LEARNING MEMORY UPDATE POLICIES FOR STREAMING VIDEO UNDERSTANDING THROUGH REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Streaming video understanding requires compact memory that preserves useful information as new observations arrive. Existing training-free methods often compress visual memory using heuristic criteria, while conventional KV-cache pruning has limited applicability to the linear-attention layers of hybrid backbones. We propose **MURL** (**M**emory **U**pdate **R**einforcement **L**earning), a framework that learns visual-token retention policies for recursive memory updates through reinforcement learning. With the vision-language backbone frozen, a lightweight selector scores tokens from retained history and newly arrived observations. A history gate combines current contextual scores with stored importance estimates to guide repeated compression. We train the policy using video-level caption feedback from a frozen backbone. Experiments with Qwen3.5-4B and Qwen3.5-9B on StreamingBench, OVO-Bench and Video-MME under a streaming protocol demonstrate the effectiveness of learned memory updates for streaming video understanding.

1 INTRODUCTION

Video understanding is increasingly moving toward streaming settings (Chen et al., 2024a; Wu et al., 2024; Xiong et al., 2025; Xia et al., 2026; Xu et al., 2026; Guan et al., 2026; Wang et al., 2026a; Yang et al., 2026; Li et al., 2026; Yao et al., 2026a; Yu et al., 2026; Sun et al., 2025; Wang et al., 2026c), where visual observations arrive continuously and models must respond using the information available so far. Applications such as live video assistance and interactive monitoring require models to interpret the current scene while recalling relevant events from the past. This setting places memory management at the center of video understanding: retaining the entire visual history incurs growing storage and processing costs, whereas relying only on recent observations can remove evidence needed to answer subsequent questions. An effective streaming model therefore needs a compact visual memory that preserves useful information as new observations arrive.

Existing methods typically maintain visual memory through heuristic token selection or merging (Zhang et al., 2026b), or prune the KV cache to reduce the inference overhead of growing video sequences (Zhang et al., 2026a). However, both strategies have limitations. First, handcrafted memory update rules rely on criteria such as visual similarity to estimate information importance, without direct guidance from downstream tasks. These criteria can identify redundant content but may fail to distinguish expendable observations from subtle details essential for video understanding. Repeated compression can further compound this problem, as information discarded during an early update is no longer available when its relevance becomes apparent later. Without task-driven optimization across successive updates, such rules can limit understanding performance and lead to the irreversible loss of critical evidence. Second, KV-cache pruning is less effective in hybrid architectures with a large proportion of linear-attention layers, such as Qwen3.5 (Qwen Team, 2026). These layers summarize historical information in recurrent states rather than explicit per-token KV entries, restricting the scope of conventional cache pruning and the efficiency gains it can provide across the model.

To address these limitations, we propose MURL, a learnable visual memory framework that optimizes memory update policies through reinforcement learning. As shown in Figure 1, MURL

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.

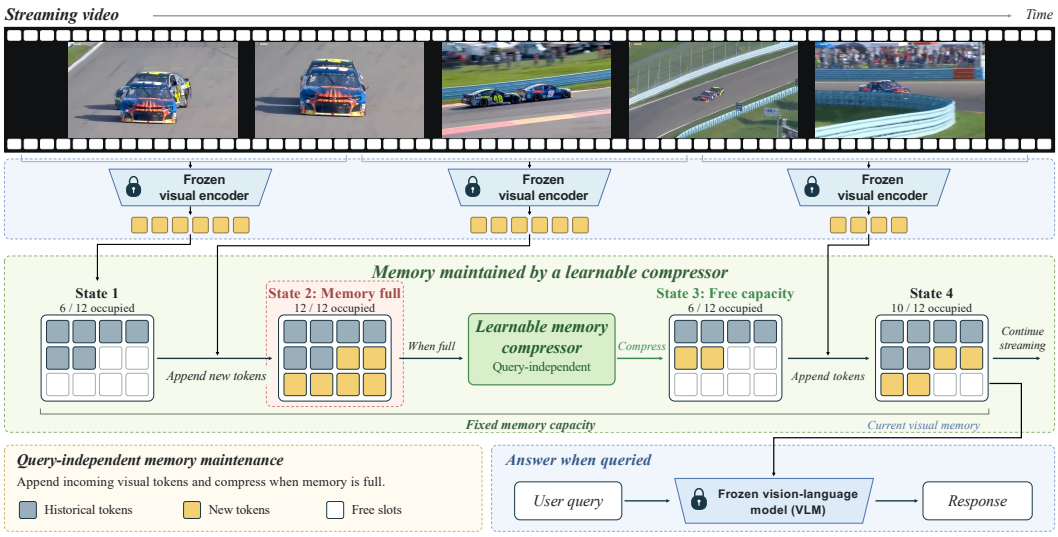


Figure 1: **Overview of our streaming visual memory framework.** When memory reaches its capacity, a learned compressor retains informative visual tokens and frees space for incoming frames. Upon receiving a query, the multimodal model generates a response from the query and current memory. Repeated compression maintains memory within a fixed budget throughout the stream.

maintains memory by selecting visual tokens from retained history and newly arriving observations under a prescribed token budget. Learning this policy requires accounting for the changing inputs and context across successive memory updates. Unlike one-shot compression of an original visual sequence, recursive memory maintenance operates on a mixture of sparsely retained historical tokens and uncompressed new observations. Strategies optimized for one-shot compression may therefore be less suited to this evolving token distribution. Moreover, earlier selections remove contextual evidence that may have informed previous importance estimates, making it difficult to assess retained tokens consistently across updates. MURL therefore learns its selection policy across successive memory updates and incorporates historical importance into current decisions. Specifically, a lightweight selector operates on visual features produced by the multimodal projector, modeling spatial and temporal context to estimate token importance. A visual history gate then adaptively combines these current scores with recursively maintained historical scores, allowing previous assessments to inform selection under the changing context. The selected tokens form the updated memory for subsequent processing, with each update using only the observations available so far.

To evaluate our approach across model scales, we train memory update policies for Qwen3.5-4B and Qwen3.5-9B (Qwen Team, 2026). We evaluate both variants on two streaming video benchmarks, StreamingBench (Lin et al., 2026) and OVO-Bench (Niu et al., 2025), as well as the offline video question-answering benchmark Video-MME (Fu et al., 2025) under a streaming protocol. The results demonstrate that MURL effectively maintains visual memory throughout video streams and achieves strong performance across diverse scenarios and model sizes.

Our main contributions are as follows:

- We propose MURL, a learnable visual memory framework that uses reinforcement learning to optimize token retention across successive memory updates for streaming video understanding.
- We design a lightweight token selector with a visual history gate that combines current contextual importance with historical scores, accounting for the changing token distribution induced by repeated compression.
- We demonstrate strong performance with Qwen3.5-4B and Qwen3.5-9B on StreamingBench, OVO-Bench, and Video-MME evaluated under a streaming protocol, validating the

effectiveness of learned memory update policies across model scales and evaluation settings.

