PRUNE ONCE, ROUTE PER TURN: CONSTRUCTING PERSISTENT VISUAL MEMORY FOR MULTI-TURN VLM INFERENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large vision-language models (VLMs) have achieved impressive performance across diverse multimodal tasks, but their substantial inference cost becomes particularly challenging in multi-turn interaction, where the same visual input is repeatedly queried. Although visual token pruning provides an effective way to reduce inference cost, our empirical analysis reveals that naively extending single-turn pruning strategies to multi-turn scenarios yields two key limitations: **cross-turn pruning mismatch**, in which visual evidence discarded for one query becomes essential for subsequent turns; and **costly query adaptation**, where static retained subsets fail to accommodate evolving queries, whereas recomputing query-aware representations every turn introduces substantial latency overhead. Motivated by these findings, we introduce **PersistRoute**, a simple *training-free* framework that decouples session-level visual memory construction from turn-wise visual Key-Value (KV) routing. Specifically, PersistRoute comprises two complementary components: (1) Session Memory Prune (SMPrune), which conducts one-shot, query-agnostic visual-token pruning to build a diverse visual memory reused throughout the whole session, mitigating cross-turn pruning mismatch; and (2) Dynamic Query-Aware Visual KV Routing (DQVR), which selects query-relevant visual KVs from this persistent memory for each incoming query, enabling adaptive query processing without repeated pruning or memory reconstruction. Evaluated across multiple multi-turn benchmarks and diverse VLM backbones, PersistRoute establishes a more favorable performance-efficiency Pareto frontier compared to prior baselines. Using 50% persistent visual memory with merely 5% routed visual KVs, it preserves 96.2% of the full-model performance on MT-VQA-v2 and 95.6% on MT-GQA, delivering up to $2.54\times$ end-to-end session-level inference speedup.

1 INTRODUCTION

Large vision-language models (VLMs) have emerged as a general-purpose framework that integrates visual perception with language understanding and supports cross-modal reasoning (Bai et al., 2023; Wang et al., 2025; Li et al., 2026b; Wu et al., 2024). Their broad capabilities, however, come at substantial computational cost because visual inputs are typically represented by far more tokens than textual queries. For example, LLaVA-NeXT (Liu et al., 2024b) can process as many as 2,880 visual tokens for a single image, whereas a simple query contains only a handful of text tokens. This token imbalance becomes particularly pronounced in multi-turn interactions, where the same image is repeatedly queried throughout a conversation, causing the visual computation to accumulate across turns and increasing session-level inference latency. Fortunately, token pruning offers a feasible solution by eliminating redundant tokens, thereby reducing inference cost while largely preserving model performance (Chen et al., 2024a).

Existing visual token pruning methods can be broadly categorized as query-agnostic or query-aware. Query-agnostic methods identify redundant tokens from visual features or token statistics, producing a fixed subset independent of the user query (Shang et al., 2025; Yang et al., 2025b; Zhang et al., 2025a). Query-aware methods instead incorporate textual guidance or cross-modal attention to select tokens relevant to the current query (Chen et al., 2024b; Zhang et al., 2025b;c). Despite their

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

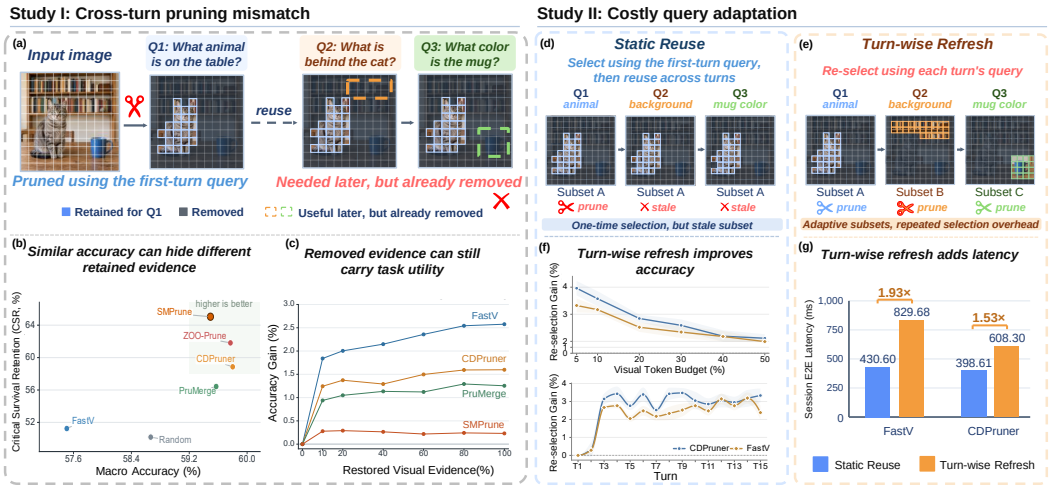

Figure 1: **Pilot studies for multi-turn VLMs inference. Left (a–c):** a subset of tokens pruned for the current query may lose evidence that remains useful for future queries. **Right (d–g):** turn-wise refresh improves query adaptation but introduces repeated selection and additional session latency.

effectiveness, both paradigms have been developed primarily for single-turn inference, making their direct extension to multi-turn conversations problematic. Recent work, MetaCompress (Wang et al., 2026), extends visual token reduction to multi-turn VLMs by learning prompt-agnostic compression conditioned only on the image. Although this design avoids repeated prompt-dependent compression, it keeps the visual representation fixed throughout the conversation, limiting its ability to adapt to the changing focus of successive queries.

