *Journal of Cybersecurity*, 2(1):40–49, 2024.

Bowen Zeng, Feiyang Ren, Jun Zhang, Xiaoling Gu, Ke Chen, Lidan Shou, and Huan Li. Hybridkv: Hybrid kv cache compression for efficient multimodal large language model inference. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13018–13034, 2026a.

Xiangyu Zeng, Kefan Qiu, Qingyu Zhang, Xinhao Li, Jing Wang, Jiaxin Li, Ziang Yan, Kun Tian, Meng Tian, Xinhai Zhao, et al. Streamforest: Efficient online video understanding with persistent event memory. *Advances in Neural Information Processing Systems*, 38:75804–75835, 2026b.

Ce Zhang, Taixi Lu, Md Mohaiminul Islam, Ziyang Wang, Shoubin Yu, Mohit Bansal, and Gedas Bertasius. A simple llm framework for long-range video question-answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 21715–21737, 2024a.

Hailin Zhang, Xiaodong Ji, Yilin Chen, Fangcheng Fu, Xupeng Miao, Xiaonan Nie, Weipeng Chen, and Bin Cui. Pqcache: Product quantization-based kvcache for long context llm inference. *Proceedings of the ACM on Management of Data*, 3(3):1–30, 2025.

Haoji Zhang, Yiqin Wang, Yansong Tang, Yong Liu, Jiashi Feng, Jifeng Dai, and Xiaojie Jin. Flash-vstream: Memory-based real-time understanding for long video streams. *arXiv preprint arXiv:2406.08085*, 2024b.

Jun Zhang, Yicheng Ji, Feiyang Ren, Yihang Li, Bowen Zeng, Zonghao Chen, Ke Chen, Lidan Shou, Gang Chen, and Huan Li. Efficient inference for large vision-language models: Bottlenecks, techniques, and prospects. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 21036–21066, 2026.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Jeff Huang, Chuyue Sun, Cody_Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E Gonzalez, et al. Efficiently programming large language models using sglang. 2023.

Jiedong Zhuang, Lu Lu, Ming Dai, Rui Hu, Jian Chen, Qiang Liu, and Haoji Hu. Q cache: Visual attention is valuable in less than half of decode layers for multimodal large language model. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 14031–14039, 2026.

Youhui Zuo, Sibo Wei, Chen Zhang, Zhuorui Liu, Wenpeng Lu, and Dawei Song. Windowkv: Task-adaptive group-wise kv cache window selection for efficient llm inference. *arXiv preprint arXiv:2503.17922*, 2025.

A Experimental Details

A.1 Baseline Details

We provide additional details of the main streaming KV-retrieval baselines considered in our experiments. Both ReKV and MuKV are training-free approaches that reuse the historical visual KV cache for streaming VideoQA, but differ in how the cache is constructed and retrieved.

Full Attention. We use the original backbone without KV retrieval or compression as the full-context reference. All historical visual KV states remain available to the Transformer, allowing each query to attend to the entire observed video context. While this provides a strong reference for answer accuracy, its GPU memory footprint and attention cost grow with the accumulated visual context.

ReKV. ReKV (Di et al., 2025) is a training-free streaming VideoQA framework based on in-context visual KV-cache retrieval. During video processing, it introduces sliding-window attention to limit the computation associated with newly arriving frames, while the processed historical KV caches are offloaded to RAM or disk rather than being kept entirely on the GPU. When a question arrives, ReKV retrieves query-relevant historical KV entries using either an external retriever or representations derived from the Video-LLM itself, and reloads the selected KV states for question answering. This design decouples continuous video processing from subsequent question answering and substantially reduces the amount of historical context that must remain active on the GPU.

MuKV. MuKV (Xiao et al., 2026) further compresses the historical visual KV memory using a multi-grained representation. Specifically, it organizes visual information at patch, frame, and segment levels and applies a dual-signal compression strategy based on attention and frequency information. For online question answering, MuKV performs semi-hierarchical retrieval: coarse segment-level representations are first used to identify globally relevant temporal regions, after which information from finer granularities is retrieved and calibrated using cross-grained consistency. The resulting multi-grained KV states are then used as the visual context for autoregressive answer generation.