2 Related Work

2.1 Visual Token Compression

Visual token compression improves vision-language model efficiency by selecting informative tokens or merging redundant ones using attention, similarity, diversity, or learned policies (Yao et al., 2026b; Yang et al., 2025a; Zhang et al., 2026c; Alvar et al., 2025; Zhang et al., 2024b; Shen et al., 2024). FastV (Chen et al., 2024b) prunes visual tokens using early-layer attention, while DyCoke (Tao et al., 2025) combines temporal token merging with dynamic KV-cache reduction. SCORE (Wang et al., 2026b) learns compression through reinforcement learning, using inter-frame residuals and frozen-model feedback. However, compressing a given video does not directly address recursive memory maintenance in streaming settings. Our method optimizes retention across successive updates while incorporating historical importance.

2.2 Memory Management for Long Videos

Video memory methods limit historical context by selecting informative content and consolidating redundancy in visual tokens or internal KV caches (Liang et al., 2026; Shen et al., 2026; Xie et al., 2026a). StreamForest (Zeng et al., 2026) builds a hierarchical event memory, QueryStream (Zhang et al., 2026b) prunes using query relevance and temporal novelty, and FluxMem (Xie et al., 2026b) combines temporal selection with spatial consolidation. For KV-cache compression, InfiniPot-V (Kim et al., 2026) uses temporal redundancy and value norms, StreamMem (Yang et al., 2025b) uses attention to proxy queries, and HERMES (Zhang et al., 2026a) maintains hierarchical memory across layers. However, training-free rules may miss task-relevant evidence, and KV-cache pruning does not directly extend to linear-attention states. We therefore learn visual-token retention from downstream rewards, independently of the backbone's attention and cache structures.

3 Method

Figure 2 provides an overview of MURL. Our core trainable module is a lightweight selector that recursively selects visual tokens from retained history and incoming observations, while keeping the vision-language backbone frozen. In this section, we first formulate streaming memory maintenance as recursive visual token selection under a bounded memory budget. We then introduce the architecture of our trainable selector, including the history gate and token selection mechanism. Finally, we describe the repeated-compression training procedure, reward design, and policy optimization.

3.1 Streaming Memory Formulation

We process a video incrementally using a frozen visual encoder and multimodal projector. Incoming visual tokens are accumulated with retained history, and compression is triggered when their total number exceeds the memory budget. Let V_t denote the visual tokens arriving between the $(t-1)$-th and t-th compression steps, and let M_{t-1} denote the retained historical tokens. At step t, the selector constructs the candidate collection and selects K_t tokens to form the updated memory:

$$X_t = M_{t-1} \cup V_t, \qquad M_t = \{x_{t,i} \mid i \in I_t\}, \qquad |I_t| = K_t, \tag{1}$$

where $M_0 = \varnothing$ and I_t contains the selected token indices. The updated memory is carried forward and combined with subsequent observations for the next compression. Discarded tokens cannot be recovered, so each selection determines both the evidence available for understanding and the candidates for future retention. This recursive process maintains a compact visual memory using only observations available so far.

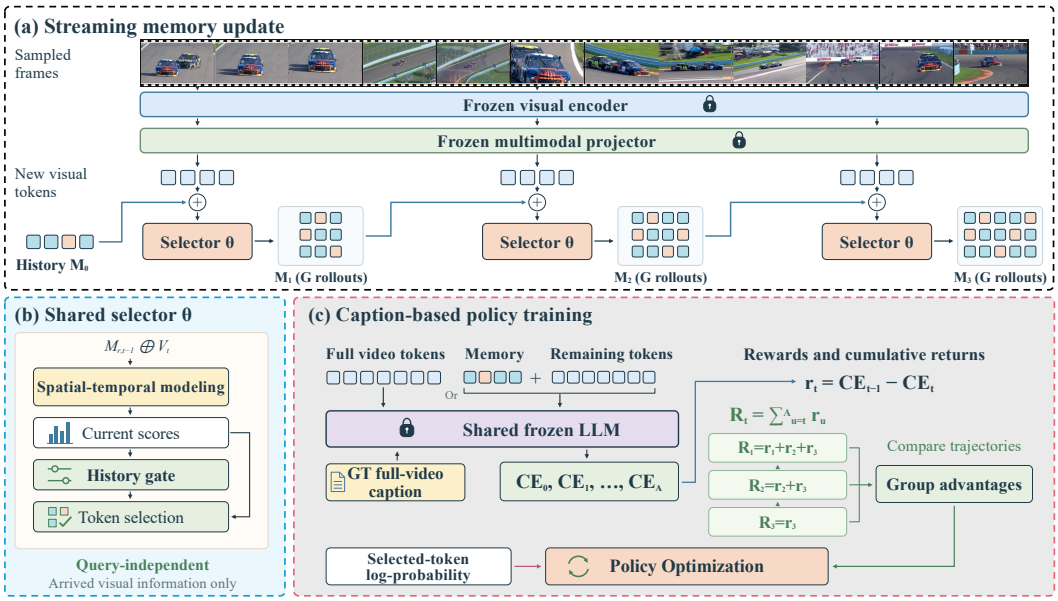


Figure 2: **Overview of MURL.** (a) We simulate streaming compression through multiple rollouts, recursively updating memory with (b) a shared selector that combines spatial-temporal modeling with a history gate. (c) A frozen backbone computes the caption cross-entropy after each compression step. Differences between consecutive losses define immediate rewards, which are summed from the current step onward to guide policy optimization.

3.2 Visual Token Selector

The selector estimates token importance from the available visual context and combines current assessments with historical scores. It consists of contextual attention blocks, a scoring head, and a visual history gate.

Contextual token scoring. Assessing token importance requires both spatial context within a frame and temporal context from preceding frames. We therefore design the selector with spatial and temporal attention to capture these complementary cues. To keep the selector lightweight and reduce the cost of contextual modeling, we first project each visual feature x_i into a lower-dimensional space:

$$h_i^{(0)} = W_{\text{in}}\,\text{LN}(x_i) + b_{\text{in}}. \tag{2}$$

A stack of spatial-temporal attention blocks then processes these representations. Spatial attention captures within-frame relationships, while temporal attention uses nearby historical tokens to assess temporal redundancy with limited computation. Since earlier compression may remove tokens at corresponding locations in preceding frames, we select nearest available neighbors rather than requiring exact positional matches.