To tackle this issue, our analysis identifies two central challenges in extending visual token pruning to multi-turn interactions. First, **cross-turn pruning mismatch** arises when visual evidence removed as irrelevant to one query becomes necessary for a later query. Second, **costly query adaptation** reflects a trade-off between using a static visual representation and recomputing query-dependent representations at every turn. The former cannot accommodate evolving queries, whereas the latter introduces additional inference latency. These challenges are illustrated in Fig. 1 and examined quantitatively through the pilot studies in Sec. 3.

Motivated by these observations, we propose **PersistRoute**, a simple and training-free framework that separates session-level visual memory construction from turn-wise visual routing. PersistRoute consists of two components. **Session Memory Prune (SMPrune)** performs one-shot, query-agnostic visual token pruning to construct a diverse visual memory shared across the entire session, preserving visual evidence that may be useful for later queries. Given this session memory, **Dynamic Query-Aware Visual KV Routing (DQVR)** selects a smaller set of query-relevant visual Key-Values (KVs) for each incoming query, enabling query adaptation without repeatedly pruning or rebuilding the visual memory. In particular, visual KVs that are inactive at the current turn remain stored and can be selected again by subsequent queries. In this way, SMPrune determines what visual evidence remains available across the conversation, while DQVR determines what visual evidence is actively accessed at each turn.

Our contributions are summarized as follows:

- We identify two challenges in extending visual token pruning to multi-turn VLM inference: cross-turn pruning mismatch and costly query adaptation. Our pilot studies further analyze these challenges and motivate preserving broad visual evidence across the conversation while adapting visual access to each query.
- We propose PersistRoute, a simple and training-free framework that separates session-level visual memory construction from turn-wise visual routing. SMPrune constructs a diverse visual memory once for the entire session, while DQVR dynamically routes query-relevant visual KVs from this memory for each incoming query.

- Extensive experiments across three multi-turn benchmarks and five VLM backbones show that PersistRoute retains 96.2% and 95.6% of the original model performance on MT-VQA-v2 and MT-GQA with 50% persistent memory and 5% routed visual KVs. On LLaVA-NeXT-7B, PersistRoute achieves up to a $2.54\times$ session end-to-end speedup, showing a more favorable performance–efficiency Pareto frontier than existing baselines.

2 Related Work

Visual token pruning. Reducing redundant visual tokens has become an effective approach to accelerating VLM inference. Existing training-free methods can be broadly categorized by where token reduction is performed. Some methods prune visual tokens within the language model: FastV (Chen et al., 2024b) uses attention patterns from early LLM layers to remove tokens in subsequent computation, while SparseVLM (Zhang et al., 2025c) performs text-guided, layer-wise token sparsification. Another line reduces visual tokens before they enter the language model. VisionZip (Yang et al., 2025b) combines dominant-token selection with similarity-based merging, whereas CDPruner (Zhang et al., 2025b) jointly models feature diversity and instruction relevance for token selection. Recent methods further perform reduction across multiple stages: IVTP (Huang et al., 2024) combines visual-side pruning with instruction-guided pruning in the LLM, while VScan (Zhang et al., 2026) progressively reduces visual tokens during both visual encoding and language processing. Other representative approaches explore token merging, budget-aware pruning, and importance- or sensitivity-based selection (Shang et al., 2025; Ye et al., 2025; Fang et al., 2026; Li et al., 2026a; Kim et al., 2026; Xing et al., 2025; Yang et al., 2025a; Alvar et al., 2025; Jiang et al., 2025).

Efficient VLM inference and multi-turn interaction. KV cache compression has also been explored specifically for multimodal inference (Wan et al., 2024; 2025; Huang et al., 2025). VL-Cache exploits layer-wise sparsity and modality-aware token importance for adaptive cache allocation (Tu et al., 2025), while MadaKV adapts KV eviction to modality preferences across attention heads (Li et al., 2025). More recently, HybridKV combines hierarchical budget allocation with head-specific compression strategies for multimodal KV caches (Zeng et al., 2026). A growing body of multimodal benchmarks evaluates perception, reasoning, and instruction-following capabilities, yet most focus primarily on single-turn dialogues (Goyal et al., 2017; Hudson & Manning, 2019; Singh et al., 2019; Lu et al., 2022; Liu et al., 2024d; Li et al., 2023a; Yue et al., 2024; Li et al., 2023b; Fu et al., 2025). Beyond single-turn inference, ConvBench evaluates multimodal perception and reasoning across conversational turns (Liu et al., 2024c), while MetaCompress studies prompt-agnostic visual token compression for multi-turn VQA.

3 Pilot Studies: Persistent Memory and Turn-Wise Routing

3.1 Study 1: cross-turn pruning mismatch

We first examine *whether visual tokens discarded for the current query remain useful in later turns*. In Fig. 1(a), the first question concerns the animal on the table, so query-conditioned pruning may remove the background and mug. If the resulting subset is reused, subsequent questions about those regions can no longer access the discarded evidence.

To quantify this risk, we compare macro accuracy with *Critical Survival Retention (CSR)*, which measures how much critical visual evidence survives after pruning, as shown in Fig. 1(b). The computation of CSR is detailed in Appendix E.1. Methods with similar macro accuracy differ substantially in CSR, showing that accuracy alone can mask differences in retained evidence. We then progressively restore discarded visual evidence and measure the resulting accuracy gain. As shown in Fig. 1(c), FastV, CDPruner, and PruMerge improve as more evidence is restored, indicating that some pruned tokens remain useful. These findings suggest that visual token pruning for multi-turn VLMs should preserve evidence beyond the current query, keeping it accessible for later turns.