Unless otherwise specified, we reproduce the retrieval baselines under the same backbone, video sampling, and evaluation protocol as LifeKV to enable controlled comparison.

A.2 Benchmark Details

We evaluate streaming video understanding on the real-time subsets of OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026). Both benchmarks associate questions with specific timestamps in the video stream, requiring the model to answer using only information available at the corresponding inference time.

OVO-Bench. OVO-Bench is designed to evaluate temporal awareness in online video understanding. The benchmark contains 644 videos and approximately 2,800 fine-grained timestamped annotations across 12 tasks, which are organized into three online understanding modes: Backward Tracing, Real-Time Visual Perception, and Forward Active Responding. Backward Tracing evaluates the ability to retrieve and reason over previously observed video content through Episodic Memory (EPM), Action Sequence Identification (ASI), and Hallucination Detection (HLD). Real-Time Visual Perception evaluates understanding of the current or nearby visual context through Optical Character Recognition (OCR), Action Recognition (ACR), Attribute Recognition (ATR), Spatial Understanding (STU), Future Prediction (FPD), and Object Recognition (OJR). Forward Active Responding requires the model to defer its response until sufficient future evidence becomes available, and consists of Repetition Event Count (REC), Sequential Steps Recognition (SSR), and Clues Reveal Responding (CRR). We evaluate LifeKV on all three task groups. Due to space constraints, the main paper presents detailed per-task results for Real-Time Visual Perception, while the complete evaluation over all three groups is reported in the appendix. We report accuracy for the individual tasks and the corresponding group averages.

StreamingBench. StreamingBench is a comprehensive benchmark for evaluating MLLMs under continuous video streams. The complete benchmark contains 900 videos and 4,500 human-curated question-answer pairs, with five timestamped questions associated with each video. Its evaluation tasks cover three major aspects: Real-Time Visual Understanding, Omni-Source Understanding, and Contextual Understanding. Since our work focuses on visual KV memory, we follow the standard visual-only setting and evaluate on the Real-Time Visual Understanding subset, which contains 500 videos and 2,500 multiple-choice questions. The subset consists of ten task categories: Object Perception (OP), Causal Reasoning (CR), Clip Summarization (CS), Attribute Perception (ATP), Event Understanding (EU), Text-Rich Understanding (TR), Prospective Reasoning (PR), Spatial Understanding (SU), Action Perception (ACP), and Counting (CT). These categories jointly evaluate instantaneous visual perception, temporal and causal reasoning, event-level understanding, textual content recognition, spatial reasoning, action understanding, and temporal counting. We report accuracy for each category together with the overall multiple-choice accuracy.

A.3 Implementation Details

General setup. LifeKV is training-free and does not require any additional model training or parameter updates. All experiments are conducted on NVIDIA A800 GPUs with 80 GB of memory. We use the official model checkpoints and preserve the original visual preprocessing, tokenizer, and attention configuration of each backbone. Unless otherwise specified, videos are processed in a streaming manner at 0.5 FPS. For multiple-choice evaluation, we use the same prompt templates and generation settings for Full Attention, ReKV, MuKV, and LifeKV to ensure a controlled comparison. Each model-method-benchmark configuration can be evaluated on a single A800 GPU, and independent accuracy evaluations are parallelized across available GPUs when resources permit.

LifeKV configuration. Unless otherwise specified, LifeKV performs retrieval over segment-level visual KV units using a fixed top-k budget, where each segment contains c consecutive frames. Retrieval is performed once per group of G consecutive Transformer layers, and the selected segment indices are reused within the group. The complete historical visual KV cache is maintained in RAM, while lightweight segment descriptors and the currently active KV blocks reside on the GPU. To hide RAM-to-GPU recall latency, LifeKV employs layer-ahead asynchronous prefetching with a prefetch distance of δ, initiating retrieval and KV transfer before the corresponding layer group is reached. By default, we set the retrieval budget to $k = 8$, the segment size to $c = 1$ frame, the layer-group size to $G = 4$, and the prefetch distance to $\delta = 2$. These settings are shared across model scales and benchmarks unless otherwise stated, while individual hyperparameters are varied only in the corresponding ablation studies.