Let $\mathcal{S}_i$ contain the available tokens in the same frame as token i. For each of the preceding W frames, we select up to k surviving tokens nearest to token i in spatial coordinates. Their union forms the temporal neighborhood $\mathcal{T}_i$, containing at most Wk tokens. Given $H = \{h_i\}$, each block computes

$$\begin{aligned} \widetilde{h}_i &= h_i + \text{SpatialAttn}\left(h_i, H_{\mathcal{S}_i}\right), \\ \widehat{h}_i &= \widetilde{h}_i + \text{TemporalAttn}\left(\widetilde{h}_i, \widetilde{H}_{\mathcal{T}_i}\right), \\ h_i^{\text{out}} &= \widehat{h}_i + \text{FFN}\left(\widehat{h}_i\right). \end{aligned} \tag{3}$$

Each attention operator uses its first argument as the query and its second as the keys and values. The feed-forward network FFN further refines each token's contextual representation.

Let $h_{t,i}$ denote the output of the final block. A scoring head comprising layer normalization and a two-layer MLP produces an importance score, standardized over the current candidates:

$$s_{t,i} = f_{\text{score}}(h_{t,i}), \qquad c_{t,i} = \frac{s_{t,i} - \mu_{X_t}(s_t)}{\max(\sigma_{X_t}(s_t), \epsilon)}. \tag{4}$$

Here, $\mu_{\mathcal{S}}$ and $\sigma_{\mathcal{S}}$ denote the mean and standard deviation over collection $\mathcal{S}$, and ϵ ensures numerical stability. The normalized score $c_{t,i}$ reflects importance under the available context.

Visual history gate. Historical tokens are rescored under different contexts as compression removes tokens and new observations arrive. A fixed retention budget may leave some redundant tokens in memory, while the contextual evidence needed to recognize their redundancy may be discarded. We therefore introduce a history gate that incorporates previous importance scores as a reference for current assessments.

Each retained token stores an importance score $b_{t-1,i}$ and a survival age $a_{t-1,i}$, counting its consecutive retention rounds. At update t, historical scores are standardized over M_{t-1} to obtain $p_{t,i}$. For each historical token, the gate takes its contextual representation, current and historical scores, their difference, and normalized survival age:

$$\begin{aligned} z_{t,i} &= f_{\text{gate}}([\text{LN}(h_{t,i}); c_{t,i}; p_{t,i}; c_{t,i} - p_{t,i}; \bar{a}_{t-1,i}]), \\ g_{t,i} &= \text{sigmoid}(z_{t,i}), \\ f_{t,i} &= \begin{cases} g_{t,i} p_{t,i} + (1 - g_{t,i}) c_{t,i}, & x_{t,i} \in M_{t-1}, \\ c_{t,i}, & x_{t,i} \in V_t, \end{cases} \end{aligned} \tag{5}$$

where $\bar{a}_{t-1,i} = \min(a_{t-1,i}, a_{\max})/a_{\max}$ and $a_{\max} = 4$. Larger gate values favor historical assessments, whereas smaller values favor current scores. New tokens use their current scores directly.

The fused scores are standardized jointly over X_t to obtain policy logits $\ell_{t,i}$. After selection, each retained token stores $b_{t,i} = \text{stopgrad}(\ell_{t,i})$ for the next update. Its survival age increases by one, with newly retained tokens initialized to age one. Historical importance is thus updated recursively. Gradients do not propagate through stored scores, but do propagate through the gate and its current contextual inputs.

Token selection. Direct supervision for token importance is difficult to obtain, so we optimize the selector through reinforcement learning by comparing the rewards of different token subsets. To enable this exploration, we add independent Gumbel noise to the policy logits before top-K_t selection:

$$\xi_{t,i} \sim \text{Gumbel}(0, 1), \qquad I_t = \text{TopK}_{K_t}\left(\{\ell_{t,i} + \xi_{t,i}\}_i\right). \tag{6}$$

This produces diverse subsets without replacement. At inference, we omit the noise and retain the K_t highest-scoring tokens.

3.3 Policy Training and Optimization

An effective compression policy should preserve visual evidence needed for continued video understanding. We use the frozen backbone's ability to predict a ground-truth caption as a proxy for how well this evidence is preserved. Since each training video has only one full-video caption, we simulate streaming memory updates by sampling several temporal positions and recursively compressing retained history together with newly arriving tokens at each position. After each compression, we evaluate the retained memory together with the uncompressed remaining frames against the same caption to calculate the reward.

Repeated-compression simulation. For each video, we first sample the number of compression steps $A \sim \text{Uniform}\{1, 2, 3, 4\}$ and their temporal positions, with the final step at the video end. We then generate G rollouts that share these A compression positions and the same retention budget at each step. Each rollout independently samples token selections, carrying the resulting memory into the next compression.

Starting from $M_{r,0} = \varnothing$, rollout r recursively updates its memory for $t = 1, \ldots, A$:

$$\begin{aligned} X_{r,t} &= M_{r,t-1} \cup V_t, & I_{r,t} &\sim \pi_\theta(\cdot \mid S_{r,t}), \\ M_{r,t} &= \{x_{r,t,i} \mid i \in I_{r,t}\}, & |I_{r,t}| &= K_t. \end{aligned} \tag{7}$$

where V_t contains the new tokens arriving between consecutive compression positions, and $S_{r,t}$ comprises the candidates $X_{r,t}$ and their historical scores and survival ages. The shared retention budget is $K_t = \mathrm{round}(\rho N_t)$, where $N_t = \sum_{j=1}^{t} |V_j|$ counts all original tokens observed so far, including previously discarded tokens. Independent Gumbel sampling produces different memory update trajectories under the same compression schedule and budgets.

To compute the policy loss, we express Gumbel Top-K sampling as sequential selection without replacement. For rollout r at step t, let $S_{r,t}$ denote the candidate tokens together with their historical scores and survival ages. Let $(i_1, \ldots, i_{K_t})$ be the selected indices ordered by descending perturbed logits. After the first $k-1$ selections, the probability of selecting token i_k is

$$\pi_\theta(i_k \mid S_{r,t}, i_{<k}) = \frac{\exp(\ell_{r,t,i_k})}{\sum_{j \in \mathcal{U}_{r,t,k}} \exp(\ell_{r,t,j})}, \tag{8}$$

where $\mathcal{U}_{r,t,k}$ contains the indices not yet selected, $\ell_{r,t,i}$ denotes the logit before adding noise, and θ includes the selector and history gate parameters.

Caption-based reward. As mentioned before, each training video has only one single full-video caption. To evaluate intermediate memory updates using this supervision, we concatenate the retained memory $M_{r,t}$ with the uncompressed tokens F_t from the remaining frames, preserving their original order. For the ground-truth caption $y = (y_1, \ldots, y_L)$, the frozen backbone computes

$$C_{r,t} = -\frac{1}{L} \sum_{j=1}^{L} \log p_\phi \left(y_j \mid y_{<j}, \mathrm{Concat}(M_{r,t}, F_t) \right), \tag{9}$$

where ϕ denotes the frozen backbone parameters. This provides feedback at successive updates without requiring segment-level captions. Future frames and the caption are used only for reward evaluation and are never provided to the selector.