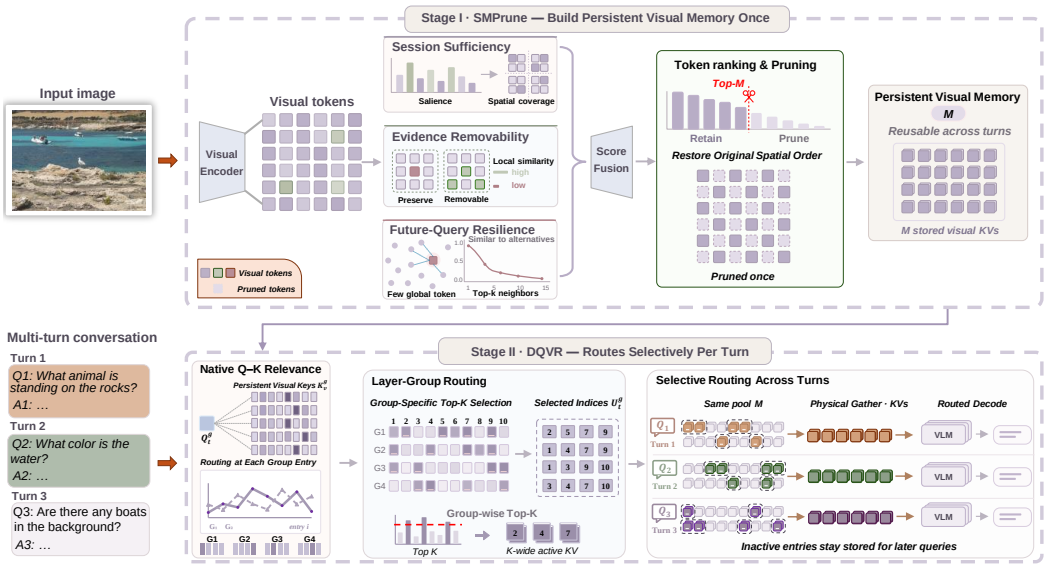

Figure 2: Overview of PersistRoute, which operates in two stages. In Stage I, SMPrune prunes visual tokens once to construct a persistent visual memory shared across the conversation. In Stage II, DQVR selects query-relevant visual KVs from this memory at each turn, while unselected KVs remain available for later queries.

3.2 Study 2: Costly Query Adaptation

If a subset selected for the first query cannot reliably serve later turns, a natural alternative is to refresh the query-conditioned subset at each turn. This motivates our second study on **costly query adaptation**: *whether this adaptation improves accuracy and how much latency it adds?* Figure 1(d)–(g) presents the two strategies and their measured trade-off.

We compare two multi-turn policies using the same query-conditioned pruning method. *Static Reuse* selects a visual subset from the first query and reuses it for later turns. This avoids repeated selection, but the subset can become outdated as the query changes. *Turn-wise Refresh* instead re-selects the visual subset at each turn as the conversation evolves. This allows the retained evidence to follow the current query, but requires repeated selection and visual-state updates.

Figure 1(f) compares *Turn-wise Refresh* with *Static Reuse*. The accuracy gain is larger under tighter visual-token budgets and becomes clearly positive as the query changes across turns, indicating that relevant visual evidence is turn-dependent. However, this improved adaptation comes at a substantial latency cost. As shown in Figure 1(g), turn-wise refresh increases session end-to-end latency from 430.60 ms to 829.68 ms for FastV and from 398.61 ms to 608.30 ms for CDPruner, corresponding to $1.93\times$ and $1.53\times$ higher latency, respectively. These results show that recomputing query-dependent visual subsets at each turn improves adaptation to evolving queries but introduces substantial session-level overhead.

Together, the above two pilot studies identify two requirements for efficient multi-turn VLM inference. Visual evidence must remain available for later queries, while access to it must adapt dynamically, without repeatedly rebuilding the representation. MetaCompress, for example, learns prompt-agnostic compression of input visual tokens, allowing the compressed representation to be reused across turns. Although this also reduces the resulting visual KV cache, MetaCompress does not explicitly select cached visual KVs according to each new query. Its compressed visual KV set therefore remains fixed across turns, leaving further opportunities to reduce per-turn computation and memory access. This distinction motivates separating the evidence retained for the session from the visual KVs accessed for each query. We therefore propose **PersistRoute**, in which **SMPrune** constructs a diverse visual memory once per session, while **DQVR** routes query-relevant visual KVs from that memory at each turn.

4 METHOD

Inspired by the pilot studies above, we view efficient multi-turn visual inference as requiring two complementary capabilities: retaining visual evidence across the session and adapting visual access to each query. As illustrated in Fig. 2, PersistRoute jointly realizes them through Session Memory Prune (SMPrune) and Dynamic Query-Aware Visual KV Routing (DQVR).

Let N denote the original number of visual tokens, M the size of the persistent visual memory after SMPrune, and K the number of visual KVs routed for the current query. PersistRoute follows

$$N \xrightarrow{\text{SMPrune}} M \xrightarrow{\text{DQVR}} K, \qquad K \leq M < N. \tag{1}$$

SMPrune constructs the persistent visual memory once and maintains it throughout the session. For each incoming query, DQVR selectively routes K query-relevant visual KVs from the M persistent entries for autoregressive decoding. Entries not routed at the current turn remain in the persistent memory and can still be selectively routed for later queries.

