LEARNING TO CARRY: KV CACHE EVICTION FOR MULTI-TURN VISION-LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision-language models (VLMs) have advanced rapidly in joint visual and textual reasoning, and are now widely applied to multi-turn dialogue. However, their key-value (KV) cache grows with the conversation, increasing memory use and latency. Compressing this cache faces two challenges: eviction must anticipate unknown future queries, and retention must balance modalities despite image entries dominating the cache while receiving little current-turn attention. We propose CarryKV, a learned KV cache eviction method that decides at each turn boundary which entries to carry forward. CarryKV combines Structure-Aware Encoding (SAE), which exposes retention-relevant content and structural cues, with Sketch-Conditioned Selection (SCS), which selects entries complementary to a sketch of randomly retained cache entries. SCS also reserves a minimum image quota before filling the remaining budget by score. We train the selector through future-turn distillation, minimizing the KL divergence between the frozen VLM's predictions with full and retained caches on the following response. Training uses only 100 dialogues, takes under 25 minutes on one A100, and yields a single checkpoint reusable across budgets. Integrated into vLLM, CarryKV compacts the retained cache and physically frees evicted blocks. Across MMDU, MultiVerse, MMRC, and ConvBench on Qwen2.5-VL-7B and GLM-4.6V-Flash-9B, CarryKV maintains strong response quality under aggressive cache compression while reaching 1.37–$1.75\times$ the peak throughput of the full-cache baseline.

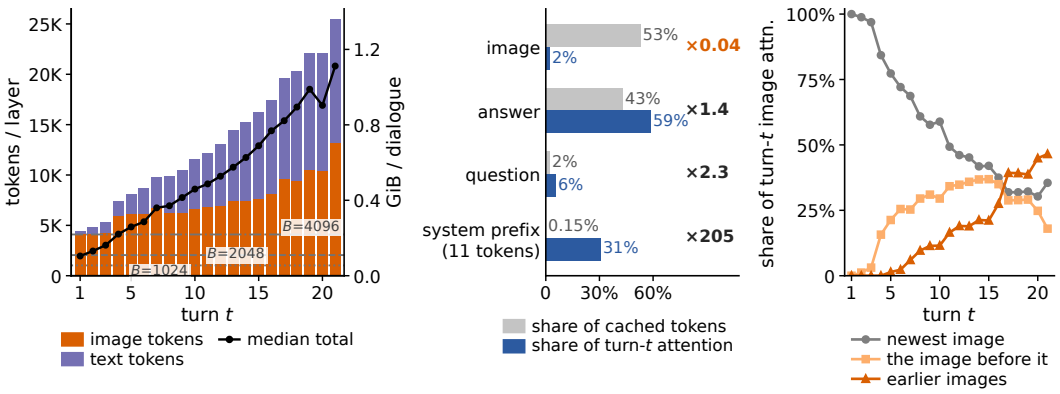
1 INTRODUCTION

Vision-language models (VLMs) (Liu et al., 2023; Chen et al., 2024b; Bai et al., 2025) extend large language models with visual modality, and are increasingly deployed as conversational assistants that sustain multi-turn interactions over multiple images, user instructions, and model responses (Liu et al., 2024b; Xue et al., 2025). In such dialogues the key-value (KV) cache is reused across turns (Gao et al., 2024; Kwon et al., 2023), so it keeps growing as the conversation proceeds, occupies an increasing share of GPU memory, and becomes the bottleneck for the later inference. As Figure 1a shows, on MMDU the median resident cache exceeds 20K tokens per layer by turn 21, more than half of them image tokens. This growth also changes the nature of compression. In single-turn use, compression is a *spatial* problem that discards redundant tokens of the image. In multi-turn use, it becomes a *temporal* one: which entries to keep for future turns.

Multi-turn VLM cache eviction presents two distinctive challenges. First, future relevance is unknown at the time of eviction. Figure 2 illustrates how visual evidence introduced early may become important again several turns later, even after new images enter the dialogue. This is not an isolated case: Figure 1c shows that, by turn 21, 46% of image attention is still allocated to images preceding the two most recent ones. Second, cache occupancy is highly imbalanced across modalities. As shown in Figure 1b, image entries constitute 53% of the cache but receive only 2% of current-turn attention, making current attention an unreliable proxy for future retention value.

Existing approaches address only part of this problem. Sparse-attention and retrieval methods reduce per-query KV access but keep the backing cache resident (Tang et al., 2024; Behnam et al., 2025; Khaki et al., 2025). Eviction methods bound resident memory using accumulated or observed attention (Zhang et al., 2023; Li et al., 2024), future-query distributions (Devoto et al., 2025), or

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.



(a) Resident cache across turns (b) Occupancy vs. attention (c) Attention by image recency

Figure 1: Cache growth and attention dynamics in multi-turn VLMs on MMDU (Qwen2.5-VL-7B, full KV cache, 110 dialogues). (a) The resident cache grows across turns and exceeds fixed budgets B. (b) By token role, image entries occupy 53% of the cache but receive only 2% of current-turn attention; the ratio is attention share over cache share. (c) Earlier images remain relevant: by turn 21, images older than the two most recent ones receive 46% of image attention.

retention coefficients predicted at insertion (Bui et al., 2026b;a). Yet in multi-turn dialogue, new images and responses continually change which historical entries matter.

We therefore propose CarryKV, a learned turn-boundary KV eviction method that re-evaluates the resident cache as the dialogue evolves. CarryKV combines *Structure-Aware Encoding (SAE)*, which exposes retention-relevant content and structural cues of each KV entry, with *Sketch-Conditioned Selection (SCS)*, which scores candidates relative to a compact sketch of a partially retained cache to preserve complementary information. To mitigate modality imbalance, SCS first reserves a minimum image quota and then fills the remaining budget by score. We train both modules through *future-turn distillation*: with the VLM frozen, each boundary decision is supervised by minimizing the KL divergence between predictions under the full and retained caches on the following response.

We further integrate CarryKV into vLLM (Kwon et al., 2023), where retained entries are compacted at the block level and evicted blocks are returned to the KV pool, physically reducing resident memory. Across two VLMs and four multi-turn benchmarks, CarryKV maintains strong response quality under aggressive cache compression while achieving the highest throughput among all evaluated methods, demonstrating a favorable quality–speed trade-off.

Our contributions are summarized as follows:

- **Empirical analysis of multi-turn KV caches.** We characterize KV cache usage in multi-turn VLMs: visual information remains relevant across distant turns despite a substantial mismatch between image cache occupancy and current-turn attention.
- **Learned turn-boundary KV cache eviction.** We propose CarryKV, which evaluates resident KV entries as the dialogue evolves through SAE and SCS, and learns retention decisions from subsequent responses via future-turn distillation.
- **System implementation and quality-speed gains.** We integrate CarryKV into vLLM with KV block reclamation. On Qwen2.5-VL-7B and GLM-4.6V-Flash-9B across MMDU, MultiVerse, MMRC, and ConvBench, CarryKV maintains strong response quality under aggressive compression while reaching 1.37–$1.75\times$ the peak throughput of the full KV.

2 Related Work

Visual token pruning and KV cache eviction. Visual token reduction is either query-agnostic, exploiting encoder attention or token redundancy (Shang et al., 2025; Yang et al., 2025; Liu et al., 2026a), or query-aware, using text-to-visual attention (Chen et al., 2024a; Zhang et al., 2025). KV eviction for LLMs estimates importance from accumulated or observed attention (H_2O, SnapKV,

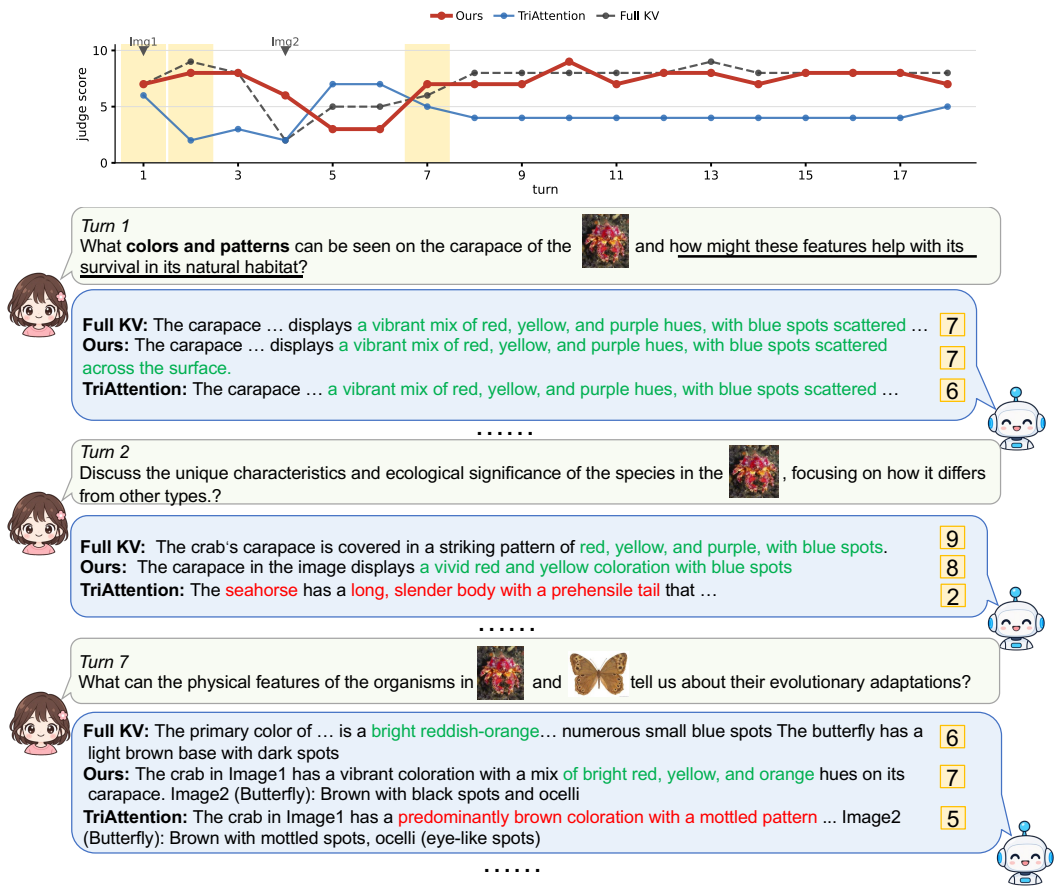


Figure 2: Example of cross-turn visual dependency. Top: per-turn judge scores (1–10); triangles mark where each image is introduced, shaded turns are expanded below. Bottom: turns 1, 2, and 7; green marks correct details, red marks errors.

PyramidKV (Zhang et al., 2023; Li et al., 2024; Cai et al., 2025)), offline-calibrated statistics (TriAttention (Mao et al., 2026)), or future-query distributions (Devoto et al., 2025). Learned methods instead train retention scorers (Huang et al., 2024; Zhou et al., 2026; Moschella et al., 2026; Dong et al., 2026; Althoubi, 2026) or compression adapters (Chari et al., 2025). For VLMs, LOOK-M, MEDA, and VL-Cache incorporate text priors, cross-modal attention, or modality-aware sparsity (Wan et al., 2024; 2025; Tu et al., 2025). Persistent multi-turn inference further requires retention decisions to generalize across queries, since entries useful for one query may not serve later ones (Kim et al., 2025; Li et al., 2025).

KV cache management for multi-turn dialogue. Multi-turn serving reuses KV caches to avoid repeated prefill (Gao et al., 2024), making cache residency a central systems concern (Kwon et al., 2023). RocketKV-MT (Behnam et al., 2025) sparsifies decoding while retaining the full history, whereas InfiniPot, EpiCache, FlowKV, and stateful eviction compress or evict historical KV states (Kim et al., 2024; 2026b; Liu et al., 2025; Poudel, 2025); MemDecay and AgentKV add region- and phase-aware policies (Matam & Kim, 2026; Liu et al., 2026b). For VLMs, MetaCompress (Wang et al., 2026) reuses prompt-agnostic compressed visual representations across turns, while SparseVILA (Khaki et al., 2025) combines prefill pruning with query-aware retrieval but keeps unselected visual entries cached. TrimKV and DBTrimKV (Bui et al., 2026b;a), the closest learned methods, predict retention coefficients at insertion and update age-dependent eviction scores during generation; DBTrimKV additionally allocates budget across layers, heads, and modalities. CarryKV instead leaves the cache untouched during a response, re-selects the retained cache at each turn boundary from the current resident state, and physically compacts KV blocks; it is trained on only 100 dialogues with supervision from the following turn.

3 PRELIMINARIES

3.1 MULTI-TURN VLM INFERENCE

We consider a dialogue of T turns with a VLM comprising L decoder layers. Each turn t consists of a user input x_t, containing text and optionally images, followed by a textual response y_t. With the full dialogue history, the model generates each response from the distribution p_{full}:

$$y_t \sim p_{\text{full}}\big(\cdot \mid x_{\le t}, y_{<t}\big). \tag{1}$$

We consider stateful serving, in which KV entries are reused across turns rather than recomputed (Gao et al., 2024; Kwon et al., 2023). At turn t, the model prefills only the new input x_t and autoregressively generates y_t, extending the existing cache during both stages. Without compression, this adds $|x_t| + |y_t|$ entries per layer, where lengths count tokens after multimodal tokenization and prompt formatting. The accumulation of textual and visual entries therefore increases memory consumption as the dialogue proceeds.

3.2 TURN-BOUNDARY KV CACHE COMPRESSION

Prior work uses KV cache eviction to control memory consumption (Bui et al., 2026b;a). Let $\mathcal{C}_t = \{\mathcal{C}_t^\ell\}_{\ell=1}^L$ denote the resident cache at the end of turn t, before compression, and let $\widehat{\mathcal{C}}_t$ denote its compressed counterpart. Each entry corresponds to one token position and contains its keys and values across all KV heads in that layer, with one shared retention decision per entry. Writing Δ_t^ℓ for the entries produced while prefilling x_t and generating y_t, the cache evolves as

$$\mathcal{C}_t^\ell = \widehat{\mathcal{C}}_{t-1}^\ell \cup \Delta_t^\ell, \qquad \ell = 1, \dots, L. \tag{2}$$

Compression policies differ in when eviction decisions are made. We illustrate two timings for historical-cache compression. Post-input compression is applied after x_t becomes available, allowing the policy to condition its decision on the new input:

$$\widehat{\mathcal{C}}_{t-1} = \Pi(\mathcal{C}_{t-1}; x_t, B), \qquad y_t \sim p_{\text{keep}}\big(\cdot \mid x_t, \widehat{\mathcal{C}}_{t-1}\big), \tag{3}$$

where Π is a cache eviction policy, B is the cache budget, and p_{keep} is the same VLM's response distribution conditioned on the retained cache. In contrast, turn-boundary compression is applied immediately after y_{t-1} is completed and before x_t arrives:

$$\widehat{\mathcal{C}}_{t-1} = \Pi(\mathcal{C}_{t-1}; B), \qquad y_t \sim p_{\text{keep}}\big(\cdot \mid x_t, \widehat{\mathcal{C}}_{t-1}\big). \tag{4}$$

These expressions isolate the timing of historical-cache compression; whether the new input itself is compressed depends on the specific policy.