We observe that an early compression decision affects the information available for later memory updates. So, in order to incorporate feedback from subsequent compressions, we assign each step the accumulated reward from that step onward. With $C_{r,0}$ denoting the caption loss on the uncompressed video, we define

$$r_{r,t} = C_{r,t-1} - C_{r,t}, \qquad R_{r,t} = \sum_{u=t}^{A} r_{r,u}. \tag{10}$$

Here, $r_{r,t}$ measures the immediate change in caption loss, while $R_{r,t}$ incorporates both the current reward and rewards from subsequent steps, encouraging selections that remain useful through repeated compression. At the final update, $F_A = \varnothing$, so evaluation relies entirely on the retained memory.

Policy optimization. We use a modified group-relative advantage to prevent substantially harmful compression decisions from receiving positive advantages merely because other rollouts perform worse. At update t, let $\mathcal{E}_t = \{r \mid r_{r,t} \geq -\delta\}$ denote the rollouts whose immediate degradation remains within tolerance. We define

$$\widehat{A}_{r,t} = \begin{cases} \dfrac{R_{r,t} - \mathrm{mean}_{q \in \mathcal{E}_t} R_{q,t}}{\mathrm{std}_{q \in \mathcal{E}_t} R_{q,t} + \epsilon}, & r \in \mathcal{E}_t, \\ r_{r,t}, & r \notin \mathcal{E}_t. \end{cases} \tag{11}$$

Eligible rollouts are compared using accumulated returns, while decisions exceeding the degradation tolerance receive negative advantages. We use $\delta = 0.01$.

Each rollout group is sampled from the current policy and used for a single parameter update. The policy loss is

$$\mathcal{L}_{\mathrm{policy}} = -\frac{1}{GA} \sum_{r=1}^{G} \sum_{t=1}^{A} \widehat{A}_{r,t} \frac{1}{K_t} \sum_{k=1}^{K_t} \log \pi_\theta(i_k \mid S_{r,t}, i_{<k}). \tag{12}$$

All selections within an update share the same advantage, and the loss is averaged over selected tokens, updates, and rollouts.

Table 1: **Main results on video understanding benchmarks.** Within each Qwen3.5 memory comparison group, methods use the same backbone, 1 fps sampling, and matched retained token counts. Our Video-MME evaluations follow a streaming protocol. Mixed indicates benchmark-dependent sampling settings. * indicates a memory budget equivalent to the visual tokens from 128 frames, while video frames are still input at 1 fps. B+R Avg. denotes the average of Backward and Realtime scores. Best results within each Qwen3.5 memory comparison group are in **bold**.

Method	Size	Frames	Video-MME				StreamingBench	OVO-Bench		
			Short	Medium	Long	Avg.	Realtime	Backward	Realtime	B+R Avg.
Open-source Offline Video MLLMs										
LLaVA-OneVision (Li et al., 2024)	7B	32	70.10	56.40	48.80	58.40	71.10	–	–	–
LLaVA-Video (Zhang et al., 2024c)	7B	1fps	–	–	–	63.3	–	–	–	–
Qwen2.5-VL-Instruct (Bai et al., 2025b)	7B	Mixed	–	–	–	65.10	73.90	44.65	59.90	52.27
Qwen3-VL-Instruct (Bai et al., 2025a)	8B	Mixed	–	–	–	71.40	73.60	42.63	62.86	52.75
KV-cache-based Memory Methods										
InfiniPot-V (Qwen2-VL) (Kim et al., 2026)	7B	768	74.11	60.78	53.44	62.80	–	–	–	–
HERMES (Qwen2.5-VL) (Zhang et al., 2026a)	7B	1 fps	–	–	53.44	60.63	79.44	49.43	68.98	59.21
HERMES (Qwen3-VL) (Zhang et al., 2026a)	8B	2 fps	–	–	–	–	81.28	49.28	73.29	61.29
Open-source Online Video MLLMs										
VideoLLM-online (Chen et al., 2024a)	8B	2 fps	–	–	–	–	36.00	–	–	19.26
Flash-VStream (Zhang et al., 2024a)	7B	1 fps	72.00	61.10	50.30	61.20	23.23	25.35	29.86	27.61
Dispider (Qian et al., 2025)	7B	1 fps	–	–	–	57.20	67.60	36.06	54.55	45.31
TimeChat-Online (Yao et al., 2025)	7B	1 fps	–	–	–	63.30	75.36	41.70	61.90	51.80
Qwen3.5 (Qwen Team, 2026)	4B	128	79.44	64.78	55.11	66.44	76.51	48.63	66.65	57.64
+ PEMF (Zeng et al., 2026)	4B	1 fps*	**79.78**	60.78	48.44	63.00	79.32	47.92	68.92	58.42
+ QueryStream (Zhang et al., 2026b)	4B	1 fps*	79.56	65.11	54.22	66.30	79.12	48.01	69.85	58.93
+ FluxMem (Xie et al., 2026b)	4B	1 fps*	79.56	69.44	55.00	68.00	78.80	47.62	68.53	58.08
+ MURL(Ours)	4B	1 fps*	**79.78**	**70.44**	**55.33**	**68.52**	**80.12**	**50.09**	**70.09**	**60.09**
Qwen3.5 (Qwen Team, 2026)	9B	128	**83.22**	70.56	60.11	**71.30**	77.64	48.48	67.67	58.07
+ PEMF (Zeng et al., 2026)	9B	1 fps*	81.67	65.22	54.56	67.15	81.72	48.51	69.47	58.99
+ QueryStream (Zhang et al., 2026b)	9B	1 fps*	81.33	69.44	59.78	70.19	81.36	47.67	68.77	58.22
+ FluxMem (Xie et al., 2026b)	9B	1 fps*	81.33	70.78	60.11	70.74	81.52	48.62	69.85	59.24
+ MURL(Ours)	9B	1 fps*	81.67	**71.44**	**60.78**	**71.30**	**81.88**	**49.67**	**70.39**	**60.03**

To discourage the history gate from globally collapsing toward zero, we add a one-sided regularizer on its mean pre-sigmoid logit. Let $\bar{z}$ be the mean gate logit over historical-token candidates across all rollouts and updates of a video. The total objective is

$$\mathcal{L} = \mathcal{L}_{\text{policy}} + \lambda_g \left[\max(0, \text{logit}(g_{\min}) - \bar{z})\right]^2, \tag{13}$$

where $g_{\min} = 0.2$ and $\lambda_g = 10^{-4}$. The regularizer acts only when the mean logit falls below the threshold; it imposes neither a per-token gate lower bound nor a direct constraint on the mean gate probability. For single-update videos, no historical gates are evaluated and this term is zero. Only the selector, including its history gate, is optimized.