4.1 SESSION MEMORY PRUNE (SMPRUNE)

SMPrune constructs a compact, query-agnostic visual memory that remains broadly useful throughout the session. Instead of estimating token importance for a specific query, it evaluates each visual token from three complementary perspectives: *Session Sufficiency*, *Evidence Removability*, and *Future-Query Resilience*.

Let $\mathbf{X}_v = [\mathbf{x}_1, \ldots, \mathbf{x}_N] \in \mathbb{R}^{N \times D}$ denote the visual tokens, and let a_i denote an image-only salience score obtained from the frozen vision encoder.

Session Sufficiency. A persistent memory should preserve informative evidence while maintaining broad spatial coverage. We therefore combine global salience with region-aware salience:

$$p_i^S = \mathrm{RankNorm}\left(\alpha\ \mathrm{RankNorm}(a_i) + (1-\alpha)\ \mathrm{RegionRankNorm}(a_i)\right). \tag{2}$$

The global term prioritizes salient evidence, while the regional term prevents retained tokens from concentrating in only a few visual regions.

Evidence Removability. A persistent memory should devote its limited capacity to evidence that cannot be readily substituted by nearby tokens. We therefore measure local replaceability as

$$p_i^R = \mathrm{RankNorm}\left[1 - \max_{j \in \mathcal{N}(i)} \cos(\mathbf{x}_i, \mathbf{x}_j)\right]. \tag{3}$$

Tokens with fewer local substitutes receive higher scores, encouraging SMPrune to retain visually distinctive evidence rather than redundant neighbors.

Future-Query Resilience. Since future queries are unknown when the persistent memory is constructed, SMPrune should preserve visual evidence that cannot be easily recovered from other parts of the image. We therefore measure global replaceability by comparing each token with its most similar visual tokens. Let $\mathcal{T}(i)$ denote the set of visual tokens most similar to $\mathbf{x}_i$. We define

$$p_i^F = \mathrm{RankNorm}\left[1 - \frac{1}{|\mathcal{T}(i)|} \sum_{j \in \mathcal{T}(i)} \cos(\mathbf{x}_i, \mathbf{x}_j)\right]. \tag{4}$$

Tokens with fewer global substitutes receive higher scores, encouraging SMPrune to retain globally distinctive evidence that may be useful under unseen future queries.

Finally, we aggregate the three criteria as

$$F_i = \tau \log \sum_{m \in \{S,R,F\}} \exp\left(\frac{w_m p_i^m}{\tau}\right), \tag{5}$$

and retain the top-M tokens according to F_i. Their original spatial order is restored to form the persistent visual memory, which is constructed once and maintained throughout the session. Detailed definitions and hyperparameters are provided in Appendix A.1.

4.2 Dynamic Query-Aware Visual KV Routing (DQVR)

Given the persistent visual memory constructed by SMPrune, the visual evidence relevant to a query may vary across conversation turns. DQVR therefore performs turn-wise query-adaptive routing, selecting a compact subset of visual KVs from the shared memory for each incoming query. The persistent memory remains unchanged, while the routed subset is refreshed at each turn.

For each incoming query, DQVR uses the native query–key representations of the frozen VLM to measure its relevance to all M persistent visual entries. To reduce routing overhead, consecutive decoder layers are partitioned into groups, with each group sharing the routing decision from its group-entry layer. Let $\mathbf{Q}_t^{(g)}$ denote the current-query representation for group g at turn t, let $\mathcal{P}_t$ denote the set of current-query token positions used for routing, and let $\mathbf{K}_v^{(g)}$ denote the corresponding persistent visual keys. Here, B is the batch size, H the number of attention heads, and d_h the per-head hidden dimension; b, h, q, and i index the batch, attention head, query position, and persistent visual entry, respectively. Using scaled dot-product attention (Vaswani et al., 2017), the relevance of visual entry i is computed as

$$s_{t,i}^{(g)} = \frac{1}{BH|\mathcal{P}_t|} \sum_{b=1}^{B} \sum_{h=1}^{H} \sum_{q \in \mathcal{P}_t} \mathrm{softmax}_i \left(\frac{\mathbf{Q}_{t,b,h,q}^{(g)} \mathbf{K}_{v,b,h,i}^{(g)\top}}{\sqrt{d_h}} \right). \tag{6}$$

Based on these relevance scores, each layer group selects a query-specific active set from the full persistent memory:

$$\mathcal{U}_t^{(g)} = \mathrm{TopK}\left(\{ s_{t,i}^{(g)} \}_{i=1}^{M}, K \right), \qquad |\mathcal{U}_t^{(g)}| = K. \tag{7}$$

The routing decision is refreshed for every incoming query, allowing different turns and layer groups to selectively route different visual evidence from the same persistent memory.

For each layer $\ell \in G_g$, DQVR physically gathers the visual keys and values indexed by $\mathcal{U}_t^{(g)}$:

$$\widehat{\mathbf{K}}_t^{(\ell)} = \mathbf{K}_v^{(\ell)}[:, :, \mathcal{U}_t^{(g)}, :], \qquad \widehat{\mathbf{V}}_t^{(\ell)} = \mathbf{V}_v^{(\ell)}[:, :, \mathcal{U}_t^{(g)}, :]. \tag{8}$$

The resulting K visual KVs are used by the corresponding layer group for the current turn, while the full M-entry persistent memory remains available for subsequent routing. The routing decisions are refreshed when the next query arrives. Detailed layer grouping and cache execution are provided in Appendix A.1.