We focus on turn-boundary compression, requiring eviction decisions to be made without access to the next query. We use a per-layer budget B, with retained token positions shared across all KV heads within each layer (Kim et al., 2026a):

$$\widehat{\mathcal{C}}_t^\ell \subseteq \mathcal{C}_t^\ell, \qquad \big|\widehat{\mathcal{C}}_t^\ell\big| = \min\big(B, |\mathcal{C}_t^\ell|\big), \qquad \forall \ell. \tag{5}$$

This budget applies at turn boundaries; the cache may grow within a turn as new tokens are processed. Evicted entries are not recovered in subsequent turns.

4 CARRYKV: KV CACHE EVICTION ACROSS TURNS

4.1 OVERVIEW

We introduce CarryKV, a turn-boundary KV cache eviction method that learns to preserve information for subsequent dialogue turns. As Figure 3 shows, it consists of two modules. *Structure-Aware Encoding (SAE)* represents each entry's content and structural attributes for selection. *Sketch-Conditioned Selection (SCS)* retains protected entries and a random subset, summarizes the sampled entries into a random sketch shared by all layers, and scores the remaining candidates conditioned on this sketch to fill the remaining budget. We jointly train both modules through future-turn distillation, minimizing the KL divergence between the frozen VLM's response distributions under full and retained caches on subsequent turns; supervision comes from future responses, but each retention decision is conditioned only on the history available at its boundary.

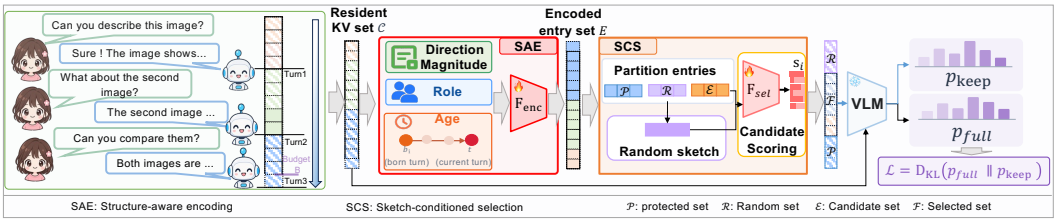


Figure 3: Overview of CarryKV. At turn boundaries with $|\mathcal{C}| > B$, SAE encodes resident entries from content and structural attributes. SCS retains protected and random entries, sketches the random subset, and fills the remaining budget by sketch-conditioned scores under an image floor. The frozen VLM supervises selection via future-turn KL between full- and retained-cache predictions.

4.2 Structure-Aware Encoding

Motivated by prior KV eviction methods (Bui et al., 2026b; Wan et al., 2024; Kim et al., 2026b), we combine content, temporal, and structural signals to construct a representation for retention decisions. As shown in Figure 4, each resident entry is characterized by its KV direction and magnitude, age, and role, which SAE encodes into a compact representation for selection.

For a fixed turn t and layer ℓ, we omit these indices and write an entry's keys and values as $\mathbf{k}_i, \mathbf{v}_i \in \mathbb{R}^{H\times D}$, with H KV heads of dimension D sharing one retention decision.

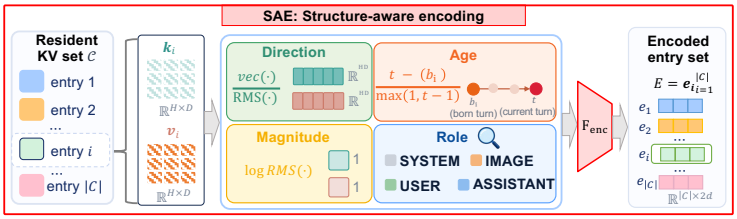


Figure 4: SAE encodes each resident KV entry from its content and structural attributes into a compact representation for selection.

To separate direction from scale, we normalize keys and values by their root mean square (RMS) and retain log RMS to preserve magnitude while compressing its range. For an entry introduced at turn b_i, with turns indexed from 1, we normalize its age by preceding turns:

$$a_i = \frac{t - b_i}{\max(1,\, t-1)} \in [0, 1]. \tag{6}$$

The resulting feature vector for each entry is

$$\boldsymbol{\phi}_i = \left[\frac{\mathrm{vec}(\mathbf{k}_i)}{\mathrm{RMS}(\mathbf{k}_i)}, \frac{\mathrm{vec}(\mathbf{v}_i)}{\mathrm{RMS}(\mathbf{v}_i)}, \log \mathrm{RMS}(\mathbf{k}_i), \log \mathrm{RMS}(\mathbf{v}_i), a_i \right] \in \mathbb{R}^{2HD+3}, \tag{7}$$

where vec flattens an array and RMS is computed over all HD elements.

SAE consists of an encoder f_{enc} shared across layers, learned layer embeddings $\mathbf{u}_\ell \in \mathbb{R}^d$, and learned role embeddings $\mathbf{r}_i \in \mathbb{R}^d$. Each entry carries one of four role labels, `SYSTEM` (system and chat-template tokens), `IMAGE` (visual tokens), `USER` (user-query text), or `ASSISTANT` (model-response text), from which $\mathbf{r}_i$ is looked up. The same labels, together with turn and token-position metadata, also identify protected entries in SCS. SAE maps each entry to

$$F_{\mathrm{enc}}(i) = \mathbf{e}_i = \left[\tilde{\mathbf{e}}_i,\ \mathbf{r}_i \right] \in \mathbb{R}^{2d}, \qquad \tilde{\mathbf{e}}_i = f_{\mathrm{enc}}(\boldsymbol{\phi}_i) + \mathbf{u}_\ell \in \mathbb{R}^d. \tag{8}$$

The encoder is a two-layer MLP with input layer normalization and a GELU activation; $\mathbf{u}_\ell$ supplies layer identity to the shared selector. The role embedding bypasses the encoder: it is attached to the content encoding $\tilde{\mathbf{e}}_i$ rather than added to it, so that the sketch in SCS can be built from $\tilde{\mathbf{e}}_i$ alone while the selector receives both parts of $\mathbf{e}_i$.

4.3 Sketch-Conditioned Selection

Given the SAE outputs $\mathbf{e}_i = [\tilde{\mathbf{e}}_i, \mathbf{r}_i]$, SCS partitions the resident cache and builds the retained cache from three disjoint parts, $\widehat{\mathcal{C}} = \mathcal{P} \cup \mathcal{R} \cup \mathcal{F}$ (Figure 5): the protected set $\mathcal{P}$ kept by rule, the random set $\mathcal{R}$ kept at random, and the selected set $\mathcal{F}$ kept by score. We omit the turn and layer indices when a single layer at one boundary is discussed, and assume $|\mathcal{C}| > B$.

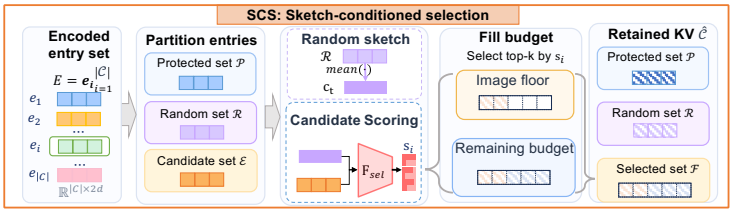


Figure 5: SCS retains a fixed budget of entries: it first keeps the protected set $\mathcal{P}$ by rule and a random subset $\mathcal{R}$, summarizes $\mathcal{R}$ into a sketch, scores the remaining candidates $\mathcal{E}$ conditioned on the sketch, and fills the image floor and the remaining capacity by score.

Structural constraints. Motivated by the importance of structural protection (Garcia, 2026), we retain a protected set $\mathcal{P} \subseteq \mathcal{C}$ containing the system prefix, the text of the current user input, and the last n_{rec} tokens of the current response, with $|\mathcal{P}| \leq B$. In addition, each layer retains at least B_{img} image entries, where the image floor B_{img} is derived from the budget and the number of resident image entries. Both constraints are enforced during selection; their values are given in Appendix C.1, and Section 5.4 examines their effects on response quality.

Random sketch. From the unprotected entries, we draw a random subset without replacement:

$$\mathcal{R} = \text{Random}(\mathcal{C} \setminus \mathcal{P},\, n_{\text{rand}}), \qquad |\mathcal{R}| = n_{\text{rand}} \leq \lfloor \rho B \rfloor, \tag{9}$$

where the sampling ratio ρ may differ between training and inference (ρ_{train}, ρ_{infer}). $\mathcal{R}$ is drawn afresh at every boundary and retained as is; during training, the changing draws expose the selector to different starting sets. We average the sampled encodings across all layers:

$$\mathbf{c}_t = \frac{1}{\sum_{\ell=1}^{L} |\mathcal{R}_t^\ell|} \sum_{\ell=1}^{L} \sum_{j \in \mathcal{R}_t^\ell} \tilde{\mathbf{e}}_{t,j}^\ell \in \mathbb{R}^d, \tag{10}$$

which we call the random sketch: a fixed-dimensional summary of the randomly retained entries at boundary t. The role embeddings are not pooled. When no entries are sampled, we set $\mathbf{c}_t = \mathbf{0}$.

Candidate scoring. The remaining entries form the candidate set $\mathcal{E} = \mathcal{C} \setminus (\mathcal{P} \cup \mathcal{R})$. Each candidate $i \in \mathcal{E}$ receives the score

$$s_i = F_{\text{sel}}(\mathbf{e}_i, \mathbf{c}_t) = f_{\text{sel}}\left([\mathbf{e}_i,\, \mathbf{c}_t,\, \tilde{\mathbf{e}}_i \odot \mathbf{c}_t]\right), \tag{11}$$

where $F_{\text{sel}} : \mathbb{R}^{2d} \times \mathbb{R}^d \to \mathbb{R}$ takes the entry and the sketch, and $f_{\text{sel}} : \mathbb{R}^{4d} \to \mathbb{R}$ is a two-layer MLP with input layer normalization, shared across layers. Conditioning on $\mathbf{c}_t$ makes each score depend on both the candidate and the sampled entries already reserved for retention, while $\tilde{\mathbf{e}}_i \odot \mathbf{c}_t$ provides a feature-wise interaction between them. Thus, SCS can adapt its selection to different partially retained cache states. Scores are computed once per boundary and are not updated during filling.

Budget filling. Let $\mathcal{I}$ denote the image entries of the layer, and let $\text{Top}_k(\mathcal{A}; s)$ return the k highest-scoring entries of $\mathcal{A}$ under s. After $\mathcal{P}$ and $\mathcal{R}$, the budget admits $m = B - |\mathcal{P}| - |\mathcal{R}|$ further entries, which we select in two stages. The first stage meets the image floor: with m_{img} image entries still missing after the random draw, it selects

$$\mathcal{F}_{\text{img}} = \text{Top}_{m_{\text{img}}}(\mathcal{E} \cap \mathcal{I}; s), \qquad m_{\text{img}} = \max(0,\, B_{\text{img}} - |\mathcal{R} \cap \mathcal{I}|). \tag{12}$$

The second stage selects the highest-scoring remaining candidates of any role until the budget is filled, which completes the retained cache $\widehat{\mathcal{C}}$:

$$\mathcal{F} = \mathcal{F}_{\text{img}} \cup \text{Top}_{m - m_{\text{img}}}(\mathcal{E} \setminus \mathcal{F}_{\text{img}}; s), \qquad \widehat{\mathcal{C}} = \mathcal{P} \cup \mathcal{R} \cup \mathcal{F}. \tag{13}$$

The two stages separate the modality constraint from learned selection. The first stage only supplies the image entries needed to satisfy B_{img}, counting those already present in $\mathcal{R}$. Once the floor is met, the remaining capacity is allocated solely by the learned scores, regardless of role.

4.4 Training and Inference

Future-turn objective. We train SAE and SCS jointly so that responses under the retained cache match those under the full cache. For each training dialogue, the frozen VLM serves as both teacher and student, teacher-forced on the reference responses. The teacher reads the full-history cache $\mathcal{C}_{t-1}^{\text{full}}$,

without eviction, whereas the student reads $\widehat{\mathcal{C}}_{t-1}(\theta)$, retained by SCS at each preceding boundary, where θ contains the parameters of SAE and SCS. The loss is the mean full-vocabulary forward KL divergence between the two response distributions:

$$\mathcal{L}(\theta) = \frac{1}{\sum_{t=2}^{T} |y_t|} \sum_{t=2}^{T} \sum_{j=1}^{|y_t|} D_{\mathrm{KL}}\Big(p_{\mathrm{full}}\big(\cdot \mid x_t, y_{t,<j}, \mathcal{C}_{t-1}^{\mathrm{full}}\big) \,\Big\|\, p_{\mathrm{keep}}\big(\cdot \mid x_t, y_{t,<j}, \widehat{\mathcal{C}}_{t-1}(\theta)\big)\Big) . \quad (14)$$

Each response loss is backpropagated only through the immediately preceding retention decision; earlier eviction masks are treated as constants but remain in effect for the rest of the dialogue. Each boundary thus receives direct supervision from the following response only, while longer-horizon effects enter indirectly through the cache that later boundaries inherit.

Optimization. Because the selection in Equation 13 is discrete, the loss provides no gradient to the scores. We therefore train through a survival gate g_i that gates candidate entries in attention: the forward pass uses the same hard selection $\mathcal{F}$ as inference, while the backward pass replaces it with a soft relaxation z_i of the scores, so that gradients reach f_{sel} and f_{enc}:

$$g_i = \mathbf{1}[i \in \mathcal{F}] + z_i - \mathrm{sg}(z_i), \qquad i \in \mathcal{E}, \quad (15)$$

where sg is the stop-gradient operator. Appendix C.2 defines the relaxation z_i, its gradient, and the treatment of protected and randomly sampled entries.

Inference. At inference, CarryKV runs as a KV policy inside vLLM (Kwon et al., 2023). When a response ends, by end-of-sequence token or length limit, and before the next query arrives, SAE and SCS select entries from the resident paged cache of every layer; the retained cache is compacted and the freed blocks are returned to the pool. Eviction thus physically reduces resident memory while retained entries keep their original positions (Appendix E).