Inference and evaluation. For the main benchmark experiments, we follow the benchmark-specific video sampling and generation protocols of OVO-Bench and StreamingBench. Videos are processed chronologically in a streaming manner, and each question is answered at its associated timestamp using only the visual observations available up to that point. For controlled comparisons, Full Attention, ReKV, MuKV, and LifeKV use the same backbone checkpoint, visual preprocessing, frame sampling, prompt format, and decoding configuration for each benchmark. LifeKV does not require additional training, fine-tuning, or task-specific calibration, and no hyperparameters are tuned on the evaluation sets.

Efficiency measurement. To obtain reproducible system-level comparisons, all experiments that report latency or GPU memory usage are profiled with exactly one model-method configuration running on a dedicated A800 GPU, without other concurrent GPU workloads. End-to-end latency includes visual prefilling, retrieval and KV recall, question processing, and answer generation. Peak GPU memory is measured over the complete inference process. For the long-context scaling experiments, we evaluate context lengths of 128, 512, and 1024 frames and report the mean latency over five videos together with the peak GPU memory usage. All compared methods are profiled under the same hardware and evaluation configuration.

Table 4: Detailed task-level accuracy (%) on OVO-Bench. Backward includes EPM, ASI, and HLD. Realtime includes OCR, ACR, ATR, STU, FPD, and OJR. Forward includes REC, SSR, and CRR. BWD, RT, and FWD denote the corresponding group averages. Avg. denotes the average over the three task groups. The best result within each model block is shown in **bold**.