5 Experiments

5.1 Experimental Setup

Benchmarks and models. We evaluate on three multi-turn benchmarks: MT-VQA-v2, MT-GQA, and ConvBench (Liu et al., 2024c). Following MetaCompress (Wang et al., 2026), MT-VQA-v2 contains 25,000 three-turn conversations constructed from VQA-v2 (Goyal et al., 2017), MT-GQA contains 4,061 three-turn conversations from GQA (Hudson & Manning, 2019), and ConvBench contains 577 multi-turn conversations. We use LLaVA-1.5-7B (Liu et al., 2024a) as the primary backbone and further evaluate LLaVA-1.5-13B (Liu et al., 2024a), LLaVA-NeXT-7B (Liu et al., 2024b), Qwen2.5-VL-7B (Bai et al., 2025) and Qwen3-VL-8B (Qwen Team, 2025). All backbone parameters remain frozen. Additional VQA-v2 and GQA full-session tracks, constructed by grouping all questions associated with the same image into one session, are used for controlled analyses.

Baselines and protocol. We compare against FastV, CDPruner, PruMerge, VisionZip, MetaCompress, and ZOO-Prune, with Full Cache as the uncompressed reference. Main comparisons use decoding-time visual-KV ratios $K/N \in \{5\%, 10\%, 30\%\}$, where K/N denotes the fraction of the original visual KVs that directly participate in subsequent autoregressive decoding. Conventional pruning methods reduce their visual representation to the corresponding ratio according to their native pruning stage, whereas our method retains a 50% persistent visual memory ($M/N = 50\%$) and dynamically selects K entries from this memory at each turn, with K/N set to the target decoding-time visual-KV ratio. The decoding-time visual-KV ratio K/N and the persistent-memory ratio M/N are therefore reported separately. For query-conditioned baselines, selection is recomputed at every turn using the current query.

Table 1: Performance on LLaVA-1.5-7B under matched decoding-time visual-KV ratios K/N. PersistRoute uses a persistent-memory ratio $M/N = 50\%$ and activates K visual KVs for routed decoding at each turn.

Method	Pre-LLM Ret.	In-LLM Ret.	MT-VQA-v2				MT-GQA				ConvBench			
			Acc_1	Acc_2	Acc_3	Avg	Acc_1	Acc_2	Acc_3	Avg	S_1	S_2	S_3	Avg
Decoding-time visual-KV ratio $K/N = 5\%$														
Full Cache (Vanilla)	1.00	1.00	77.43	77.27	77.12	77.27	62.50	63.95	66.02	64.16	3.99	6.07	5.89	5.32
FastV ECCV'24	1.00	0.05	59.16	58.94	59.51	59.20	48.78	52.01	53.66	51.48	3.99	5.03	4.33	4.45
PruMerge ICCV'25	0.05	1.00	64.67	64.66	64.71	64.68	51.81	54.03	56.37	54.07	4.15	6.76	4.33	5.08
CDPruner NeurIPS'25	0.05	1.00	72.88	67.00	66.79	68.89	57.30	56.41	58.66	57.46	**4.33**	6.59	4.95	5.29
VisionZip CVPR'25	0.05	1.00	65.21	64.69	64.65	64.85	52.57	54.59	56.51	54.56	3.64	5.76	4.68	4.69
MetaCompress CVPR'26	0.05	1.00	68.31	68.14	69.53	68.66	54.83	57.12	59.31	57.09	3.99	3.81	4.33	4.04
ZOO-Prune CVPR'26	0.05	1.00	70.84	70.32	70.39	70.52	56.00	58.21	59.12	57.78	3.47	6.24	4.85	4.85
PersistRoute (Ours)	0.50	0.10	**74.06**	**74.45**	**74.45**	**74.32**	**59.52**	**61.17**	**63.31**	**61.33**	3.64	**7.63**	**5.20**	**5.49**
Decoding-time visual-KV ratio $K/N = 10\%$														
Full Cache (Vanilla)	1.00	1.00	77.43	77.27	77.12	77.27	62.50	63.95	66.02	64.16	3.99	6.07	5.89	5.32
FastV ECCV'24	1.00	0.10	66.84	66.67	66.30	66.60	53.58	55.65	57.38	55.54	4.16	5.24	5.49	4.96
PruMerge ICCV'25	0.10	1.00	69.89	70.03	69.53	69.82	54.89	56.56	58.14	56.53	**5.20**	5.11	4.79	5.03
CDPruner NeurIPS'25	0.10	1.00	74.62	71.42	71.28	72.44	58.93	58.56	60.90	59.46	3.99	5.97	5.03	5.00
VisionZip CVPR'25	0.10	1.00	70.73	70.39	70.05	70.39	55.55	57.33	59.10	57.33	4.68	4.45	5.11	4.75
MetaCompress CVPR'26	0.10	1.00	70.92	70.09	71.20	70.74	56.62	58.60	61.26	58.83	4.33	3.99	4.16	4.16
ZOO-Prune CVPR'26	0.10	1.00	73.71	73.67	73.43	73.60	58.46	60.48	61.71	60.22	3.64	6.45	**5.76**	5.28
PersistRoute (Ours)	0.50	0.20	**75.69**	**75.84**	**75.65**	**75.73**	**60.28**	**61.96**	**63.97**	**62.07**	4.51	**7.45**	5.03	**5.66**
Decoding-time visual-KV ratio $K/N = 30\%$														
Full Cache (Vanilla)	1.00	1.00	77.43	77.27	77.12	77.27	62.50	63.95	66.02	64.16	3.99	6.07	5.89	5.32
FastV ECCV'24	1.00	0.30	73.94	73.54	73.41	73.63	57.97	59.52	61.07	59.52	5.03	6.45	5.20	5.56
PruMerge ICCV'25	0.30	1.00	75.22	75.03	75.05	75.10	58.24	60.06	61.44	59.91	4.51	6.07	6.26	5.61
CDPruner NeurIPS'25	0.30	1.00	76.45	75.37	75.57	75.80	60.53	**62.40**	63.51	62.15	**5.41**	7.28	4.28	5.66
VisionZip CVPR'25	0.30	1.00	75.59	75.26	75.48	75.44	59.67	60.85	62.82	61.11	4.51	4.85	6.07	5.14
MetaCompress CVPR'26	0.30	1.00	76.37	76.39	76.29	76.35	59.32	60.85	64.29	61.49	3.99	5.03	4.85	4.62
ZOO-Prune CVPR'26	0.30	1.00	76.12	75.92	75.90	75.98	60.50	61.86	63.80	62.05	3.99	6.80	6.07	5.62
PersistRoute (Ours)	0.50	0.60	**76.75**	**76.72**	**76.55**	**76.67**	**60.75**	62.35	**64.84**	**62.65**	4.16	**7.45**	**6.37**	**5.99**