5 Experiments

5.1 Experimental Settings

Training and runtime. For each VLM, CarryKV is trained once at B=2048 on 100 dialogues sampled from MMDU-45K (Liu et al., 2024b), taking under 25 minutes on a single NVIDIA A100 GPU, and is reused across budgets and benchmarks without retraining. The selector has 161K parameters on Qwen2.5-VL-7B and 95K on GLM-4.6V-Flash-9B. We set ρ_{train}=0.25 and ρ_{infer}=0.10; the optimizer and the remaining hyperparameters are listed in Appendix C.2.

Models and baselines. We evaluate on Qwen2.5-VL-7B (Bai et al., 2025) and GLM-4.6V-Flash-9B (GLM-V Team et al., 2025). CarryKV is compared with Full KV and four cache methods spanning the two retention regimes: RocketKV-MT (Behnam et al., 2025) and SparseVILA (Khaki et al., 2025) sparsify decoding while retaining the full history, whereas TriAttention (Mao et al., 2026) and EpiCache (Kim et al., 2026b) evict from the resident cache. All baselines run in the same vLLM runtime with the official configurations listed in Appendix B.4.

Benchmarks and protocol. We evaluate on four multi-turn multimodal benchmarks, each scored by its official protocol with Qwen3-VL-32B-Instruct as the judge: MMDU (Liu et al., 2024b) (10-point), MMRC (Xue et al., 2025) (5-point), MultiVerse (Lee et al., 2025) (0-100), and ConvBench (Liu et al., 2024a) (win rate against the reference); Appendix B.1 gives further details. Each dialogue is served as one persistent request, in which each response's end-of-sequence token triggers the boundary and the history consists of the model's responses rather than the references, except on MMRC, whose official protocol fixes the history; Appendix B.2 details the serving protocol.

5.2 Main Results

Quality results. As shown in Table 1, CarryKV maintains competitive generation quality across both models and four multi-turn benchmarks; results at all budgets are provided in Appendix D.1. Under relaxed budgets, CarryKV performs comparably to Full KV. Under tight budgets, CarryKV maintains strong performance on MultiVerse, trailing Full KV by 0.57 and 0.23 points on Qwen2.5-VL-7B and GLM-4.6V-Flash-9B, respectively, while outperforming TriAttention on both models.

Table 1: Generation quality of CarryKV and baselines on Qwen2.5-VL-7B and GLM-4.6V-Flash-9B under a relaxed budget and a tight budget. ConvBench is reported as win rate in %; the other benchmarks use their official scales. CarryKV is the same trained checkpoint at every budget.

Budget	Method	Qwen2.5-VL-7B				GLM-4.6V-Flash-9B			
		MMDU	MultiVerse	MMRC	ConvBench	MMDU	MultiVerse	MMRC	ConvBench
–	Full KV	6.58	49.47	2.62	13.20%	7.13	54.97	3.34	29.40%
Relaxed	RocketKV-MT* (ICML'25)	**6.62**	37.24	2.29	10.80%	7.16	**55.42**	**3.31**	28.20%
	SparseVILA* (ICCV'25)	6.08	42.40	2.48	8.50%	6.66	47.63	3.22	18.50%
	EpiCache (ICML'26)	4.51	40.13	1.70	4.60%	5.54	43.32	2.35	9.00%
	TriAttention (ICML'26)	6.55	48.76	**2.58**	**12.60%**	7.14	53.97	**3.31**	**28.60%**
	CarryKV (ours)	6.60	**49.71**	**2.58**	12.00%	**7.22**	55.24	**3.31**	28.30%
Tight	RocketKV-MT* (ICML'25)	4.46	21.24	1.40	6.80%	**7.05**	**55.06**	**3.31**	27.90%
	SparseVILA* (ICCV'25)	5.97	41.95	2.10	8.70%	6.60	47.29	2.82	18.80%
	EpiCache (ICML'26)	4.30	40.00	1.69	4.90%	5.38	43.45	2.33	8.50%
	TriAttention (ICML'26)	**6.18**	48.05	**2.42**	11.60%	6.78	52.44	3.18	26.00%
	CarryKV (ours)	**6.18**	**48.90**	2.25	**11.90%**	6.96	54.74	3.12	**28.90%**

*Sparse retrieval without eviction. Relaxed: B=4096 on MMDU, 1024 elsewhere. Tight: B=1024 on MMDU, 512 elsewhere.

Table 2: Serving efficiency on MMDU at B=1024: peak throughput over $C \in \{16, 32, 64, 96\}$, TTFT at C=16, and wall-clock at $C_{\max}$, the largest C that completes without running out of GPU memory. Parentheses: speedup over Full KV. Wall-clock also reflects generated length.

Method	Qwen2.5-VL-7B				GLM-4.6V-Flash-9B			
	$C_{\max}$	Throughput (tok/s) ↑	TTFT (s) ↓	Wall-clock (s) ↓	$C_{\max}$	Throughput (tok/s) ↑	TTFT (s) ↓	Wall-clock (s) ↓
Full KV	32	727	0.226	2,606	64	759	0.232	1,767
RocketKV-MT (ICML'25)	32	586 (0.81×)	0.245	1,918 (1.36×)	32	647 (0.85×)	0.262	2,028 (0.87×)
SparseVILA (ICCV'25)	32	673 (0.93×)	0.344	2,594 (1.00×)	96	624 (0.82×)	0.414	2,108 (0.84×)
EpiCache (ICML'26)	96	97 (0.13×)	0.601	12,999 (0.20×)	96	127 (0.17×)	0.537	12,731 (0.14×)
TriAttention (ICML'26)	96	1,147 (1.58×)	0.180	1,608 (1.62×)	96	974 (1.28×)	0.191	1,392 (1.27×)
CarryKV (ours)	96	**1,272** (1.75×)	**0.175**	**1,332** (1.96×)	96	**1,039** (1.37×)	**0.183**	**1,356** (1.30×)

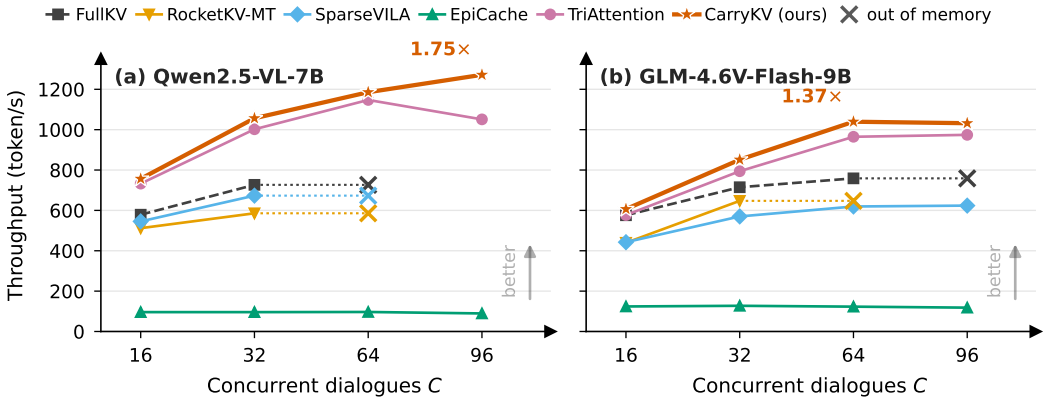


Figure 6: Throughput versus concurrency C on MMDU at B=1024 with (a) Qwen2.5-VL-7B and (b) GLM-4.6V-Flash-9B. CarryKV achieves the highest throughput at every tested C on both models. Crosses mark runs that run out of GPU memory before completing the sweep.

On MMDU and ConvBench it matches or exceeds the best evicting method, with modest degradation on MMRC. Appendix B.3 reports paired bootstrap confidence intervals; Table 8 marks the differences from TriAttention whose interval excludes zero. Appendix D.2 presents case studies of multi-turn dialogues, showing where the eviction baselines lose the entries a later turn needs.

Serving efficiency. To obtain stable throughput measurements, we serve the MMDU dialogues twice on one A100 at B=1024 with concurrency $C \in \{16, 32, 64, 96\}$. End-to-end measurements include cache scoring and compaction overhead. Table 2 and Figure 6 show that CarryKV achieves the highest throughput at every tested concurrency on both models, with peak throughput reaching 1.75× and 1.37× that of Full KV on Qwen2.5-VL-7B and GLM-4.6V-Flash-9B, respectively. Most of the gain over Full KV comes from cache block reclamation, which raises the largest concurrency that completes the sweep from 32 to 96 (the highest tested) on Qwen2.5-VL-7B and from 64 to 96 on GLM-4.6V-Flash-9B, while the shorter history reduces cache access during decoding. At matched

Table 3: Module ablation on MMDU (B=1024). Each variant is retrained.

SAE	SCS	Qwen	GLM
×	×	6.13	6.82
✓	×	6.11 (−0.02)	6.89 (+0.07)
×	✓	6.16 (+0.03)	6.89 (+0.07)
✓	✓	6.18 (+0.05)	6.96 (+0.14)

Table 4: Effect of the inference sampling ratio on Qwen2.5-VL-7B. Shading marks the default.

ρ_{infer}	MultiVerse	MMDU
0	48.94	6.252
0.10	48.90	6.177
0.25	48.92	6.206
0.50	48.92	6.206
Random-fill	48.81	5.928

Random-fill in Table 4 uses random scoring under the same structural constraints.

Table 5: Parameter sensitivity on Qwen2.5-VL-7B (MultiVerse at B=512, MMDU at B=1024). Shading marks the default; Max protects as much of the latest response as fits the budget.

	ρ_{train}			Image floor				n_{rec}				
Setting	.1	.25	.5	0	$B/8$	$5B/16$	$B/2$	0	64	128	256	Max
MultiVerse	48.64	48.90	48.93	48.58	48.96	48.90	48.90	49.24	49.28	48.96	48.90	47.58
MMDU	6.187	6.177	6.216	6.212	6.190	6.177	6.214	6.251	6.302	6.274	6.177	5.824

concurrency, CarryKV achieves lower mean TTFT and TPOT than Full KV and every compression baseline at every C at which that baseline completes; latency results appear in Appendix E. Sparse retrieval baselines retain the full cache and do not exceed Full KV's throughput, while other eviction methods trail CarryKV. Together with the quality results, these findings demonstrate practical serving gains with competitive response quality.

5.3 Ablation Study

We ablate SAE and SCS, retraining each variant and evaluating on MMDU at B=1024 (Table 3). Without SAE, we remove the encoder and layer embeddings, use the raw features $\boldsymbol{\phi}$ of Eq. 7 for both entries and their sketch, and adjust the selector input to $3(2HD+3)+d$ dimensions. Without SCS, we zero the sketch and interaction terms while retaining the random subset. Role embeddings and structural constraints are unchanged. Using either module alone does not reach the full score. The full method exceeds the variant without either module by 0.14 points on GLM, where the paired bootstrap interval of Appendix B.3 excludes zero, and by a sign-consistent 0.05 points on Qwen, within noise. These results support combining SAE with SCS.

5.4 Sensitivity Analysis

As shown in Tables 4 and 5, we vary one parameter at a time on Qwen2.5-VL-7B: the sampling ratios at training and inference, the minimum image budget, and the answer protection length (0–256 tokens). All inference-time sweeps reuse the same checkpoint; only the training sampling ratio requires retraining. Across these settings, excluding Random-fill and the maximum-tail setting discussed below, scores deviate from the default by at most 0.38 points on MultiVerse and 0.125 points on MMDU, so CarryKV is robust to these choices. In particular, setting ρ_{infer}=0 matches the default, which shows that the random subset acts as a training device and can be dropped at deployment, making the policy deterministic. Random-fill, which keeps every structural constraint and replaces the learned scores by random ones, lowers MMDU by 0.25 points. Protecting the entire latest response (Max) lowers both benchmarks, since it leaves little capacity for other history. Appendix D.3 provides the complete results and experimental settings.

6 Conclusion

We presented CarryKV, a learned turn-boundary KV cache eviction method for multi-turn VLMs. CarryKV combines Structure-Aware Encoding and Sketch-Conditioned Selection to select resident KV entries, and learns retention decisions through future-turn distillation on the following response. Training uses only 100 dialogues and under 25 minutes on one A100, with one checkpoint reused across budgets and benchmarks. Integrated into vLLM, CarryKV physically reclaims evicted KV blocks. Across two VLMs and four multi-turn benchmarks, CarryKV maintains competitive quality under tight cache budgets while reaching 1.37–1.75× Full KV's peak throughput, demonstrating an effective quality–efficiency trade-off for persistent multi-turn VLM serving.

AI USE STATEMENT

In this work, we used generative AI tools to assist with translation. We have not used generative AI tools for things like generating data and supporting qualitative and thematic data analysis, and the rest of the required disclosure tasks like mathematical proofs are not applicable to this work. Additionally, we used generative AI tools for polishing our language. We have reviewed all AI-assisted work. We checked LLM-generated research ideas for potential plagiarism through a manual literature survey. LLM-generated code was verified and tested for correctness by all of the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our paper does not include any human subjects, practices to data set releases, potentially harmful insights, methodologies and applications, potential conflicts of interest, discrimination/bias/fairness concerns, privacy and security issues, legal compliance, and research integrity issues.

REPRODUCIBILITY STATEMENT

The experimental settings, including the training data, models, baselines, and benchmarks, are described in Section 5.1, with the full benchmark and data details in Appendix B.1. Training details and hyperparameters are given in Section 4.4 and Appendix C.2. The serving and judging protocols are described in Appendices B.2 and B.3, and the efficiency measurements in Appendix E. We will release all our code and useful artifacts once the paper is accepted.

REFERENCES

Asaad Althoubi. DistillCache: KL-guided adaptive KV-cache eviction for memory-efficient LLM inference. *arXiv preprint arXiv:2608.08878*, 2026. URL https://arxiv.org/abs/2608.08878.

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. *arXiv preprint arXiv:2502.13923*, 2025.

Payman Behnam, Yaosheng Fu, Ritchie Zhao, Po-An Tsai, Zhiding Yu, and Alexey Tumanov. RocketKV: Accelerating long-context LLM inference via two-stage KV cache compression. In *ICML*, 2025.

Ngoc Bui, Hieu Trung Nguyen, Arman Cohan, and Rex Ying. Make each token count: Towards improving long-context performance with KV cache eviction. *arXiv preprint arXiv:2605.09649*, 2026a.