4 Experiments

4.1 Experimental Setup

Models and benchmarks. We train our memory module with Qwen3.5-4B and Qwen3.5-9B (Qwen Team, 2026) as frozen backbones. We evaluate on two streaming video understanding benchmarks, OVO-Bench (Niu et al., 2025) and StreamingBench (Lin et al., 2026), and the offline video question-answering benchmark Video-MME (Fu et al., 2025), which we evaluate under a streaming protocol. Together, these benchmarks cover diverse video understanding tasks, allowing us to assess the effectiveness of our method across tasks and model scales.

Baselines. We compare against three visual memory baselines adapted to the same frozen Qwen3.5 backbones. For StreamForest (Zeng et al., 2026), we use its Persistent Event Memory Forest (PEMF). For QueryStream (Zhang et al., 2026b), we use its Query-Aware Differential Pruning (QDP), which selects visual tokens based on query relevance and temporal novelty. FluxMem (Xie et al., 2026b) combines Temporal Adjacency Selection and Spatial Domain Consolidation to remove temporal redundancy and merge spatially repetitive content. All adapted baselines use the same

Table 2: **Ablation of the history gate on Qwen3.5-9B.** OVO: OVO-Bench Backward–Realtime average.

Variant	Video-MME	StreamingBench	OVO
w/o gate	70.19	81.32	58.80
w/ gate	**71.30**	**81.88**	**60.03**

Table 3: **Ablation of repeated-compression training on Qwen3.5-9B.** STB: Streaming-Bench Realtime; OVO: Backward–Realtime average.

Training	Video-MME	STB	OVO
Single compression	68.93	81.68	58.74
Repeated compression	**71.30**	**81.88**	**60.03**

frame sampling, frozen backbone, QA prompts, and decoding settings, with visual-token counts matched at each query. Their memory-maintenance procedures differ, as detailed in Appendix A.4.

4.2 Main Results

Table 1 compares MURL with existing video understanding models and memory management methods. Under matched backbones and retained token counts, MURL achieves the best aggregate scores among the compared memory methods on all three benchmarks at both model scales. With Qwen3.5-4B, MURL achieves 68.52 on Video-MME, 80.12 on StreamingBench, and 60.09 on OVO-Bench B+R Avg.; with Qwen3.5-9B, it achieves 71.30, 81.88, and 60.03, respectively. Compared with the 128-frame Qwen3.5 baselines, our method improves performance on both streaming benchmarks while improving or matching average Video-MME performance. Notably, on Video-MME, our gains over the competing memory baselines are more pronounced on medium-length and long videos. These subsets place greater demands on repeated memory compression, suggesting that our learned update policy helps preserve useful evidence across successive updates. These results support the effectiveness of learned memory maintenance across model scales, enabling continuous video understanding under a bounded visual-token budget.

4.3 Ablation Study

To assess the effectiveness of the key components of our training framework, we conduct a series of ablation studies on the history gate, the training strategy that simulates streaming video inference, and the reward design.

Ablation on History Gate. To assess the effectiveness of the history gate, we train a variant without the gate using the same training data and all other training settings unchanged. This variant selects tokens using only current importance scores, without incorporating historical scores. As shown in Table 2, incorporating the gate improves performance across all three benchmarks on Qwen3.5-9B. These results suggest that historical importance provides useful context for token retention, complementing current assessments as memory undergoes successive compression.

Ablation on Training Strategy. We hypothesize that policies trained to compress complete videos in a single step may not generalize well to streaming settings, where memory is recursively updated from retained history and new observations. We train a variant that selects tokens once from the complete video using the same reinforcement learning procedure, with the history gate disabled during training and inference. The reward measures the change in ground-truth caption prediction loss:

$$r = \mathcal{L}_{\text{cap}}(V, y) - \mathcal{L}_{\text{cap}}(\widetilde{V}, y), \tag{14}$$

where V is the full visual token sequence, $\widetilde{V}$ contains the tokens retained after a single compression action, and y is the ground-truth video caption. Here, $\mathcal{L}_{\text{cap}}(X, y)$ denotes the token-averaged cross-entropy loss of predicting y with the frozen vision-language backbone conditioned on X. A higher reward indicates better preservation of information needed for caption prediction. As shown in Table 3, our repeated-compression framework with the history gate outperforms this single-compression baseline across all three benchmarks. Together with the history-gate ablation, these results support the joint design of repeated-compression training and history-aware selection for streaming memory maintenance.

Ablation on Reward Design. To assess the effectiveness of our reward design, we compare three reward assignment schemes using the same training data, repeated-compression schedule, and other training settings. *Path Reward* assigns the total trajectory reward to all token-selection decisions along that trajectory. *Immediate Reward* uses only the reward from the current compression step, without accumulating subsequent rewards. Our approach instead assigns each step its accumulated reward from that step onward, accounting for the effects of current selections on subsequent memory updates. As shown in Table 4, our approach outperforms both alternatives across all three benchmarks. Immediate Reward also consistently outperforms Path Reward, suggesting that assigning feedback separately to individual compression steps is beneficial. The further gains from our approach support the importance of considering future consequences when optimizing recursive memory maintenance.

Table 4: **Ablation of reward design on Qwen3.5-9B.** STB: StreamingBench Realtime; OVO: Backward–Realtime average.

Reward	Video-MME	STB	OVO
Path Reward	69.22	81.00	58.32
Immediate Reward	69.68	81.16	59.07
Ours	**71.30**	**81.88**	**60.03**

4.4 Qualitative Results

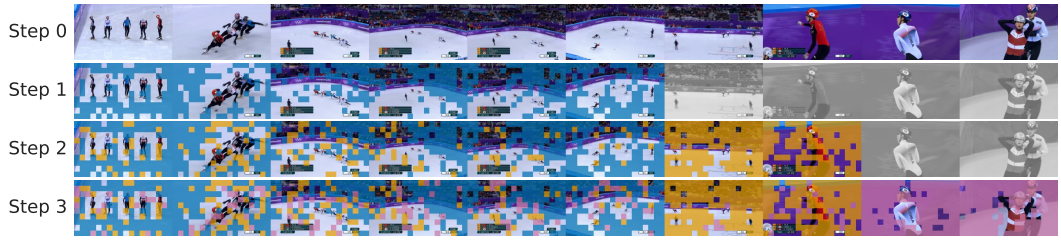


Figure 3: **Visualization of token selection across successive memory compressions.** The example is selected from Video-MME. "Step" denotes the number of compression operations performed, and different colors indicate tokens discarded at different compression steps. Most visual information about the main subjects remains preserved after repeated compression.