Pre-LLM Ret. is relative to the original visual sequence and **In-LLM Ret.** to the retained visual representation. For our method, **Pre-LLM Ret.** corresponds to M/N and **In-LLM Ret.** to K/M, so their product gives the decoding-time visual-KV ratio K/N.

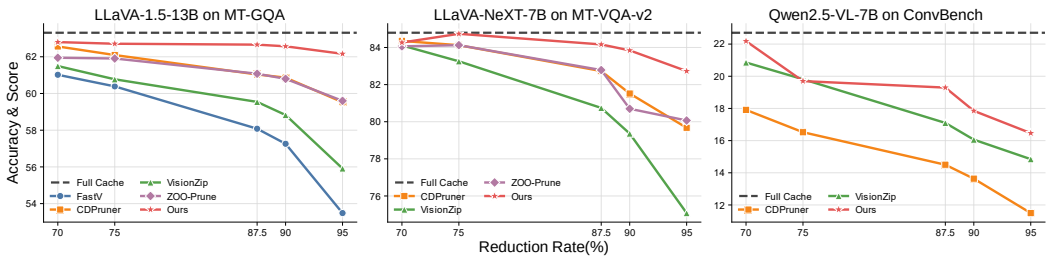

Figure 3: Cross-backbone performance under varying decoding-time visual-KV ratios.

5.2 Main Results

Results on LLaVA-1.5-7B. Table 1 compares PersistRoute with representative visual token pruning methods on LLaVA-1.5-7B under decoding-time visual-KV ratios $K/N \in \{5\%, 10\%, 30\%\}$. PersistRoute achieves the best average performance among compressed methods across all three benchmarks and decoding-time visual-KV ratios. The advantage is particularly clear under aggressive compression. At a decoding-time visual-KV ratio of $K/N = 5\%$, PersistRoute retains 96.2% and 95.6% of Full Cache performance on MT-VQA-v2 and MT-GQA, while outperforming the strongest compared baseline by 3.80 and 3.55 percentage points. At $K/N = 10\%$, the gains remain 2.13 and 1.85 points on the two benchmarks. As K/N increases to 30%, PersistRoute further approaches Full Cache, reaching 76.67% on MT-VQA-v2 and 62.65% on MT-GQA.

Results across different VLM backbones. We further evaluate PersistRoute on LLaVA-1.5-13B with MT-GQA, LLaVA-NeXT-7B with MT-VQA-v2, and Qwen2.5-VL-7B with ConvBench. As shown in Fig. 3, PersistRoute consistently preserves strong performance across different VLM backbones as the decoding-time visual-KV ratio decreases. Complete results for LLaVA-1.5-13B, LLaVA-NeXT-7B, Qwen2.5-VL-7B, and the more recent Qwen3-VL-8B across all three benchmarks and visual-KV ratios are provided in Appendix D.5. These results further demonstrate that

Table 2: Efficiency on VQA-v2 Full-Session ($K/N = 10\%$; Base uncompressed). Blue: Static Reuse; orange: Turn-wise Refresh; green: DQVR.