Ngoc Bui, Shubham Sharma, Simran Lamba, Saumitra Mishra, and Rex Ying. Cache what lasts: Token retention for memory-bounded KV cache in LLMs. In *ICLR*, 2026b.

Zefan Cai, Yichi Zhang, Bofei Gao, Yuliang Liu, Yucheng Li, Tianyu Liu, Keming Lu, Wayne Xiong, Yue Dong, Junjie Hu, and Wen Xiao. PyramidKV: Dynamic KV cache compression based on pyramidal information funneling. In *COLM*, 2025.

Vivek Chari, Guanghui Qin, and Benjamin Van Durme. KV-Distill: Nearly lossless learnable context compression for LLMs. *arXiv preprint arXiv:2503.10337*, 2025.

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 *ECCV*, 2024a.

Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, Bin Li, Ping Luo, Tong Lu, Yu Qiao, and Jifeng Dai. InternVL: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *CVPR*, 2024b.

Alessio Devoto, Maximilian Jeblick, and Simon Jégou. Expected attention: KV cache compression by estimating attention from future queries distribution. *arXiv preprint arXiv:2510.00636*, 2025.

Zican Dong, Peiyu Liu, Junyi Li, Zhipeng Chen, Han Peng, Shuo Wang, and Wayne Xin Zhao. ForesightKV: Optimizing KV cache eviction for reasoning models by learning long-term contribution. In *ICML*, 2026.

Bin Gao, Zhuomin He, Puru Sharma, Qingxuan Kang, Djordje Jevdjic, Junbo Deng, Xingkun Yang, Zhou Yu, and Pengfei Zuo. Cost-efficient large language model serving for multi-turn conversations with CachedAttention. In *USENIX ATC*, 2024.

Gabriel Garcia. Protection is (nearly) all you need: Structural protection dominates scoring in globally capped KV eviction. *arXiv preprint arXiv:2605.18053*, 2026.

GLM-V Team, Wenyi Hong, Wenmeng Yu, Xiaotao Gu, Guo Wang, Guobing Gan, Haomiao Tang, Jiale Cheng, Ji Qi, Junhui Ji, Lihang Pan, Shuaiqi Duan, Weihan Wang, Yan Wang, Yean Cheng, Zehai He, Zhe Su, Zhen Yang, Ziyang Pan, Aohan Zeng, Baoxu Wang, Bin Chen, Boyan Shi, Changyu Pang, Chenhui Zhang, Da Yin, Fan Yang, Guoqing Chen, Haochen Li, Jiale Zhu, Jiali Chen, Jiaxing Xu, Jiazheng Xu, Jing Chen, Jinghao Lin, Jinhao Chen, Jinjiang Wang, Junjie Chen, Leqi Lei, Letian Gong, Leyi Pan, Mingdao Liu, Mingde Xu, Mingzhi Zhang, Qinkai Zheng, Ruiliang Lyu, Shangqin Tu, Sheng Yang, Shengbiao Meng, Shi Zhong, Shiyu Huang, Shuyuan Zhao, Siyan Xue, Tianshu Zhang, Tianwei Luo, Tianxiang Hao, Tianyu Tong, Wei Jia, Wenkai Li, Xiao Liu, Xiaohan Zhang, Xin Lyu, Xinyu Zhang, Xinyue Fan, Xuancheng Huang, Yadong Xue, Yanfeng Wang, Yanling Wang, Yanzi Wang, Yifan An, Yifan Du, Yiheng Huang, Yilin Niu, Yiming Shi, Yu Wang, Yuan Wang, Yuanchang Yue, Yuchen Li, Yusen Liu, Yutao Zhang, Yuting Wang, Yuxuan Zhang, Zhao Xue, Zhengxiao Du, Zhenyu Hou, Zihan Wang, Peng Zhang, Debing Liu, Bin Xu, Juanzi Li, Minlie Huang, Yuxiao Dong, and Jie Tang. GLM-4.5V and GLM-4.1V-Thinking: Towards versatile multimodal reasoning with scalable reinforcement learning. *arXiv preprint arXiv:2507.01006*, 2025. URL `https://arxiv.org/abs/2507.01006v6`. Version 6 (January 2026) includes GLM-4.6V-Flash (9B).

Yuxiang Huang, Binhang Yuan, Xu Han, Chaojun Xiao, and Zhiyuan Liu. Locret: Enhancing eviction in long-context LLM inference with trained retaining heads on consumer-grade devices. *arXiv preprint arXiv:2410.01805*, 2024.

Samir Khaki, Junxian Guo, Jiaming Tang, Shang Yang, Yukang Chen, Konstantinos N. Plataniotis, Yao Lu, Song Han, and Zhijian Liu. SparseVILA: Decoupling visual sparsity for efficient VLM inference. In *ICCV*, 2025.

Hyungmin Kim, Minsoo Kim, Hongseok Kim, and Jungwook Choi. Tangram: Unlocking non-uniform KV cache compression for efficient multi-turn LLM serving. *arXiv preprint arXiv:2606.06302*, 2026a.

Jang-Hyun Kim, Jinuk Kim, Sangwoo Kwon, Jae W. Lee, Sangdoo Yun, and Hyun Oh Song. KVzip: Query-agnostic KV cache compression with context reconstruction. In *NeurIPS*, 2025.

Minsoo Kim, Kyuhong Shim, Jungwook Choi, and Simyung Chang. InfiniPot: Infinite context processing on memory-constrained LLMs. In *EMNLP*, 2024.

Minsoo Kim, Arnav Kundu, Han-Byul Kim, Richa Dixit, and Minsik Cho. EpiCache: Episodic KV cache management for long-term conversation on resource-constrained environments. In *ICML*, 2026b.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *SOSP*, 2023.

Young-Jun Lee, Byung-Kwan Lee, Jianshu Zhang, Yechan Hwang, Byungsoo Ko, Han-Gyu Kim, Dongyu Yao, Xuankun Rong, Eojin Joo, Seung-Ho Han, Bowon Ko, and Ho-Jin Choi. Multi-Verse: A multi-turn conversation benchmark for evaluating large vision and language models. In *ICCV*, 2025.

Yucheng Li, Huiqiang Jiang, Qianhui Wu, Xufang Luo, Surin Ahn, Chengruidong Zhang, Amir H. Abdi, Dongsheng Li, Jianfeng Gao, Yuqing Yang, and Lili Qiu. SCBench: A KV cache-centric analysis of long-context methods. In *ICLR*, 2025.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *NeurIPS*, 2024.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *NeurIPS*, 2023.

Jizhihui Liu, Guangdao Zhu, Feiyi Du, Niu Lian, Jun Li, Bin Chen, Weili Guan, and Yaowei Wang. HiPrune: Hierarchical attention for efficient token pruning in vision-language models. In *Findings of ACL*, 2026a.

Shuo Liu, Kaining Ying, Hao Zhang, Yue Yang, Yuqi Lin, Tianle Zhang, Chuanhao Li, Yu Qiao, Ping Luo, Wenqi Shao, and Kaipeng Zhang. ConvBench: A multi-turn conversation evaluation benchmark with hierarchical ablation capability for large vision-language models. In *NeurIPS*, 2024a.

Taowen Tony Liu, Jeffrey T. H. Wong, Can Xiao, Bowen Yang, Hao Mark Chen, and Yiren Zhao. AgentKV: Phase-aware KV eviction for agentic LLMs. *arXiv preprint arXiv:2609.14872*, 2026b.

Xiang Liu, Hong Chen, Xuming Hu, and Xiaowen Chu. FlowKV: Enhancing multi-turn conversational coherence in LLMs via isolated key-value cache management. *arXiv preprint arXiv:2505.15347*, 2025.

Ziyu Liu, Tao Chu, Yuhang Zang, Xilin Wei, Xiaoyi Dong, Pan Zhang, Zijian Liang, Yuanjun Xiong, Yu Qiao, Dahua Lin, and Jiaqi Wang. MMDU: A multi-turn multi-image dialog understanding benchmark and instruction-tuning dataset for LVLMs. In *NeurIPS*, 2024b.

Weian Mao, Xi Lin, Wei Huang, Yuxin Xie, Tianfu Fu, Bohan Zhuang, Song Han, and Yukang Chen. TriAttention: Efficient long reasoning with trigonometric KV compression. In *ICML*, 2026.

Venkatesha Matam and Keon Kim. MemDecay: Region-aware KV cache eviction for efficient LLM agent inference. *arXiv preprint arXiv:2607.10582*, 2026.

Luca Moschella, Laura Manduchi, and Ozan Sener. Learning to evict from key-value cache. In *ICML*, 2026.

Pratik Poudel. Stateful KV cache management for LLMs: Balancing space, time, accuracy, and positional fidelity. *arXiv preprint arXiv:2511.04686*, 2025.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. LLaVA-PruMerge: Adaptive token reduction for efficient large multimodal models. In *ICCV*, 2025.

Jiaming Tang, Yilong Zhao, Kan Zhu, Guangxuan Xiao, Baris Kasikci, and Song Han. QUEST: Query-aware sparsity for efficient long-context LLM inference. In *ICML*, 2024.

Dezhan Tu, Danylo Vashchilenko, Yuzhe Lu, and Panpan Xu. VL-Cache: Sparsity and modality-aware KV cache compression for vision-language model inference acceleration. *ICLR*, 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 EMNLP*, 2024.

Zhongwei Wan, Hui Shen, Xin Wang, Che Liu, Zheda Mai, and Mi Zhang. MEDA: Dynamic KV cache allocation for efficient multimodal long-context inference. In *NAACL*, 2025.

Yi Wang, Haofei Zhang, Qihan Huang, Anda Cao, Gongfan Fang, Wei Wang, Xuan Jin, Jie Song, Mingli Song, and Xinchao Wang. Rethinking token reduction for large vision-language models. In *CVPR*, 2026.

Haochen Xue, Feilong Tang, Ming Hu, Yexin Liu, Qidong Huang, Yulong Li, Chengzhi Liu, Zhongxing Xu, Chong Zhang, Chun-Mei Feng, Yutong Xie, Imran Razzak, Zongyuan Ge, Jionglong Su, Junjun He, and Yu Qiao. MMRC: A large-scale benchmark for understanding multimodal large language model in real-world conversation. In *ACL*, 2025.

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 *CVPR*, 2025.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A. Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. SparseVLM: Visual token sparsification for efficient vision-language model inference. In *ICML*, 2025.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *NeurIPS*, 2023.

Enshuai Zhou, Yifan Hao, Chao Wang, Rui Zhang, Di Huang, Jiaming Guo, Xing Hu, Zidong Du, Qi Guo, and Yunji Chen. LKV: End-to-end learning of head-wise budgets and token selection for LLM KV cache eviction. *arXiv preprint arXiv:2605.06676*, 2026.

APPENDIX

Appendix A collects the notation. Appendix B gives the experimental protocol: benchmarks and data, serving and judging, and baselines. Appendix C gives the method details that the main text defers: the retention quotas and the training procedure, gate, and gradient. Appendix D reports the quality results at every budget and per dimension, a case study, and the complete sensitivity sweep. Appendix E gives the serving details and latencies.

A NOTATION AND CONVENTIONS

At a turn boundary, CarryKV maps the resident cache $\mathcal{C}$ to the retained cache $\widehat{\mathcal{C}}$ through its two modules, SAE (Section 4.2) and SCS (Section 4.3):

$$\mathcal{C} \xrightarrow{\text{SAE}} \{\mathbf{e}_i\} \xrightarrow{\text{SCS}} \widehat{\mathcal{C}} = \mathcal{P} \cup \mathcal{R} \cup \mathcal{F}.$$

Table 6 lists the symbols, grouped by where they first appear. Subscripts t and i denote the turn and entry, and superscript ℓ the decoder layer; we omit t and ℓ when a fixed boundary and layer are understood. The sketch $\mathbf{c}_t$ is shared across layers and carries no layer superscript. A cache entry is the keys and values of one token position across all KV heads of one layer, and the budget B counts entries per layer at a turn boundary; the cache may grow within a turn.

Table 6: Notation for cache selection and future-turn distillation.

Symbol	Meaning
Cache and sets	
$\mathcal{C}_t^\ell$	Resident entries at the end of turn t, before that boundary's selection.
$\widehat{\mathcal{C}}_t^\ell$	Retained cache carried into the next turn; $\mathcal{P}_t^\ell \cup \mathcal{R}_t^\ell \cup \mathcal{F}_t^\ell$.
B, B_{eff}	Per-layer budget and effective capacity $\min(B, \lvert\mathcal{C}\rvert)$.
$\mathcal{P}_t^\ell$	Protected set: system prefix, current question text, and the last n_{rec} tokens of the current response, subject to capacity.
$\mathcal{R}_t^\ell$	Random set: uniformly sampled entries outside $\mathcal{P}_t^\ell$; retained directly and used to construct the sketch.
$\mathcal{E}_t^\ell$	Candidate set: scored entries $\mathcal{C}_t^\ell \setminus (\mathcal{P}_t^\ell \cup \mathcal{R}_t^\ell)$.
$\mathcal{F}_t^\ell$	Selected set: learned completion chosen from $\mathcal{E}_t^\ell$, with size $B_{\text{eff}} - \lvert\mathcal{P}_t^\ell\rvert - \lvert\mathcal{R}_t^\ell\rvert$.
$\mathcal{I}$	Image entries in the resident cache of the current layer; turn and layer indices are omitted.
Δ_t^ℓ	Entries added while prefilling the current input and generating its response.
Π	Cache eviction policy; the boundary policy uses the available history and budget.
Entry representation	
$\mathbf{k}_i^\ell$, $\mathbf{v}_i^\ell$	Cached key and value arrays of entry i, each in $\mathbb{R}^{H \times D}$.
$\boldsymbol{\phi}_{t,i}^\ell$	Input features: normalized keys and values, their log RMS magnitudes, and relative age; dimension $2HD + 3$.
$a_{t,i}$, b_i	Relative age $(t - b_i)/\max(1, t-1)$ and the turn at which entry i entered the cache.
$\mathbf{u}_\ell$	Learned layer embedding in $\mathbb{R}^d$.
$\mathbf{r}_i$	Learned role embedding in $\mathbb{R}^d$, looked up from `SYSTEM`, `IMAGE`, `USER`, or `ASSISTANT`; shared by entries of the same role.
$\tilde{\mathbf{e}}_{t,i}^\ell$	Content encoding $f_{\text{enc}}(\boldsymbol{\phi}_{t,i}^\ell) + \mathbf{u}_\ell \in \mathbb{R}^d$; the only part of the SAE output used by the sketch.
$\mathbf{e}_{t,i}^\ell$, F_{enc}	SAE output $[\tilde{\mathbf{e}}_{t,i}^\ell, \mathbf{r}_i] \in \mathbb{R}^{2d}$ and the map that produces it; the selector receives both parts.
f_{enc}	Two-layer entry encoder shared across decoder layers.
Scoring and selection	
$\mathbf{c}_t$	Mean of the sampled content encodings $\tilde{\mathbf{e}}$ across all layers at boundary t; a shared vector in $\mathbb{R}^d$, set to zero if no entries are sampled.