Figure 3 illustrates token retention across successive memory updates using an example from Video-MME, with different colors marking tokens discarded at different compression steps. Two observations emerge from this visualization. First, within a single compression step, our method preserves most regions associated with the main subjects while removing substantial background content, indicating that the learned policy can distinguish informative visual regions from less useful context. Second, across repeated compressions, the policy continues to identify redundant tokens in previously retained history and further reduces them while preserving core subject tokens. The main subjects therefore remain recognizable even in earlier frames that undergo multiple updates. These observations suggest that our policy supports both effective token selection at individual steps and sustained evidence retention throughout recursive memory maintenance.

5 Conclusion

We present MURL, a learnable visual memory framework for streaming video understanding. By optimizing recursive token selection through reinforcement learning with video-level caption feedback, our method learns to preserve useful evidence across successive memory updates while keeping the vision-language backbone frozen. A lightweight selector and visual history gate combine current contextual importance with historical scores to guide retention as the memory context evolves. Operating directly on visual tokens enables memory maintenance under a fixed token budget without relying on backbone-specific attention or cache structures. Experiments with Qwen3.5-4B and Qwen3.5-9B on StreamingBench, OVO-Bench, and Video-MME under a streaming protocol demonstrate the effectiveness of our approach. These results highlight the value of learning memory update policies from downstream feedback for streaming video understanding.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 9392–9401. IEEE, 2025.

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-vl technical report, 2025a. URL https://arxiv.org/abs/2511.21631.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report, 2025b. URL https://arxiv.org/abs/2502.13923.

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

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, pp. 19–35. Springer, 2024b.

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, 2025.

Yiran Guan, Liang Yin, Dingkang Liang, Jianzhong Ju, Zhenbo Luo, Jian Luan, Yuliang Liu, and Xiang Bai. Video streaming thinking: Videollms can watch and think simultaneously. In *European Conference on Computer Vision*, pp. 80–99. Springer, 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.

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.

Xinhao Li, Yuhan Zhu, Xiangyu Zeng, Yuhao Dong, Haoning Wu, Zhiqiu Zhang, Yuandong Yang, Changlian Ma, Qingyu Zhang, Yansong Shi, et al. Videochat3: Fully open video mllm for efficient and generalist video understanding. *arXiv preprint arXiv:2607.14935*, 2026.

Zhijia Liang, Jiaming Li, Weikai Chen, Yanhao Zhang, Haonan Lu, and Guanbin Li. Oasis: On-demand hierarchical event memory for streaming video reasoning. *arXiv preprint arXiv:2604.17052*, 2026.

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.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

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.

Rui Qian, Shuangrui Ding, Xiaoyi Dong, Pan Zhang, Yuhang Zang, Yuhang Cao, Dahua Lin, and Jiaqi Wang. Dispider: Enabling video llms with active real-time interaction via disentangled perception, decision, and reaction. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24045–24055. IEEE, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Xiaoqian Shen, Yunyang Xiong, Changsheng Zhao, Lemeng Wu, Jun Chen, Chenchen Zhu, Zechun Liu, Fanyi Xiao, Balakrishnan Varadarajan, Florian Bordes, et al. Longvu: Spatiotemporal adaptive compression for long video-language understanding. *arXiv preprint arXiv:2410.17434*, 2024.

Yujiao Shen, Shulin Tian, Jingkang Yang, and Ziwei Liu. A simple baseline for streaming video understanding. *arXiv preprint arXiv:2604.02317*, 2026.

Guangzhi Sun, Yixuan Li, Xiaodong Wu, Yudong Yang, Wei Li, Zejun Ma, and Chao Zhang. video-salmonn s: Memory-enhanced streaming audio-visual llm. *arXiv preprint arXiv:2510.11129*, 2025.

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.

Haibo Wang, Bo Feng, Zhengfeng Lai, Mingze Xu, Shiyu Li, Weifeng Ge, Afshin Dehghan, Meng Cao, and Ping Huang. Streambridge: Turning your offline video large language model into a proactive streaming assistant. *Advances in Neural Information Processing Systems*, 38:132332–132359, 2026a.

Shida Wang, YongXiang Hua, Zhou Tao, Haoyu Cao, and Linli Xu. Dynamic token compression for efficient video understanding through reinforcement learning. *arXiv preprint arXiv:2603.26365*, 2026b.

Yiyu Wang, Xuyang Liu, Xiyan Gui, Xinying Lin, Boxue Yang, Chenfei Liao, Tailai Chen, and Linfeng Zhang. Accelerating streaming video large language models via hierarchical token compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18523–18533, 2026c.

Shiwei Wu, Joya Chen, Kevin Qinghong Lin, Qimeng Wang, Yan Gao, Qianli Xu, Tong Xu, Yao Hu, Enhong Chen, and Mike Zheng Shou. Videollm-mod: Efficient video-language streaming with mixture-of-depths vision computation. *Advances in Neural Information Processing Systems*, 37:109922–109947, 2024.

Jiaer Xia, Peixian Chen, Mengdan Zhang, Xing Sun, and Kaiyang Zhou. Streaming video instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31219–31229, 2026.

Ming Xie, Zizheng Huang, Xudong Tan, Chao Wang, Xiangyu Zeng, Wenxiao Wu, Tao Chen, Limin Wang, and Yanwei Fu. Streamov: Streaming omni-video understanding via evidence-guided memory and response triggering. *arXiv preprint arXiv:2605.25621*, 2026a.

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*, 2026b.

Haomiao Xiong, Zongxin Yang, Jiazuo Yu, Yunzhi Zhuge, Lu Zhang, Jiawen Zhu, and Huchuan Lu. Streaming video understanding and multi-round interaction with memory-enhanced knowledge. In *International Conference on Learning Representations*, volume 2025, pp. 69332–69351, 2025.

Ruyi Xu, Guangxuan Xiao, Yukang Chen, Liuning He, Kelly Peng, Yao Lu, and Song Han. Streamingvlm: Real-time understanding for infinite video streams. In *International Conference on Learning Representations*, volume 2026, pp. 61463–61475, 2026.

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

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

Zhenyu Yang, Kairui Zhang, Yuhang Hu, Bing Wang, Shengsheng Qian, Bin Wen, Fan Yang, Tingting Gao, Weiming Dong, and Changsheng Xu. Livestar: Live streaming assistant for real-world online video understanding. *Advances in Neural Information Processing Systems*, 38:31266–31304, 2026.

Dingyu Yao, Junhao Zhou, Chenxu Yang, Chuanyu Qin, Haowen Hou, Zheming Liang, Congcong Wang, Yuhang Cao, Shenglong Ye, Shuai Xie, et al. Joyai-vl-interaction: Real-time vision-language interaction intelligence. *arXiv preprint arXiv:2606.14777*, 2026a.