Method	Visual Update	TTFT (ms) ↓	Prefill (ms) ↓	Decode (ms) ↓	Session E2E (ms) ↓	Peak Mem. (GB) ↓	Acc. (%) ↑
LLaVA-1.5-7B							
Base	Static Reuse	118.00 (1.00×)	189.03 (1.00×)	200.49 (1.00×)	614.10 (1.00×)	16.660 (reference)	59.23
	Turn-wise Refresh	115.54 (1.02×)	489.12 (0.39×)	202.66 (0.99×)	1007.00 (0.61×)	16.467 (↓1.2%)	59.23
CDPruner	Static Reuse	68.10 (1.73×)	**115.65** (1.63×)	172.98 (1.16×)	398.61 (1.54×)	15.290 (↓8.2%)	54.26
	Turn-wise Refresh	65.44 (1.80×)	213.44 (0.89×)	172.68 (1.16×)	608.30 (1.01×)	15.310 (↓8.1%)	57.14
VisionZip	Static Reuse	52.39 (2.25×)	118.70 (1.59×)	171.60 (1.17×)	**361.21** (1.70×)	**15.020** (↓9.8%)	54.13
	Turn-wise Refresh	**51.93** (2.27×)	208.04 (0.91×)	170.65 (1.17×)	535.33 (1.15×)	15.110 (↓9.3%)	54.13
ZOO-Prune	Static Reuse	84.72 (1.39×)	121.53 (1.56×)	171.50 (1.17×)	402.52 (1.53×)	16.330 (↓2.0%)	55.81
	Turn-wise Refresh	83.38 (1.42×)	215.98 (0.88×)	**168.67** (1.19×)	700.26 (0.88×)	16.650 (↓0.1%)	55.81
MetaCompress	Static Reuse	79.05 (1.49×)	141.86 (1.33×)	173.25 (1.16×)	432.47 (1.42×)	15.220 (↓8.6%)	56.57
PersistRoute	DQVR	86.35 (1.37×)	137.93 (1.37×)	181.57 (1.10×)	456.12 (1.35×)	15.470 (↓7.1%)	**58.22**
LLaVA-NeXT-7B							
Base	Static Reuse	222.32 (1.00×)	780.12 (1.00×)	243.87 (1.00×)	1800.91 (1.00×)	25.420 (reference)	59.80
	Turn-wise Refresh	226.51 (0.98×)	1644.08 (0.47×)	247.38 (0.99×)	2771.64 (0.65×)	27.970 (↑10.0%)	59.80
CDPruner	Static Reuse	135.36 (1.64×)	**148.09** (5.27×)	168.47 (1.45×)	522.29 (3.45×)	15.950 (↓37.3%)	56.53
	Turn-wise Refresh	134.85 (1.65×)	360.16 (2.17×)	169.14 (1.44×)	972.87 (1.85×)	**15.870** (↓37.6%)	57.55
VisionZip	Static Reuse	**123.91** (1.79×)	151.78 (5.14×)	177.06 (1.38×)	**520.33** (3.46×)	16.000 (↓37.1%)	56.00
	Turn-wise Refresh	124.52 (1.79×)	357.33 (2.18×)	171.64 (1.42×)	918.11 (1.96×)	16.620 (↓34.6%)	56.00
ZOO-Prune	Static Reuse	198.19 (1.12×)	160.46 (4.86×)	**167.24** (1.46×)	664.59 (2.71×)	16.360 (↓35.6%)	57.60
	Turn-wise Refresh	200.36 (1.11×)	367.72 (2.12×)	179.47 (1.36×)	1359.90 (1.32×)	16.990 (↓33.2%)	57.60
MetaCompress	Static Reuse	178.05 (1.25×)	209.31 (3.73×)	181.79 (1.34×)	693.03 (2.60×)	17.090 (↓32.8%)	56.47
PersistRoute	DQVR	191.99 (1.16×)	160.37 (4.86×)	202.74 (1.20×)	708.43 (2.54×)	17.370 (↓31.7%)	**58.75**

the combination of persistent-memory construction and turn-wise selective routing remains effective across different VLM architectures and evaluation settings.

5.3 Efficiency Analysis

We evaluate practical efficiency on VQA-v2 Full-Session at a 10% decoding-time visual-KV ratio, with Base left uncompressed. *Static Reuse* constructs the visual state once and reuses it across turns, while *Turn-wise Refresh* recomputes the method-specific visual state for each query. PersistRoute instead retains a 50% persistent memory and uses DQVR to selectively route 20% of the persistent visual KVs at each turn. All speedups and memory reductions in Table 2 are measured against Base with Static Reuse under the same GPU allocation for each backbone.

As shown in Table 2, compared with Static Reuse, PersistRoute incurs a moderate increase in session latency but substantially improves accuracy, outperforming the strongest static compressed baseline by 1.65 points on LLaVA-1.5-7B and 1.15 points on LLaVA-NeXT-7B. Compared with Turn-wise Refresh, it avoids repeated visual state reconstruction and achieves both lower latency and higher accuracy than all compressed refresh baselines. PersistRoute reaches 58.22% accuracy with 456.12 ms session latency on LLaVA-1.5-7B and 58.75% with 708.43 ms on LLaVA-NeXT-7B, corresponding to $1.35\times$ and $2.54\times$ speedups over the uncompressed Static Reuse reference. These results place PersistRoute on a more favorable performance–efficiency Pareto frontier among the compared strategies. Extended results at $K/N \in \{5\%, 10\%, 30\%\}$ are provided in Appendix C.

5.4 Ablation Studies

Component analysis. We factorize persistent-memory construction and turn-wise routing on VQA-v2 Full-Session with LLaVA-NeXT-7B. We compare Full persistent memory ($M = N$) against SMPrune ($M/N = 50\%$), and Static-Q1 against DQVR for routing. As shown in Table 3, DQVR consistently improves over Static-Q1 under both persistent-memory settings, and the combination of SMPrune and DQVR achieves the best performance across all evaluated ratios. These results support the complementary roles of persistent-memory construction and turn-wise selective routing. Detailed efficiency results for this factorized ablation are provided in Appendix D.4.