Continued on the next page.

Table 6 continued.

Symbol	Meaning
$s^{\ell}_{t,i}$	Scalar score of a candidate, conditioned on its encoding $\mathbf{e}_i$ (content and role), the random sketch $\mathbf{c}_t$, and their feature-wise interaction $\tilde{\mathbf{e}}_i \odot \mathbf{c}_t$.
F_{sel}, f_{sel}	Scoring module $F_{\text{sel}}(\mathbf{e}_i, \mathbf{c}_t)$ and the two-layer MLP inside it, mapping $4d$ input features to one score and shared across layers.
$\text{Top}_k(\mathcal{A}; s)$	The k highest-scoring entries in a set $\mathcal{A}$ under the scores s.
m	Number of entries selected by score in a layer: $B - \lvert\mathcal{P}\rvert - \lvert\mathcal{R}\rvert = \lvert\mathcal{F}\rvert$ when $\lvert\mathcal{C}\rvert > B$.
m_{img}, $\mathcal{F}_{\text{img}}$	Number of image entries still needed to meet B_{img} after the random draw, $\max(0, B_{\text{img}} - \lvert\mathcal{R} \cap \mathcal{I}\rvert)$, and the image candidates selected in the first stage.
ρ_{train}, ρ_{infer}	Nominal sampling ratios relative to the cache budget during training and inference; ρ denotes the applicable ratio.
$n^{\ell}_{\text{rand},t}$	Actual number of sampled entries, $\lvert\mathcal{R}^{\ell}_t\rvert$, after reserving capacity for protection and the image floor.
n_{rec}	Requested number of tokens protected at the end of the current response.
B_{img}	Effective minimum total number of image entries retained in a layer, pooled across resident images and capped by availability and capacity.
n_{img}	Number of resident image entries in the layer before selection.
b_{img}, f_{img}	Absolute image floor and proportional floor relative to n_{img}; their maximum determines the requested floor before capping.
Training	
p_{full}	Response-token distribution of the frozen VLM with the full-history cache $\mathcal{C}^{\text{full}}_{t-1}$.
p_{keep}	Response-token distribution of the same frozen VLM with the retained cache $\widehat{\mathcal{C}}_{t-1}(\theta)$; θ enters only through retention decisions.
θ, $\mathcal{L}(\theta)$	Trainable SAE and SCS parameters, and the mean full-vocabulary forward KL loss on subsequent response tokens.
g_i	Survival gate: hard selection in the forward pass and a straight-through gradient in the backward pass. Protected and sampled entries have fixed gates of one.
z_i	Sigmoid relaxation of candidate selection, with its sum constrained to m.
η, τ	Threshold enforcing the soft cardinality constraint and temperature of the sigmoid relaxation.
w_i, γ_i, $\bar{\gamma}$	Sigmoid slope $z_i(1 - z_i)$, loss gradient with respect to the gate $\partial\mathcal{L}/\partial g_i$, and its w-weighted mean over $\mathcal{E}$.
sg	Stop-gradient operator: preserves its input value and blocks its gradient.
Indices, inputs, and dimensions	
t, T	Turn index and number of turns in a dialogue.
ℓ, L	Decoder-layer index and number of decoder layers.
i, j	Entry index in selection formulas; j indexes response tokens in the training loss and sampled entries in the sketch sum.
x_t, y_t	User input and response at turn t; $y_{t,<j}$ is the reference response prefix before token j during teacher forcing.
H, D, d	Number of KV heads, dimension per KV head, and dimension of entry and role embeddings.

B Experimental Protocol

This section fixes what is held constant across methods: the benchmarks and their scoring, the serving runtime, the judge, and the baseline configurations.

B.1 Benchmarks and Data

Table 7 lists the evaluated split, the number of dialogues, their length, and the judged score of each benchmark. MMDU (Liu et al., 2024b) and MMRC (Xue et al., 2025) are long-horizon: dialogues run for up to 27 and 30 turns, images enter at later turns, and the resident cache exceeds every budget within a few turns (Figures 1a and 7). MMRC differs in protocol: each served dialogue

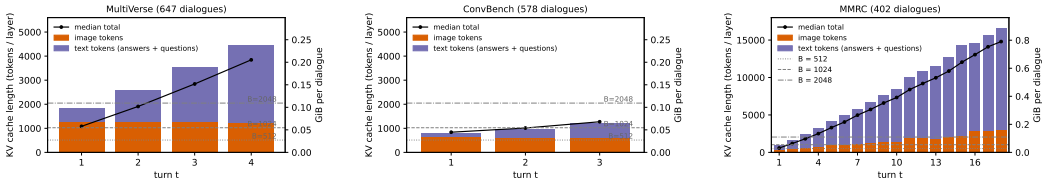


Figure 7: Resident KV cache per dialogue by turn on MultiVerse, ConvBench, and MMRC (Qwen2.5-VL-7B, full KV; the MMDU panel is Figure 1a). Bars split the mean cache into image and text tokens, the line is the median total, and dashed lines mark $B \in \{512, 1024, 2048\}$.

Table 7: Evaluated benchmarks. Turn and image counts are per dialogue (minimum / median / maximum). For MMRC, a served dialogue is one probe appended to its fixed history; the 1,853 probes cover 402 of the 405 histories.

Benchmark	Split	Dialogues	Turns	Images	Per-turn judge score and aggregation
MMDU	val	110	5 / 16 / 27	2 / 3 / 20	Overall score in $[1, 10]$ from the official meta prompt with all dialogue images attached; micro-average over the 1,645 judged turns.
MMRC	history	1,853	1 / 12 / 30	0 / 2 / 8	Score in $[0, 5]$ per probe from the official rubric; mean over probes within each ability, then mean of the six ability means.
MultiVerse	all	647	3 / 4 / 4	1 / 1 / 1	Checklist satisfaction ratio times a quality score in $[1, 10]$, scaled to $[0, 100]$; mean over the turns at each of the four turn positions, then mean of the four position means, as in the original protocol.
ConvBench	all	578	3 / 3 / 3	1 / 1 / 1	Pairwise preference against the human reference at turns 1–3 and for the whole dialogue; we report the mean of the three per-turn win rates averaged with the overall win rate.

is a human-written probe appended to a fixed history, so it measures recall from a given history rather than drift. MultiVerse (Lee et al., 2025) and ConvBench (Liu et al., 2024a) are short-horizon single-image dialogues whose resident cache exceeds B by a small factor rather than by an order of magnitude (Figure 7): the median cache reaches about 3.8K entries per layer on MultiVerse and 1.3K on ConvBench, so at B=2048 ConvBench is effectively uncompressed and at B=512 both are compressed by roughly 2.5–7.5×. They test whether compression harms dialogues that only moderately exceed B. CarryKV is trained on 100 dialogues from MMDU-45K, the training corpus released with MMDU, which is disjoint from the 110 evaluation dialogues; the other three benchmarks are never seen during training. Training teacher-forces the reference responses, whereas evaluation on MMDU, MultiVerse, and ConvBench conditions on the model's own responses, so the selector is evaluated under a history distribution it was not trained on.

B.2 Serving Protocol

All methods run in the same vLLM V1 (Kwon et al., 2023) runtime on one A100 (80 GB) with bf16 weights and cache, and with prefix caching, chunked prefill, and asynchronous scheduling disabled, so that only their KV policy differs. Each dialogue is one persistent request: turn t appends x_t and any new images, the model decodes greedily for at most 4096 tokens or until its end-of-sequence token, and that event triggers the boundary policy before x_{t+1} arrives. On MMDU, MultiVerse, and ConvBench the history consists of the model's own responses, so retention errors compound as in deployment. MMRC follows its official protocol: the ground-truth history is teacher-forced turn by turn through the same request, with the boundary policy applied after each turn, and only the probe answer is decoded. We sweep $B \in \{1024, 2048, 4096\}$ on MMDU and $B \in \{512, 1024, 2048\}$ on the other three benchmarks; CarryKV uses the same B=2048 checkpoint at every budget.

Table 8: Paired bootstrap 95% confidence intervals for CarryKV minus TriAttention under the tight budget (Table 1). Differences whose interval excludes zero are marked *. ConvBench differences are in win-rate points.

Model	Benchmark	Diff.	95% CI	n
Qwen2.5-VL-7B	MMDU (B=1024)	-0.00	$[-0.14, 0.14]$	110
	MultiVerse (B=512)	$+0.85^*$	$[0.26, 1.43]$	647
	MMRC (B=512)	-0.17^*	$[-0.23, -0.10]$	402
	ConvBench (B=512)	$+0.3$	$[-1.2, 1.8]$	578
GLM-4.6V-Flash-9B	MMDU (B=1024)	$+0.18^*$	$[0.10, 0.27]$	110
	MultiVerse (B=512)	$+2.30^*$	$[1.76, 2.84]$	647
	MMRC (B=512)	-0.07	$[-0.14, 0.01]$	402
	ConvBench (B=512)	$+2.9^*$	$[0.5, 5.5]$	578

B.3 Judging and Judge Noise

All benchmarks are scored by Qwen3-VL-32B-Instruct at temperature 0, using the per-benchmark prompts and aggregation of Table 7. MMDU, MMRC, and ConvBench use their official prompts and scripts; ConvBench compares the model response with the human reference pairwise without images. MultiVerse has not released its evaluation code, so we reimplement the protocol of the paper: the judge receives the reference history, the turn's checklist, and the model response, and returns the fraction of satisfied checklist items and a quality score. The same judge scores every method, so judge bias shifts all methods together and the comparisons within a table are paired.

Judge noise. To calibrate the differences in Tables 1, 3, and 4, we compute paired bootstrap 95% confidence intervals by resampling dialogues (5,000 replicates; MMRC is resampled by history, and MultiVerse is aggregated by turn position as in the official metric). Table 8 lists the intervals for CarryKV minus TriAttention, the strongest evicting baseline, under the tight budget. CarryKV leads significantly on MultiVerse for both models and on MMDU and ConvBench for GLM-4.6V-Flash-9B; in the remaining cells the two methods are within noise of each other, and TriAttention leads significantly in only one of the eight cells, MMRC on Qwen2.5-VL-7B, where the gap is concentrated in information extraction and memory recall and the overall scores tie at B=1024 (Appendix D.1). The two significant benchmarks test different parts of the method. On MultiVerse at B=512 the protected set and the image floor fix most of the budget before scoring (Appendix D.3), so the margin there reflects boundary timing and structural protection. On MMDU the scorer places most of the budget: there Random-fill (Table 4) is significantly below the default (-0.25, $[-0.38, -0.12]$), and CarryKV is significantly above TriAttention on GLM-4.6V-Flash-9B. In the module ablation (Table 3), the full model exceeds the variant without either module significantly on GLM-4.6V-Flash-9B ($+0.14$, $[0.05, 0.24]$) and by a sign-consistent but non-significant margin on Qwen2.5-VL-7B ($+0.05$). All ρ_{infer} settings in Table 4 are within noise of the default.

B.4 Baselines

Table 9 summarizes when each baseline compresses, what signal it uses, what remains resident afterwards, and what B means for it; all run inside the same runtime and harness. RocketKV-MT (Behnam et al., 2025) and SparseVILA (Khaki et al., 2025) restrict which entries attention reads but never free memory, so their resident history equals Full KV's at every budget; RocketKV-MT keeps it in the vLLM pool, whereas SparseVILA keeps the full history in a side buffer outside the pool and holds only the pruned entries in the pool. TriAttention and CarryKV evict physically and EpiCache parks inactive episodes on the host. Equal B therefore does not imply equal resident memory: Table 1 holds the nominal budget fixed, and the memory-side consequences appear in Table 2 and Appendix E. We do not run TrimKV and DBTrimKV (Bui et al., 2026b;a), the closest learned methods, because their released gates cover only Qwen3-VL-4B/8B, and transferring them to our models would require training a 57M-parameter gate per model on MMDU-45K and M4-Instruct on multiple H100s. They also differ from CarryKV in mechanism: a retention coefficient is predicted once when an entry is written and decayed with age, and eviction happens continuously during generation whenever the budget is exceeded, whereas CarryKV leaves the cache untouched

Table 9: Baseline configuration. All methods share the runtime, model weights, decoding parameters, and dialogue workloads of Table 7.

Method	When	Signal and configuration	Resident after	Meaning of B
Full KV	never	—	everything	unused
RocketKV-MT	prefill of every user input	Official SCBench multi-turn variant: SnapKV selection per layer and KV head with observation window 128 and kernel 63; exact top-k retrieval of $B/2$ entries plus the coarse estimation traffic.	everything (sparse read)	equivalent read bandwidth per layer
SparseVILA	image prefill and every turn	Visual tokens pruned to 25% at prefill by received attention in the last vision layer; at each turn 10% of the remaining visual tokens are re-selected from the full backing store.	everything (pruned entries in the pool, full history in a side buffer)	read cap on the active prefix at turn start, plus the generated tail
TriAttention	every 128 generated tokens while resident $> B$	Official Triton scorer with calibration statistics from 50 MMDU-45K dialogues; keeps the top-B entries of every head.	top-B per head (blocks reclaimed)	per-head, per-layer cap
EpiCache	when resident $> M{+}512$	Blocks of 512 tokens routed by MiniLM-L6-v2 embeddings into 4 episode clusters (τ=0.5, window 4); the system prompt and image prefix are protected; inactive episodes move to host memory.	active episode (plus protected prefix)	soft target $M{=}B$; not a hard cap
CarryKV	every turn boundary	Sections 4.2–4.3; ρ_{infer}=0.10, n_{rec}=256, image floor $\max(5B/16, \lceil 0.1\, n_{\text{img}} \rceil)$ distributed across images.	exactly $\min(B, \lvert\mathcal{C}\rvert)$ per layer (blocks reclaimed)	per-layer cap at the boundary

during a response, re-selects the retained cache at the turn boundary conditioned on the current cache, and is trained with a loss on the subsequent turn only. TriAttention, which we do run, occupies a middle point: it also evicts during generation under a fixed per-head budget, but with a training-free score.

C Method Details

This section completes Section 4 with the exact retention quotas, the training procedure, the gate and its gradient, and the configuration of the reported checkpoints.