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

Linli Yao, Long Xing, Yang Shi, Sida Li, Yuanxin Liu, Yuhao Dong, Yi-Fan Zhang, Lei Li, Qingxiu Dong, Xiaoyi Dong, et al. Towards efficient multimodal large language models: A survey on token compression. 2026b.

Xueyang Yu, Cheng Shi, Yang Wang, and Sibei Yang. Eyes wide open: Ego proactive video-llm for streaming video. *Advances in Neural Information Processing Systems*, 38:13420–13463, 2026.

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, 2026.

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*, 2024a.

Haowei Zhang, Shudong Yang, Jinlan Fu, See Kiong Ng, and Xipeng Qiu. Hermes: Kv cache as hierarchical memory for efficient streaming video understanding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8411–8430, 2026a.

Kairui Zhang, Zhenyu Yang, Bing Wang, Shengsheng Qian, and Changsheng Xu. Querystream: Advancing streaming video understanding with query-aware pruning and proactive response. In *The Fourteenth International Conference on Learning Representations*, 2026b.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. *Advances in Neural Information Processing Systems*, 38:25438–25468, 2026c.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, et al. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024b.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024c.

A Appendix

A.1 Statement on the use of large language models

Large language models (LLMs) assisted with finding and understanding related work, polishing the manuscript, and writing a limited portion of the code. All LLM-assisted code and text were reviewed by the authors. All research ideas, methodological designs, and scientific contributions originated solely from the authors, who take full responsibility for the implementation, reported results and paper.

A.2 Reproducibility Statement

To support reproducibility, we describe the memory maintenance framework, selector architecture, reward design, and policy optimization procedure in Section 3. Section A.3 provides implementation details, including training data, frame sampling, model configuration, optimization hyperparameters, and evaluation settings. These descriptions specify how the model is trained and evaluated under the reported memory budgets.

A.3 Implementation Details

We insert a lightweight token selector between the frozen vision projector and language model, containing 10.91M parameters for Qwen3.5-9B and 10.12M for Qwen3.5-4B. The selector projects visual features to 512 dimensions and applies two spatial-temporal attention blocks. The only architectural difference between the two configurations is the input projection, which maps from 4096 dimensions for Qwen3.5-9B and 2560 dimensions for Qwen3.5-4B. All other settings remain the same. For temporal attention, we set $W = 2$ and $k = 4$. Each token attends to up to four spatially nearest surviving tokens in each of the preceding two frames, yielding at most eight temporal keys. A content head predicts token importance, while a history gate combines current and historical scores using contextual features, score variation, and survival age. New tokens use only their current scores. We train the module using the reinforcement learning procedure described in the method section on 40K video-caption pairs selected from the captioning subset of LLaVA-Video-178K (Zhang et al., 2024c), uniformly sampling 128 frames per video. Each video uses one to four sampled compression events and 16 policy rollouts, retaining 37.5% of all original visual tokens observed up to each event. We use AdamW (Loshchilov & Hutter, 2017) with constant learning rates of 3×10^{-4} for the content selector and 3×10^{-5} for the history gate, an effective batch size of 32 videos, and bfloat16 precision.

During evaluation, videos arrive at 1 fps. Memory accumulates up to 11,520 tokens, equivalent to 128 frames under our visual encoding configuration, and is then compressed to 5,760 tokens using deterministic Top-K selection. Selected tokens retain their original projector features and multimodal rotary position indices. This accumulation–compression cycle repeats throughout the video. We use official benchmark prompts and greedy decoding with a maximum output length of 4,096 tokens.

A.4 Baseline Implementation Details

To compare understanding performance under matched visual token budgets, we implement three adapted baselines using Qwen3.5 visual features and a shared question-answering interface. We determine the target token count at each query time using a memory schedule with a capacity of 11,520 tokens and a post-compression budget of 5,760 tokens.

For our FluxMem (Xie et al., 2026b) baseline, we apply TAS temporal redundancy filtering and SDC spatial redundancy merging to the video observed so far. We then match the target budget using temporal novelty, the original selection results, and priority for recent tokens. When the output exceeds the target count, we further prune tokens; otherwise, we fill any shortfall from the currently observed candidates. This implementation matches the output token count at query time without reproducing our method's irreversible, repeated compression process.

For our QueryStream (Zhang et al., 2026b) baseline, we use CLIP to estimate semantic relevance between the question and visual patches, and combine it with temporal novelty and DSH history

updates to compute continuous QDP scores. We map these scores to the Qwen visual grid and align them with its temporal slices through max aggregation over pairs of frames. To enforce the target budget, we replace the original binary threshold-based removal with score ranking and retain the top 5,760 tokens whenever the candidate memory reaches capacity. We exclude answer options from the question text used for scoring.

For our PEMF/StreamForest (Zeng et al., 2026) baseline, we adopt its ToMe merging mechanism under the same capacity-based compression trigger. We reduce the candidate collection to 5,760 tokens through iterative bipartite matching and mass-weighted merging, carrying token masses across compression rounds. We do not incorporate the full hierarchical event maintenance procedure in this implementation.

For all three baselines, we preserve the original spatiotemporal positions of retained tokens and assign merged tokens the positions of their representative anchors for compatibility with Qwen3.5's mRoPE. Our adaptations match the visual token budget at query time and use the same downstream model, while retaining differences in memory maintenance procedures. They should therefore be regarded as adapted baselines rather than exact reproductions of the complete official implementations. However, this controlled setting enables a fair comparison of understanding performance under the same backbone and matched visual token budgets at query time.

A.5 Further Experiments

Ablation on Memory Capacity and Token Retention Ratio. We evaluate MURL and the adapted FluxMem baseline on Qwen3.5-9B using 1 fps input. For MURL, we vary the memory capacity between 11,520 and 23,040 tokens and the retention ratio per compression among 25%, 50%, and 75%. For each configuration, MURL recursively compresses memory whenever its capacity is exceeded. The adapted FluxMem baseline follows Appendix A.4, with its output token count matched to MURL at each query. As shown in Table 5, our method consistently outperforms FluxMem in overall accuracy across all tested configurations, with improvements on both medium and long videos. Our method also achieves higher overall accuracy as the retention ratio increases. Although the maximum memory capacity remains unchanged, a higher retention ratio leaves more tokens after each compression and results in more retained tokens per video on average at evaluation, providing more visual evidence for understanding. These results demonstrate the robustness of our learned memory update policy across different memory capacities and compression strengths.

Table 5: **Memory capacity and retention ratio on Video-MME.** Paired entries report FluxMem / Ours; better results are bold.