Method	Backward			Realtime						Forward			BWD	RT	FWD	Avg.
	EPM	ASI	HLD	OCR	ACR	ATR	STU	FPD	OJR	REC	SSR	CRR				
LLaVA-OV-0.5B																
Full Attention	45.45	**43.92**	22.04	61.07	51.38	50.00	39.33	**58.42**	47.28	**18.34**	63.12	52.50	**37.14**	51.25	44.65	**44.35**
ReKV	43.77	41.22	19.35	55.03	47.71	**52.59**	37.08	56.44	46.74	18.05	62.32	**60.42**	34.78	49.26	**46.93**	43.66
MuKV	39.73	31.08	**26.34**	27.52	24.77	28.45	34.27	30.69	29.89	15.33	**64.86**	53.75	32.39	29.26	44.65	35.43
LifeKV-GPU	45.45	41.22	21.51	**63.76**	**55.05**	50.86	42.13	54.46	**56.52**	18.19	49.92	54.58	36.06	**53.80**	40.90	43.58
LifeKV-Sync	**47.47**	37.84	23.66	59.73	50.46	49.14	**42.70**	53.47	53.80	18.19	52.62	51.67	36.32	51.55	40.83	42.90
LifeKV	45.45	41.22	21.51	**63.76**	**55.05**	50.86	42.13	54.46	**56.52**	18.19	49.92	54.58	36.06	**53.80**	40.90	43.58
LLaVA-OV-7B																
Full Attention	53.87	56.08	32.26	71.81	60.55	**76.72**	50.56	**75.25**	65.76	**26.65**	**69.00**	50.83	47.40	66.78	48.83	54.34
ReKV	55.56	56.08	29.57	71.14	55.96	73.28	51.12	74.26	63.04	26.07	68.20	52.92	47.07	64.80	**49.06**	53.64
MuKV	42.42	54.05	30.65	44.97	51.38	62.07	38.20	61.39	45.65	20.20	55.80	41.67	42.37	50.61	39.22	44.07
LifeKV-GPU	55.56	60.14	**33.33**	76.51	71.56	72.41	51.69	73.27	**72.28**	25.50	66.30	**53.33**	49.67	69.62	48.38	**55.89**
LifeKV-Sync	56.23	**60.81**	32.26	76.51	**74.31**	72.41	**52.81**	**75.25**	70.11	24.50	66.61	49.58	**49.77**	**70.23**	46.90	55.63
LifeKV	**57.58**	59.46	31.18	**77.18**	69.72	**76.72**	51.69	73.27	70.11	24.79	66.77	50.00	49.41	69.78	47.19	55.46
Qwen3-VL-2B-Instruct																
Full Attention	**52.86**	**54.05**	19.35	79.87	64.22	**75.00**	51.12	72.28	64.67	26.79	68.20	**57.08**	42.09	67.86	**50.69**	53.55
ReKV	34.01	43.92	17.20	50.34	44.95	45.69	31.46	54.46	36.96	**27.65**	65.66	20.83	31.71	43.98	38.05	37.91
MuKV	29.29	37.16	18.82	32.89	17.43	22.41	31.46	40.59	28.26	14.33	67.25	48.75	28.42	28.84	43.44	33.57
LifeKV-GPU	50.17	50.68	27.42	84.56	62.39	65.52	56.74	73.27	65.76	22.78	65.98	48.33	42.75	68.04	45.70	52.16
LifeKV-Sync	50.51	**54.05**	26.34	86.58	66.06	70.69	52.25	72.28	71.20	22.35	67.41	44.58	43.63	69.84	44.78	52.75
LifeKV	52.19	51.35	**33.33**	**87.25**	**69.72**	73.28	**58.99**	**75.25**	**71.74**	21.92	**68.52**	46.25	**45.62**	**72.70**	45.56	**54.63**
Qwen3-VL-8B-Instruct																
Full Attention	**57.24**	**66.22**	22.58	83.89	70.64	73.28	62.36	71.29	67.93	29.51	**65.34**	**50.83**	48.68	71.57	**48.56**	56.27
ReKV	53.54	55.41	16.67	60.40	54.13	50.86	44.94	64.36	47.28	**29.80**	63.75	40.42	41.87	53.66	44.66	46.73
MuKV	35.35	52.03	**48.92**	30.20	26.61	29.31	33.15	30.69	32.61	5.01	60.10	40.42	45.44	30.43	35.18	37.01
LifeKV-GPU	54.55	62.16	30.65	89.93	73.39	73.28	61.24	73.27	69.57	24.07	61.21	41.67	49.12	73.45	42.31	54.96
LifeKV-Sync	55.89	64.86	28.49	90.60	**80.73**	76.72	**65.73**	73.27	**77.17**	25.79	64.71	46.67	49.75	77.37	45.72	57.61
LifeKV	56.57	62.84	31.72	**91.28**	79.82	**78.45**	**65.73**	**74.26**	**77.17**	23.93	64.55	47.92	**50.37**	**77.78**	45.46	**57.87**

B Additional Experimental Results

B.1 Detailed Results on OVO-Bench

Table 4 reports the complete task-level results on OVO-Bench across all 12 tasks. Due to space constraints, the main paper focuses on the six Real-Time Visual Perception tasks, while here we further report results for Backward Tracing and Forward Active Responding. Overall, LifeKV preserves or improves the full-context performance across different model families and scales. On LLaVA-OV-7B, Qwen3-VL-2B, and Qwen3-VL-8B, LifeKV improves the overall average over Full Attention by 1.12, 1.08, and 1.60 percentage points, respectively, while remaining within 0.77 points on LLaVA-OV-0.5B. The improvement is particularly consistent on Real-Time Visual Perception, where LifeKV outperforms Full Attention across all four evaluated backbones. LifeKV also improves Backward Tracing on the larger LLaVA-OV model and both Qwen3-VL models. In contrast, the remaining accuracy gap is mainly concentrated in Forward Active Responding, whose questions require the model to wait for and integrate future observations rather than retrieve previously observed visual evidence. These detailed results further show that LifeKV maintains strong task-level performance while substantially reducing the active visual KV context used during inference.