C.1 Retention Quotas

The following rules apply independently to each layer at each turn boundary; we omit the indices t and ℓ and write $B_{\text{eff}} = \min(B, |\mathcal{C}|)$ for the effective capacity. The protected set $\mathcal{P}$ contains the system prefix, the current question text, and the last n_{rec} tokens of the current response; if these exceed B_{eff}, the response tail is clipped to the capacity left after the system prefix and the question text, so that $|\mathcal{P}| \le B_{\text{eff}}$. Let n_{img} be the number of resident image entries, b_{img} the absolute image floor, and f_{img} the proportional image floor. Since $\mathcal{P}$ contains only textual entries, the effective image floor is

$$B_{\text{img}} = \min\{n_{\text{img}},\ B_{\text{eff}} - |\mathcal{P}|, \max(b_{\text{img}}, \lceil f_{\text{img}} n_{\text{img}} \rceil)\}\,, \tag{16}$$

with defaults $b_{\text{img}} = 5B/16$ and $f_{\text{img}} = 0.10$. Equation 12 fills this floor in its pooled form, which is the form used during training; at inference the same total is distributed across the resident images as evenly as their remaining entries allow, each image meeting its share with its highest-scoring entries. Capacity for $\mathcal{P}$ and the image floor is reserved before sampling, giving the random subset

size

$$n_{\text{rand}} = \min(\lfloor \rho B_{\text{eff}} \rfloor, B_{\text{eff}} - |\mathcal{P}| - B_{\text{img}}, |\mathcal{C} \setminus \mathcal{P}|), \tag{17}$$

where sampled image entries count toward B_{img}.

C.2 Training

Procedure. Training runs for one epoch with one optimizer step per dialogue. The frozen VLM first runs teacher-forced without eviction to obtain p_{full} at every response token, then runs again with the boundary policy applied after every turn and the survival gates of Equation 15 multiplied into attention to obtain p_{keep}; the loss of Equation 14 is then backpropagated once. A dialogue whose cache never exceeds B produces no eviction and hence no gradient and is skipped; at B=2048, 94 of the 100 dialogues produce a step on both models. The random subset $\mathcal{R}$ is drawn once per boundary and held fixed within the step.

Gate. For a query $\mathbf{q}$ over the resident entries of one layer with attention weights $\alpha_i \propto \exp(\mathbf{q}^\top \mathbf{k}_i)$, the gated output is

$$\mathbf{o} = \frac{\sum_i \alpha_i \, g_i \, \mathbf{v}_i}{\sum_i \alpha_i \, g_i}, \tag{18}$$

which for binary g_i equals attention restricted to the retained entries, so the forward pass matches inference exactly. We compute it with a single FlashAttention call by augmenting the value of entry i to $[g_i \mathbf{v}_i, \, g_i]$, so that the numerator and the denominator are read off the same output.

Gradient. Protected and randomly sampled entries have fixed gates of one and receive no gradient. For the candidates $\mathcal{E}$ of a fixed layer and boundary, let $m = B_{\text{eff}} - |\mathcal{P}| - |\mathcal{R}|$ be the number of entries selected by score. The soft values in Equation 15 are

$$z_i = \sigma\left(\frac{s_i - \eta}{\tau}\right), \qquad \sum_{i \in \mathcal{E}} z_i = m, \tag{19}$$

where σ is the sigmoid, τ=0.5 is its temperature, and the threshold η is found by bisection. The image floor shapes only the hard forward selection; the relaxation runs over all candidates. Differentiating the constraint implicitly, with $w_i = z_i(1 - z_i)$ and $\gamma_i = \partial \mathcal{L} / \partial g_i$, gives

$$\frac{\partial \mathcal{L}}{\partial s_i} = \frac{w_i}{\tau}(\gamma_i - \bar{\gamma}), \qquad \bar{\gamma} = \frac{\sum_{j \in \mathcal{E}} w_j \gamma_j}{\sum_{j \in \mathcal{E}} w_j}. \tag{20}$$

The gradient sums to zero over $\mathcal{E}$, so training changes the relative ranking of candidates rather than their absolute scores, and it is concentrated on entries near the threshold, where w_i is largest. It reaches f_{sel} through s_i and f_{enc} through both $\tilde{\mathbf{e}}_i$ and the sketch $\mathbf{c}_t$.

Configuration. Table 10 lists the configuration of the two reported checkpoints; besides the budget, only the sampling ratio and the image-floor distribution differ between training and inference.

D Additional Quality Results

This section reports the quality results that the main text summarizes: the full budget sweep and per-dimension scores behind Table 1, case studies of cross-turn visual dependency, and the complete parameter sensitivity behind Section 5.4.

D.1 Detailed Quality Results

Throughout this appendix, * marks RocketKV-MT and SparseVILA, which retrieve sparsely without eviction so the full cache stays resident, and bold and underline mark the best and second-best compression method per column within each budget (ties share the mark).

Table 10: Configuration of the reported CarryKV checkpoints. One checkpoint per VLM, trained at B=2048 and evaluated at every budget.

	Qwen2.5-VL-7B	GLM-4.6V-Flash-9B
Backbone and selector		
Decoder layers L / KV heads H / head dim. D	28 / 4 / 128	40 / 2 / 128
Entry feature dimension $2HD + 3$	1,027	515
Embedding dimension d	64	64
Hidden width of f_{enc} / f_{sel}	128 / 64	128 / 64
Trainable parameters	161K	95K
Protected system prefix (tokens)	11	2
Retention quotas (training / inference)		
Budget B (per layer)	2048 / $\{512, 1024, 2048, 4096\}$	
Sampling ratio ρ	0.25 / 0.10	
Recent-response protection n_{rec}	256 / 256	
Image floor b_{img}, f_{img}	$5B/16$, 0.10 (pooled / distributed per image)	
Optimization		
Optimizer	AdamW, lr 2×10^{-4}, $\beta = (0.9, 0.95)$, weight decay 10^{-4}	
Gradient clipping / temperature τ	1.0 / 0.5	
Epochs / dialogues / optimizer steps	1 / 100 / 94	
Training dialogues: turns / images	4–25 (mean 14.7) / 2–5 (mean 3.0), from MMDU-45K	
Response token limit	4096	
Compute		
Precision	bf16 frozen backbone, fp32 selector	
Peak GPU memory	23.9 GB	27.7 GB
Wall-clock time (one A100)	17.7 min	22.6 min

Table 11: Generation quality at every evaluated budget (extends Table 1): MMDU at $B \in \{1024, 2048, 4096\}$ and the other benchmarks at $B \in \{512, 1024, 2048\}$ tokens per layer. ConvBench is reported as win rate in %.

Budget	Method	Qwen2.5-VL-7B				GLM-4.6V-Flash-9B			
		MMDU	MultiVerse	MMRC	ConvBench	MMDU	MultiVerse	MMRC	ConvBench
–	Full KV	6.58	49.47	2.62	13.20%	7.13	54.97	3.34	29.40%
1024	RocketKV-MT* (ICML'25)	4.46	37.24	2.29	10.80%	**7.05**	**55.42**	**3.31**	28.20%
	SparseVILA* (ICCV'25)	5.97	42.40	2.48	8.50%	6.60	47.63	3.22	18.50%
	EpiCache (ICML'26)	4.30	40.13	1.70	4.60%	5.38	43.32	2.35	9.00%
	TriAttention (ICML'26)	**6.18**	48.76	**2.58**	**12.60%**	6.78	53.97	**3.31**	**28.60%**
	CarryKV (ours)	**6.18**	**49.71**	**2.58**	12.00%	6.96	55.24	**3.31**	28.30%
2048	RocketKV-MT* (ICML'25)	6.40	49.28	2.61	11.70%	**7.10**	55.25	**3.35**	28.20%
	SparseVILA* (ICCV'25)	5.97	42.43	2.61	7.40%	6.70	47.52	3.30	18.20%
	EpiCache (ICML'26)	4.40	40.12	1.70	4.70%	5.54	43.63	2.36	9.40%
	TriAttention (ICML'26)	**6.45**	**49.41**	**2.63**	12.00%	7.01	55.06	3.32	**29.00%**
	CarryKV (ours)	6.37	49.10	2.61	**12.70%**	7.04	**55.26**	3.33	28.80%
512 / 4096†	RocketKV-MT* (ICML'25)	**6.62**	21.24	1.40	6.80%	7.16	**55.06**	**3.31**	27.90%
	SparseVILA* (ICCV'25)	6.08	41.95	2.10	8.70%	6.66	47.29	2.82	18.80%
	EpiCache (ICML'26)	4.51	40.00	1.69	4.90%	5.54	43.45	2.33	8.50%
	TriAttention (ICML'26)	6.55	48.05	**2.42**	11.60%	7.14	52.44	3.18	26.00%
	CarryKV (ours)	6.60	**48.90**	2.25	**11.90%**	**7.22**	54.74	3.12	**28.90%**

†MMDU at B=4096; the other benchmarks at B=512.

All budgets. Table 11 extends Table 1 to every evaluated budget. At the budgets of Table 1, CarryKV is the best or second-best compression method in 15 of the 16 columns of the B=1024 and the $512/4096$ rows; the exception is MMRC on GLM-4.6V-Flash-9B at B=512. At B=2048, RocketKV-MT, TriAttention, and CarryKV are all within 0.25 points of Full KV on MMDU, 0.4 on MultiVerse, and 0.02 on MMRC, so the budget rather than the policy determines quality there. SparseVILA does not approach Full KV as the budget grows: it stays 7–8 points below on MultiVerse at every budget, because it prunes visual tokens at prefill regardless of B.

Table 12: MMDU scores by dimension behind Table 1: creativity, richness, visual perception, logical coherence, answer accuracy, and image relationship understanding, each micro-averaged over the 1,645 judged turns; Overall is the reported score.

Budget	Method	Creativity	Richness	Visual	Logic	Accuracy	Image rel.	Overall
Qwen2.5-VL-7B								
–	Full KV	5.98	6.39	6.43	7.83	6.66	6.63	6.58
Relaxed (B=4096)	RocketKV-MT*	5.99	6.42	**6.48**	**7.91**	**6.74**	**6.69**	**6.62**
	SparseVILA*	5.63	6.00	5.80	7.48	6.11	6.06	6.08
	EpiCache	4.24	4.47	4.24	5.85	4.56	4.39	4.51
	TriAttention	5.96	6.40	6.42	7.85	6.65	6.62	6.55
	CarryKV	**6.01**	**6.43**	6.43	7.85	6.68	6.61	6.60
Tight (B=1024)	RocketKV-MT*	4.22	4.05	4.64	5.36	4.58	4.56	4.46
	SparseVILA*	5.56	5.91	5.61	7.39	5.99	5.90	5.97
	EpiCache	4.08	4.27	3.98	5.66	4.37	4.13	4.30
	TriAttention	**5.71**	6.06	**5.93**	7.50	6.20	**6.19**	**6.18**
	CarryKV	5.67	**6.09**	5.84	**7.53**	**6.23**	6.15	**6.18**
GLM-4.6V-Flash-9B								
–	Full KV	6.44	6.50	7.26	8.35	7.48	7.38	7.13
Relaxed (B=4096)	RocketKV-MT*	6.44	6.50	7.23	8.35	7.51	7.43	7.16
	SparseVILA*	6.13	6.12	6.60	7.99	6.92	6.85	6.66
	EpiCache	5.55	5.63	5.26	7.00	5.57	5.46	5.54
	TriAttention	6.46	6.49	7.24	8.34	7.50	7.40	7.14
	CarryKV	**6.56**	**6.58**	**7.36**	**8.38**	**7.56**	**7.44**	**7.22**
Tight (B=1024)	RocketKV-MT*	**6.41**	**6.44**	**7.11**	**8.26**	**7.38**	**7.29**	**7.05**
	SparseVILA*	6.09	6.08	6.49	7.94	6.87	6.74	6.60
	EpiCache	5.51	5.61	5.04	6.86	5.32	5.23	5.38
	TriAttention	6.26	6.26	6.69	8.07	7.01	6.95	6.78
	CarryKV	6.38	**6.44**	7.03	8.24	7.24	7.11	6.96

Per-dimension scores. Tables 12–15 decompose each cell of Table 1 and answer three questions that the aggregates leave open.

When is quality lost? Only after the first turn boundary. On MultiVerse at B=512, CarryKV matches Full KV at turn 1 (47.15 versus 47.00 on Qwen2.5-VL-7B; 51.85 versus 51.32 on GLM-4.6V-Flash-9B) because it does not touch the cache before the first boundary, whereas SparseVILA, which prunes at prefill, is already 15–17 points below Full KV at turn 1, and TriAttention, which evicts during generation, 1.2–1.6 points below. Over the later turns CarryKV stays within 1.2 points of Full KV on both models, while EpiCache drops 13–14 points from turn 1 to turn 2. The judge quality factor is nearly constant across methods (6.78–7.00), so every difference in the official score is a difference in checklist satisfaction, that is, in how much of the history the answer could still draw on. ConvBench shows the same turn-1 match (18.0% versus 18.0% and 22.7% versus 21.8%).

What does the tight budget remove? Image evidence, not fluency. On MMDU at B=1024 the loss of CarryKV relative to Full KV is concentrated in the three visually grounded dimensions, visual perception, image relationship understanding, and answer accuracy (0.59, 0.48, and 0.43 on Qwen2.5-VL-7B; 0.23, 0.27, and 0.24 on GLM-4.6V-Flash-9B), whereas creativity, richness, and logical coherence lose at most 0.31 and 0.11 on the two models.

Where is the residual MMRC gap? In the abilities that must retrieve one specific earlier detail. At B=512, CarryKV trails the best compression method by 0.33 on information extraction and 0.39 on memory recall on Qwen2.5-VL-7B, and by 0.40 on information extraction on GLM-4.6V-Flash-9B, while every other ability is within 0.24 of the best method and answer refusal is above Full KV on Qwen2.5-VL-7B. This is the expected cost of deciding at the boundary without seeing the next question: at the tightest budget the carried set cannot hold every detail a later probe may ask for. The overall scores tie at B=1024 (2.58 for both), where information extraction reaches 3.49 against 3.47 for TriAttention, although memory recall remains 0.14 below.

D.2 Case Study

Figures 8 and 9, together with Figure 2 of the main text, show four dialogues in which a later turn depends on an entry introduced at an earlier turn. Each figure plots the per-turn judge score at B=1024 (the five compression methods, or CarryKV, TriAttention, and the unpruned Full KV in Figure 2) and expands the key turns: the question, the referenced images with the turn at which each was first shown, what the turn needs from the history, and for every method a verbatim excerpt of its answer (markdown removed, nothing paraphrased) with the decisive entity highlighted, green

Table 13: MultiVerse scores by turn position behind Table 1. Turn k is the mean score of the turns at position k; Checklist and Quality are the dialogue-level means of the checklist satisfaction ratio and the judge quality score in $[1, 10]$; Official is the reported score.