Capacity	Retain	Short	Medium	Long	Avg.
11,520	25%	81.33/81.33	68.67/**69.67**	55.44/**56.00**	68.48/**69.00**
	50%	81.33/81.67	70.78/**71.44**	60.11/**60.78**	70.74/**71.30**
	75%	81.33/81.33	70.89/**72.22**	60.00/**61.89**	70.74/**71.81**
23,040	25%	81.33/81.33	71.00/**72.00**	54.89/**57.78**	69.07/**70.37**
	50%	81.33/81.33	71.33/**73.56**	60.43/**61.22**	71.03/**72.04**
	75%	81.33/81.33	73.67/**74.67**	61.32/**62.33**	72.10/**72.78**

Ablation on Token Retention Ratio during Training. We investigate the sensitivity of our training procedure to the retention ratio ρ using Qwen3.5-9B, comparing 25%, 37.5%, and 50% while keeping all other training and evaluation settings unchanged. As shown in Table 6, performance remains stable across the tested ratios on StreamingBench and both OVO-Bench subsets, suggesting that the learned memory policy is robust to moderate changes in the training retention budget. The intermediate ratio of 37.5% achieves the best StreamingBench and OVO-Bench Real-

Table 6: **Ablation on token retention ratio during training.** ρ denotes the training retention ratio.

ρ	StreamingBench	OVO-Bench	
	Realtime	Backward	Realtime
25%	81.80	**49.72**	70.08
37.5%	**81.88**	49.67	**70.39**
50%	81.68	49.52	69.82

time scores, while 25% performs slightly better on OVO-Bench Backward.

The retention ratio also controls the difficulty of compression and the informativeness of the resulting reward signal. An excessively small ratio may discard substantial visual evidence across most sampled selections, increasing caption cross-entropy and producing predominantly negative immediate rewards. Although negative rewards do not inherently prevent policy optimization, consistently destructive compression may leave few successful selections for meaningful group-wise comparison. Conversely, an excessively large ratio preserves most visual information, so different token combinations may yield similar caption losses. The resulting small reward differences can weaken the learning signal and make it difficult to distinguish better retention decisions from worse ones. An intermediate ratio therefore offers a useful balance: compression is sufficiently challenging to expose differences between selections, while retaining enough evidence for informative choices to emerge. Our results support using $\rho = 37.5\%$ as the default, although the modest performance differences indicate that the method does not depend on a narrowly tuned retention ratio.

Table 7: **Efficiency comparison of memory maintenance methods.** Time denotes the average duration per update cycle. Peak memory and weights denote additional GPU memory usage.

Method	Time (ms)	Peak (GiB)	Weights (GiB)
FluxMem	199.7	0.532	0
QueryStream	1622.2	0.285	0.797
MURL (Ours)	121.8	0.930	0.041

Efficiency Analysis. Table 7 compares the computational overhead of memory maintenance. Each method follows the memory maintenance procedure described in its original paper, processing the same incoming video tokens. To compare the time required to produce memories of the same length, we apply an additional compression using each method's own strategy when its final memory exceeds the target token count. Time reports the average total processing time, including this final compression when needed. Peak reports the additional peak GPU memory during maintenance, while weights reports the memory occupied by additional model parameters. Our method requires less average maintenance time than FluxMem and QueryStream. However, accumulating tokens before compression, rather than continuously consolidating memory, incurs higher peak GPU memory usage. The learned selector also introduces additional weights. These overheads remain below 1 GiB of additional peak memory and 0.05 GiB of weights, while our method achieves better understanding performance than FluxMem and QueryStream in the main evaluations.

A.6 VISUALIZATION

Additional Visualizations. Figures 4, 5 and 6 provide additional visualizations of token retention across successive memory updates. These examples illustrate how our method preserves salient visual content while progressively removing redundant background tokens.

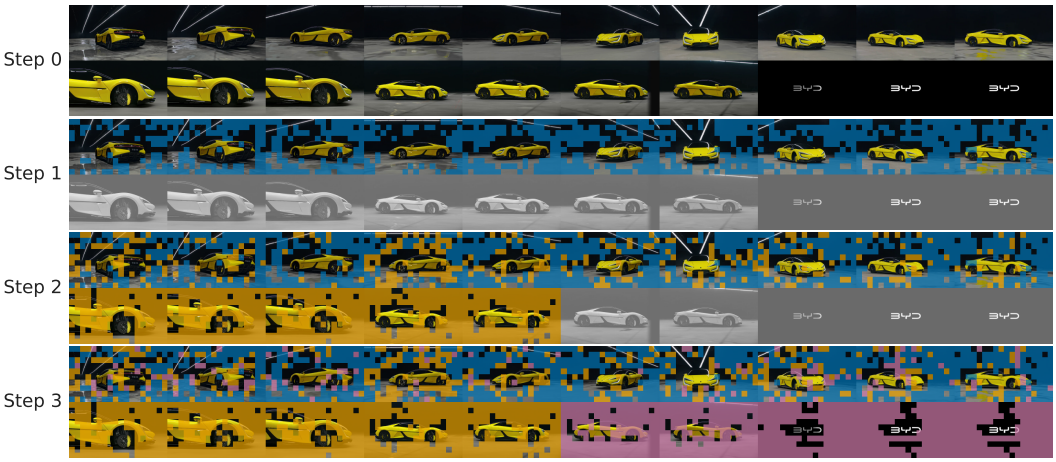


Figure 4: **Visualization of token selection across successive memory compressions.**

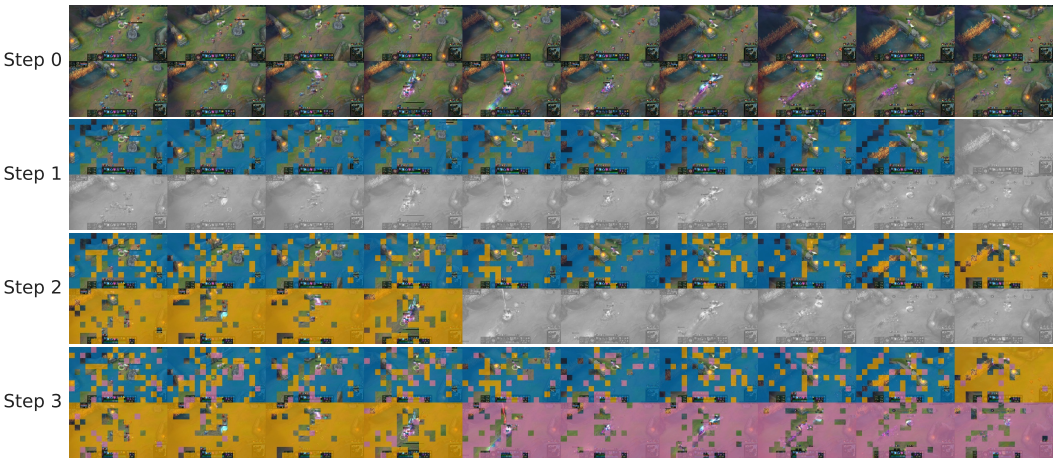


Figure 5: **Further visualization.**



Figure 6: **Further visualization.**