B.2 Ablation on Segment Size c

We further examine the sensitivity of LifeKV to the temporal segment size c, i.e., the number of consecutive frames grouped into a single retrieval unit. Table 5 reports the results with $c \in \{1, 2, 3, 4\}$ on OVO-Bench and StreamingBench. Overall, LifeKV is insensitive to the choice of segment size within the evaluated range. Although the numerically best result varies across model scales and

Table 5: Ablation on the temporal segment size c using LLaVA-OV-0.5B and LLaVA-OV-7B. We report accuracy (%) on OVO-Bench and StreamingBench. The results remain stable across different segment sizes, showing limited sensitivity to this hyperparameter. The best result for each model and benchmark is shown in **bold**.

Model	c	OVO-Bench	StreamingBench
0.5B	1	43.58	55.03
	2	44.94	55.31
	3	**45.45**	**55.35**
	4	45.35	55.03
7B	1	55.46	71.06
	2	56.34	**71.38**
	3	**56.52**	70.98
	4	55.60	70.94

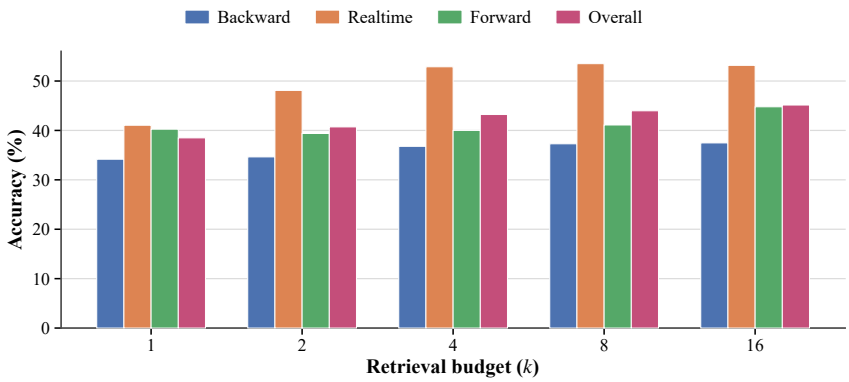


Figure 5: Effect of retrieval budget k on OVO-Bench using LLaVA-OV-0.5B. Results are reported for the three OVO-Bench task groups and their overall average.

benchmarks, the performance differences remain small in most settings. In particular, the default setting $c = 1$, which is adopted without benchmark-specific tuning, already achieves competitive accuracy across both benchmarks and model scales. This suggests that LifeKV does not rely on careful tuning of the temporal retrieval granularity to maintain strong performance. We therefore use $c = 1$ as the default configuration throughout the main experiments for simplicity and consistency, while varying c only in this ablation.

B.3 Ablation on Retrieval Budget k

Figure 5 studies the effect of the retrieval budget k on OVO-Bench. Overall accuracy improves as more historical KV is recalled, while the gain gradually saturates at larger budgets. This trend suggests that overly aggressive retrieval sparsity may miss useful historical evidence, whereas a moderate budget recovers most of the benefit. We use $k = 8$ as the default setting throughout our experiments. Importantly, this value is chosen as a fixed intermediate budget rather than tuned for the best benchmark accuracy, providing a practical trade-off between historical context coverage and retrieval overhead.

C Limitations

LifeKV is designed to retrieve a compact subset of historical visual KV states under a fixed retrieval budget, which provides an effective accuracy-efficiency trade-off for streaming VideoQA. One limitation of this sparse retrieval formulation is that it is better suited to questions whose supporting evidence is concentrated in a limited portion of the video. For queries that require comprehensive coverage of the entire stream, such as detailed video summarization, relevant evidence may be distributed across a large number of frames and cannot always be captured by a small fixed retrieval

budget. In such cases, LifeKV can be complemented with coarse uniform temporal sampling, while dynamically adapting the retrieval budget according to query scope is an interesting direction for future work. In addition, LifeKV performs retrieval at the segment level to keep similarity computation and KV recall efficient. Although this granularity works well in our experiments, finer-grained retrieval at the visual-token level could potentially provide more precise evidence selection for detail-sensitive queries. Such fine-grained retrieval also introduces substantially larger scoring, indexing, and gathering overheads. Developing hierarchical or adaptive retrieval mechanisms that selectively increase retrieval granularity while preserving the system efficiency of LifeKV therefore remains a promising direction for future work.