Budget	Method	Turn 1	Turn 2	Turn 3	Turn 4	Checklist	Quality	Official
Qwen2.5-VL-7B								
–	Full KV	47.00	47.69	49.99	53.19	0.706	6.98	49.47
Relaxed (B=1024)	RocketKV-MT*	46.43	39.57	31.88	31.07	0.534	**6.97**	37.24
	SparseVILA*	31.54	41.06	46.40	50.61	0.605	6.90	42.40
	EpiCache	47.15	34.07	37.44	41.86	0.574	6.90	40.13
	TriAttention	46.60	46.88	49.22	52.33	0.696	**6.97**	48.76
	CarryKV	**47.35**	**47.28**	**50.32**	**53.88**	**0.710**	**6.97**	**49.71**
Tight (B=512)	RocketKV-MT*	41.12	18.61	12.70	12.51	0.306	6.78	21.24
	SparseVILA*	31.32	40.60	46.26	49.64	0.599	6.87	41.95
	EpiCache	47.02	33.68	37.81	41.49	0.571	6.91	40.00
	TriAttention	45.80	45.40	48.66	52.35	0.686	**6.97**	48.05
	CarryKV	**47.15**	**46.52**	**48.80**	**53.14**	**0.698**	**6.97**	**48.90**
GLM-4.6V-Flash-9B								
–	Full KV	51.32	53.42	56.68	58.44	0.784	6.99	54.97
Relaxed (B=1024)	RocketKV-MT*	**51.89**	**54.18**	56.26	59.35	**0.791**	6.99	**55.42**
	SparseVILA*	34.97	46.99	52.15	56.42	0.679	6.95	47.63
	EpiCache	51.36	38.06	39.63	44.22	0.619	6.96	43.32
	TriAttention	51.19	52.57	55.07	57.05	0.770	**7.00**	53.97
	CarryKV	51.62	53.16	**56.74**	**59.46**	0.788	6.99	55.24
Tight (B=512)	RocketKV-MT*	**52.16**	**53.24**	**56.28**	**58.56**	**0.786**	6.99	**55.06**
	SparseVILA*	34.63	47.11	51.76	55.68	0.674	6.94	47.29
	EpiCache	51.48	37.63	41.00	43.70	0.621	6.97	43.45
	TriAttention	49.75	50.75	52.76	56.51	0.748	**7.00**	52.44
	CarryKV	51.85	52.61	56.19	58.33	0.781	6.99	54.74

Table 14: MMRC scores by ability behind Table 1: information extraction (IE), cross-turn reasoning (CR), information update (IU), image management (IM), memory recall (MR), and answer refusal (AR), each the mean probe score in $[0, 5]$ within the ability; Overall is the mean of the six.

Budget	Method	IE	CR	IU	IM	MR	AR	Overall
Qwen2.5-VL-7B								
–	Full KV	3.54	3.30	3.08	1.55	3.31	0.90	2.62
Relaxed (B=1024)	RocketKV-MT*	2.50	2.88	2.82	1.54	3.11	0.88	2.29
	SparseVILA*	3.29	**3.30**	2.92	**1.57**	2.93	0.89	2.48
	EpiCache	2.16	2.66	1.97	0.50	2.07	0.86	1.70
	TriAttention	3.47	3.23	3.13	1.54	**3.28**	0.80	**2.58**
	CarryKV	**3.49**	3.23	**3.14**	1.51	3.14	**0.95**	**2.58**
Tight (B=512)	RocketKV-MT*	1.31	1.56	1.84	1.00	2.11	0.60	1.40
	SparseVILA*	2.61	2.81	2.71	1.26	2.32	0.91	2.10
	EpiCache	2.11	2.66	1.95	0.52	2.00	0.90	1.69
	TriAttention	**3.20**	**3.08**	**2.96**	**1.50**	**3.10**	0.67	**2.42**
	CarryKV	2.87	2.93	2.72	1.34	2.71	**0.94**	2.25
GLM-4.6V-Flash-9B								
–	Full KV	3.86	3.93	3.51	3.26	3.91	1.58	3.34
Relaxed (B=1024)	RocketKV-MT*	3.83	3.85	3.60	**3.13**	3.84	1.60	**3.31**
	SparseVILA*	3.53	3.87	3.58	2.94	3.56	**1.83**	3.22
	EpiCache	2.24	3.24	2.53	1.83	2.54	1.73	2.35
	TriAttention	**3.86**	3.87	3.50	3.09	**3.89**	1.66	**3.31**
	CarryKV	3.77	**3.93**	**3.61**	3.02	3.81	1.73	**3.31**
Tight (B=512)	RocketKV-MT*	**3.70**	**3.85**	**3.55**	**3.13**	**3.77**	1.83	**3.31**
	SparseVILA*	2.83	3.47	2.90	2.83	3.07	**1.85**	2.82
	EpiCache	2.26	3.26	2.40	1.89	2.46	1.74	2.33
	TriAttention	3.58	3.77	3.46	2.76	3.73	1.80	3.18
	CarryKV	3.30	3.61	3.43	2.92	3.63	1.81	3.12

Table 15: ConvBench win rates (%) by turn behind Table 1: pairwise win rate against the human reference at turns 1–3, their mean (Per-turn), the whole-dialogue win rate (Overall), and the reported composite, the mean of Per-turn and Overall.

Budget	Method	Turn 1	Turn 2	Turn 3	Per-turn	Overall	Composite
Qwen2.5-VL-7B							
–	Full KV	18.00%	20.40%	10.50%	16.30%	10.00%	13.20%
Relaxed (B=1024)	RocketKV-MT*	17.00%	18.20%	8.10%	14.40%	7.30%	10.80%
	SparseVILA*	12.50%	13.50%	7.80%	11.30%	5.70%	8.50%
	EpiCache	13.50%	4.80%	3.10%	7.10%	2.10%	4.60%
	TriAttention	17.80%	19.20%	**9.90%**	15.60%	**9.50%**	**12.60%**
	CarryKV	**18.20%**	**20.60%**	9.30%	**16.00%**	8.00%	12.00%
Tight (B=512)	RocketKV-MT*	15.20%	10.90%	3.50%	9.90%	3.60%	6.80%
	SparseVILA*	11.40%	12.30%	7.80%	10.50%	6.90%	8.70%
	EpiCache	12.80%	5.40%	3.10%	7.10%	2.80%	4.90%
	TriAttention	17.60%	18.50%	9.20%	15.10%	8.10%	11.60%
	CarryKV	**18.00%**	**19.00%**	**9.50%**	**15.50%**	**8.30%**	**11.90%**
GLM-4.6V-Flash-9B							
–	Full KV	21.80%	44.50%	28.50%	31.60%	27.20%	29.40%
Relaxed (B=1024)	RocketKV-MT*	**23.00%**	**42.70%**	26.60%	**30.80%**	25.60%	28.20%
	SparseVILA*	13.70%	29.10%	20.40%	21.00%	15.90%	18.50%
	EpiCache	12.60%	13.00%	9.50%	11.70%	6.40%	9.00%
	TriAttention	20.90%	41.70%	27.50%	30.00%	**27.20%**	**28.60%**
	CarryKV	20.80%	42.40%	**28.00%**	30.40%	26.30%	28.30%
Tight (B=512)	RocketKV-MT*	20.90%	**42.90%**	26.80%	30.20%	25.60%	27.90%
	SparseVILA*	14.20%	31.00%	19.70%	21.60%	15.90%	18.80%
	EpiCache	13.70%	11.90%	8.80%	11.50%	5.50%	8.50%
	TriAttention	21.60%	42.00%	25.30%	29.60%	22.30%	26.00%
	CarryKV	**22.70%**	41.70%	**28.20%**	**30.80%**	**27.00%**	**28.90%**

if consistent with the image and the earlier turns and red if substituted or fabricated, next to the judge's score. We selected the dialogues by the margin of CarryKV over the eviction baselines, so they illustrate the intended mechanism rather than the average case (Table 12, Appendix D.1). RocketKV-MT and SparseVILA keep the full cache resident and only sparsify reads, so their panels show the effect of sparsifying reads without eviction.

Dialogue 85: long-range recall. The eviction baselines track CarryKV until the fourth poster enters at turn 10 and the cache exceeds the budget. TriAttention then scores 3–5 on every turn from 13 to 17 but 9 and 10 at turns 12 and 18, the two late turns that do not involve Image2, so what it lost is specifically Image2. Those entries are seven turns old at turn 11 and no recent question attends to them, so any criterion driven by recency or by the attention of recent queries ranks them low. CarryKV re-selects the retained cache at the boundary and is trained on the loss of the following turn, so the entries carrying the title survive, and its scores at turns 11, 14, and 17 (9, 10, 10) match RocketKV-MT, which never evicts. The same figure shows the selector's limit: what it keeps for Image1 suffices to describe the poster but not to name the film.

Dialogue 27: one boundary. Dialogue 27 (Figure 2) is the shortest-range version of the same failure: the image entries are only one boundary old at turn 2, but the question no longer re-shows the image, and TriAttention's ranking already drops enough of them that the model substitutes a different animal and carries the substitution through the rest of the dialogue (it scores 4 on every turn from 8 to 17). CarryKV stays within one point of Full KV on 15 of the 18 turns; of the remaining three, turn 4 is one where CarryKV exceeds Full KV, and turns 5 and 6, where it falls below both TriAttention and Full KV, are butterfly questions, so its margin is specifically on Image1. The species names offered at turn 2 by CarryKV (*Porcellanidae*) and Full KV (*Hymenocera*, a shrimp genus) are both wrong and are left unhighlighted in the figure.

Dialogues 49 and 1320: single turns. Dialogue 49 is the Qwen counterpart of dialogue 85 with a single late question: the eviction baselines do not merely lose the title but answer with invented cover text, and the two non-evicting methods fail for unrelated reasons. Dialogue 1320 isolates a

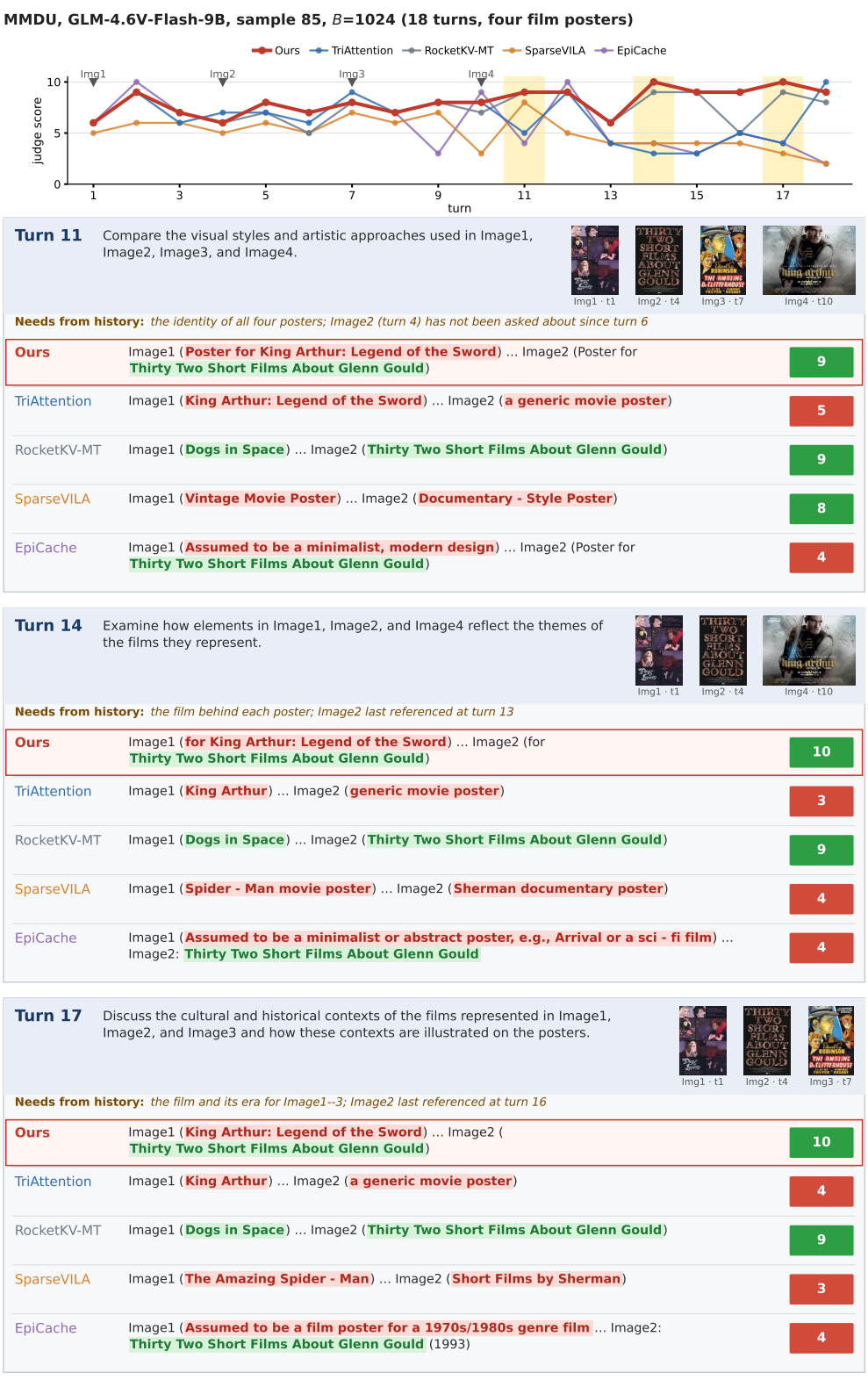
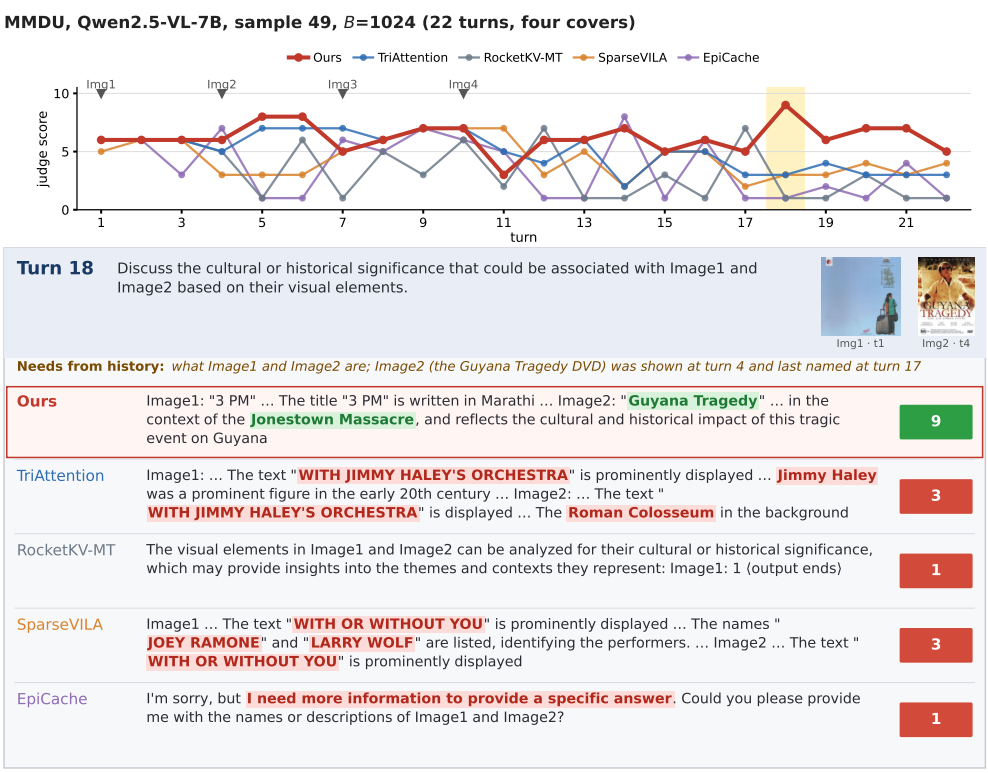
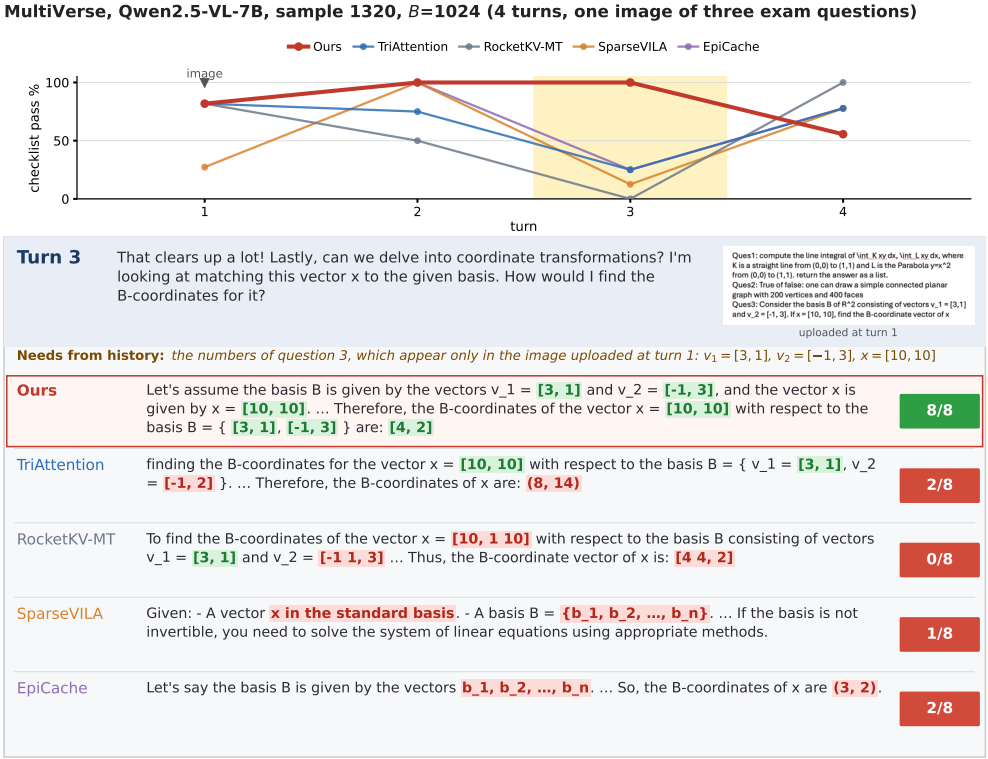


Figure 8: MMDU dialogue 85 on GLM-4.6V-Flash-9B, B=1024. Four posters enter at turns 1, 4, 7, and 10. The decisive entry is the title of Image2 (*Thirty Two Short Films About Glenn Gould*): shown at turn 4, last asked about at turn 6, and needed again at turns 11, 14, and 17. CarryKV still names it at all three turns, as do RocketKV-MT (which never evicts) and EpiCache (which instead loses Image1); TriAttention and SparseVILA do not ("a generic movie poster", "Short Films by Sherman"). CarryKV's own error on Image1 ("King Arthur", the title of Image4) is marked in red.



(a) MMDU dialogue 49 on Qwen2.5-VL-7B, turn 18. Image2 (*Guyana Tragedy*) was shown at turn 4 and last named at turn 17; only CarryKV still names it. TriAttention and SparseVILA answer with cover text found on none of the four covers (invented at turn 17 and repeated since), RocketKV-MT truncates, and EpiCache refuses. CarryKV's "3 PM" in Marathi for Image1 is itself wrong (the script is Bengali) and is left unhighlighted.



(b) MultiVerse dialogue 1320 on Qwen2.5-VL-7B, turn 3. The numbers of question 3 (v_1=[3, 1], v_2=[−1, 3], x=[10, 10]) exist only in the image uploaded at turn 1. CarryKV reads all three and answers [4, 2]; the other methods alter v_2, corrupt the digits, or fall back to placeholder vectors.

Figure 9: Two cases at B=1024, each expanded at one key turn.

Table 16: Parameter sensitivity on Qwen2.5-VL-7B. Scores use the official MultiVerse metric at $B = 512$ and the turn-level micro-average on MMDU at $B = 1024$. Shaded rows are defaults; Δ is relative to the default.

Parameter	Setting	MultiVerse	Δ	MMDU	Δ
ρ_{infer}	0	48.94	+0.04	6.252	+0.075
	0.10	48.90	0.00	6.177	0.000
	0.25	48.92	+0.02	6.206	+0.029
	0.50	48.92	+0.02	6.206	+0.029
Filler	Random-fill†	48.81	−0.09	5.928	−0.249
ρ_{train}	0.10	48.64	−0.26	6.187	+0.010
	0.25	48.90	0.00	6.177	0.000
	0.50	48.93	+0.03	6.216	+0.039
Image floor	0	48.58	−0.32	6.212	+0.035
	$B/8$	48.96	+0.06	6.190	+0.013
	$5B/16$	48.90	0.00	6.177	0.000
	$B/2$	48.90	0.00	6.214	+0.037
n_{rec}	0	49.24	+0.34	6.251	+0.074
	64	49.28	+0.38	6.302	+0.125
	128	48.96	+0.06	6.274	+0.097
	256	48.90	0.00	6.177	0.000
	Maximum tail‡	47.58	−1.32	5.824	−0.353

†Reused random-filling baseline under the same structural constraints, not a separately executed $\rho_{\text{infer}} = 1$ run.
‡The requested answer tail is clipped as in Appendix C.1. Deltas are computed from the reported scores (two decimals for MultiVerse, three for MMDU); the default MMDU score 6.177 is the 6.18 of Tables 1 and 3.

single image: every method sees the same text history, so the difference between $[4, 2]$ and the wrong or placeholder answers comes down to whether the entries encoding the numbers were still in the cache at turn 3. Finally, CarryKV's own scores move by several points between neighbouring turns (turn 11 in dialogue 49, turn 4 in dialogue 1320), so the figures should be read for the pattern at the key turns rather than turn by turn.

D.3 Parameter Sensitivity

Table 16 reports every run behind Tables 5 and 4 for Qwen2.5-VL-7B. We change one parameter at a time, holding the others at $\rho_{\text{train}} = 0.25$, $\rho_{\text{infer}} = 0.10$, an image floor of $5B/16$, and $n_{\text{rec}} = 256$. The two alternative training ratios are retrained with the data, schedule, and training budget of Table 10; every other row reuses the reference checkpoint. The zero-floor setting disables both the absolute and the proportional image floor of Appendix C.1. Setting $n_{\text{rec}} = 0$ removes answer-tail protection while keeping the system prefix and current question protected; the maximum-tail setting requests 4096 answer tokens, so the protected tail is clipped by the rule of Appendix C.1. Random-fill reuses the random-scoring baseline with the same protection, image floor, and 10% random subset, and is listed apart from the inference-ratio sweep († in Table 16).

Where the scorer acts. The table shows where the sensitivity comes from. MultiVerse is evaluated at B=512, where the default answer tail (256 entries) and image floor ($5B/16 = 160$) fix 416 of the 512 entries before any scoring, so the selector places fewer than 96 entries per layer; on MMDU at B=1024 the scorer places most of the budget. Random-fill differs from the default only in the scores, so its gap isolates the learned scorer: 0.25 on MMDU but 0.09 on MultiVerse. The same split explains why two settings lower MultiVerse by about 0.3 while leaving MMDU within 0.04 of the default ($\rho_{\text{train}} = 0.10$, -0.26; no image floor, -0.32): on MultiVerse the structural constraints, not the scorer, decide most of the retained cache.

Role of the random subset. The ρ_{train} rows complete the argument of Section 5.4 that the random subset matters during training rather than at deployment: removing it at inference costs nothing, whereas training with the smaller draw does on MultiVerse. $\rho_{\text{infer}} = 0.25$ and 0.50 give identical scores on MultiVerse because n_{rand} is capped by the capacity left after the protected set and the

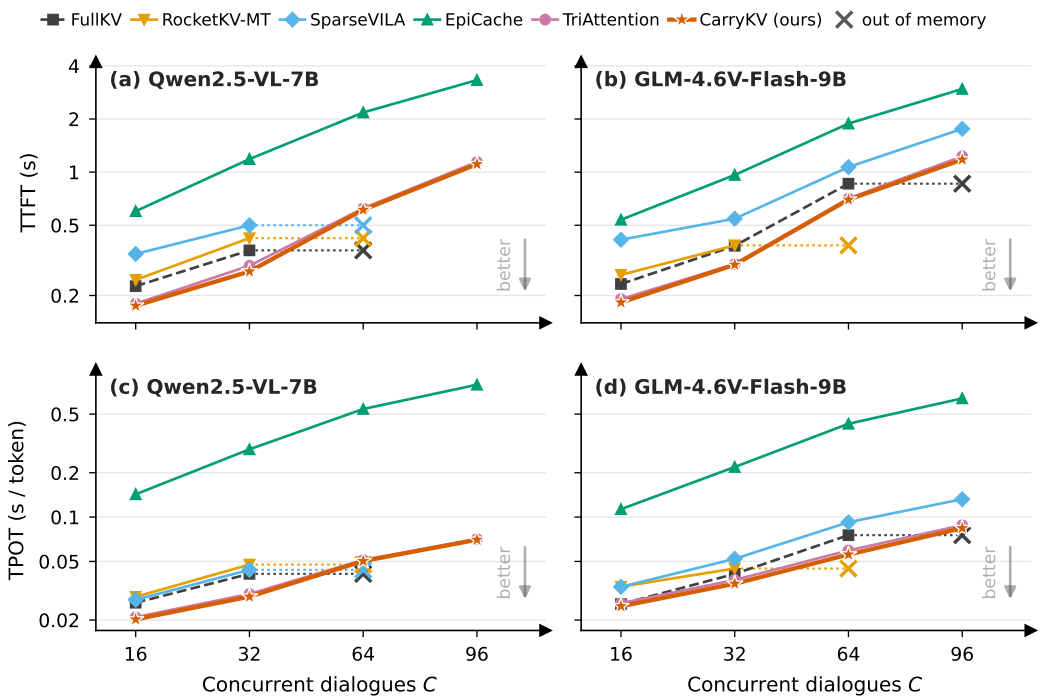


Figure 10: Time to first token (top) and time per output token (bottom) as a function of the concurrency C on MMDU (same sweep as Figure 6; log scale). A cross marks the first C where a method runs out of GPU memory, and the dotted segment leading to it extends the last completed run.

image floor (Appendix C.1): at B=512 both requests (128 and 256) exceed the cap and draw the same subset. On MMDU the cap depends on the per-dialogue image floor, and the two settings coincide on every judged turn.

Answer protection. The maximum answer tail is the largest change in the table (-1.32 and -0.353), since it leaves the scorer almost no capacity. Conversely, reducing n_{rec} returns capacity to the scorer and improves both benchmarks ($n_{\text{rec}} = 64$: $+0.38$ and $+0.125$); the default of 256 was fixed before evaluation and not retuned.

E Serving Details

This section describes how the retained set is turned into freed memory inside vLLM and reports the memory and latency measurements behind Table 2.

Cache compaction. Because an entry holds all KV heads of a token, the retained set of a layer can be compacted at the block level: after SCS returns the retained set of every layer, the retained entries are packed into contiguous blocks, the block table of the request is rewritten, and the freed tail blocks are returned to the pool. Retained entries keep their original positions, so no positional re-encoding is required.

Memory. The sweep serves the 110 MMDU dialogues twice (220 requests) after a warm-up of 16 dialogues that is excluded from all numbers, and we record the peak occupancy of the vLLM KV pool over the measured segment. It is set by the budget rather than by the policy: at matched C, CarryKV, TriAttention, and EpiCache occupy the same peak within 1.3 GiB, whereas Full KV and RocketKV-MT, which keep the full history resident, need 3–5$\times$ more. Full KV exhausts the pool at C=64 on Qwen2.5-VL-7B and C=96 on GLM-4.6V-Flash-9B. RocketKV-MT and SparseVILA fail earlier for a different reason: their retrieval structures (the candidate cache of RocketKV-MT, the full-history buffer of SparseVILA) are allocated outside the pool, and this allocation runs out of GPU

memory at $C{=}64$ on Qwen2.5-VL-7B for both, and on GLM-4.6V-Flash-9B for RocketKV-MT; SparseVILA completes $C{=}96$ on GLM-4.6V-Flash-9B with its process-level peak within 1.6 GB of the 80 GB card. SparseVILA's pool occupancy therefore excludes its history and is not comparable to the other rows. The freed memory is spent on concurrency: at $C{=}96$ CarryKV still occupies less of the pool than Full KV does at its own $C_{\max}$.

Latencies. Figure 10 reports the per-concurrency TTFT and TPOT of both models. Both grow with C for every method. At every C at which a baseline completes, CarryKV matches or slightly improves on TriAttention and lies below every other method on both models.