Controlled routing analysis. On MT-GQA with LLaVA-1.5-7B, we fix the persistent visual memory ($M/N = 50\%$), the decoding-time visual-KV ratio K/N, and the routed execution path, varying only the $M \to K$ routing rule. As shown in Table 4, DQVR achieves the highest average accuracy

Table 3: Memory construction and routing.

Persistent	Access	Decoding-Time Visual-KV Ratio K/N		
		5%	12.5%	25%
Full	Static-Q1	57.35	57.79	58.66
Full	DQVR	57.65	58.54	58.98
SMPrune	Static-Q1	57.69	58.25	59.67
SMPrune	**DQVR**	**58.34**	**59.16**	**59.90**

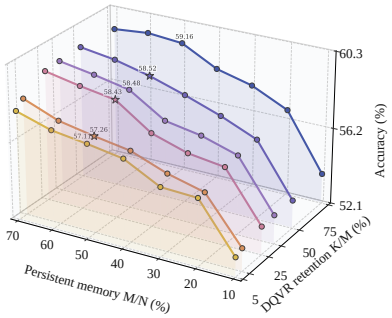

Figure 4: Accuracy vs. M/N and K/M.

Table 4: Routing with fixed memory.

Routing Rule	5%	10%	30%
Random	59.23	60.54	60.83
Static-Q1	59.89	60.33	61.25
Global Query-Aware Top-K	60.79	61.60	62.24
DQVR	**61.33**	**62.07**	**62.65**

Table 5: SMPrune criterion ablation.

Metric	S only	w/o S	w/o R	w/o F	**Full**
Macro Acc. ↑	58.85	58.85	59.02	58.94	**59.49**
Utility ↑	56.78	57.42	57.59	57.01	**57.75**
CSR ↑	63.19	64.36	64.65	63.75	**65.04**
Future Drop ↓	0.73	0.85	0.49	0.68	**0.48**

at all three decoding-time visual-KV ratios, outperforming Random, Static-Q1, and Global Query-Aware Top-K; the gains over Static-Q1 range from 1.40 to 1.74 percentage points and those over the global query-aware control from 0.41 to 0.54 points. Static-Q1 and DQVR use identical Turn-1 routing decisions by construction, while the gains emerge on later turns, showing that adapting visual selection to the current query improves later-turn performance. Detailed controls and turn-wise results are provided in Appendix D.1.

SMPrune criterion analysis. At $M/N = 50\%$ with DQVR disabled, Table 5 compares full SMPrune with S-only and leave-one-out variants. The full formulation achieves the strongest overall performance across the evaluated metrics. Removing S causes the largest Macro Accuracy drop (0.64 points), while removing F produces the largest degradation in Utility Capture and Critical Survival Retention (CSR). Removing R leads to smaller but consistent performance losses, and the S-only variant remains clearly below the full formulation. These results show that the three criteria provide complementary signals for constructing a robust persistent visual memory.

Persistent-memory and DQVR ratio analysis. On VQA-v2 Full-Session with LLaVA-1.5-7B, we independently vary the persistent-memory ratio M/N and the DQVR retention ratio K/M. As shown in Fig. 4, over nonzero routing settings, increasing M/N from 10% to 70% improves accuracy by 4.8–5.7 points, while varying K/M from 5% to 75% changes accuracy by 0.57–2.05 points at a fixed M/N. The $K = 0$ control in Appendix D.6 further shows that increasing K/M from 0% to 5% yields gains of 5.92–10.71 points across different persistent-memory ratios. These results reveal a clear interaction between memory capacity and routing: expanding the persistent pool steadily improves the available evidence basis, whereas introducing even a small routed subset captures much of the benefit of query-conditioned routing.

6 Conclusion

In this paper, we present **PersistRoute** for efficient multi-turn VLM inference, where visual evidence must remain available across a conversation while access adapts to each query. SMPrune constructs a shared visual memory once, and DQVR selects query-relevant visual KVs from that memory at each turn. Together, they follow a simple principle: *prune once, route per turn*. Across three multi-turn benchmarks and five VLM backbones, PersistRoute achieves a more favorable performance-efficiency trade-off than existing baselines. More broadly, decoupling persistent visual memory from turn-wise access offers a new design direction for efficient multi-turn VLM inference and may inspire future work on adaptive visual memory.

AI USE STATEMENT

In this work, we used generative AI tools in three capacities. First, for writing: AI tools assisted in drafting selected parts of the manuscript and provided suggestions for revising and refining the writing. Second, for retrieval and discovery: AI tools assisted in identifying, retrieving, and summarizing relevant prior work and related literature. Third, for research ideation and execution: AI tools assisted in discussing and refining research ideas proposed by the authors, supported experimental code implementation and debugging, helped check the consistency of experimental results, and provided suggestions for figure design. The research ideas, experimental designs, scientific decisions, and final figures were developed by the authors. We have reviewed all AI-assisted work and take full responsibility for the final content of this paper, including all text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

We provide detailed methodological and experimental specifications to facilitate reproducibility. Appendix A.1 describes the implementation of SMPrune and DQVR, including key hyperparameters, routing and cache execution, and pseudocode. Appendix B further documents the multi-turn evaluation protocol, efficiency profiling setup, hardware configuration, and other experimental settings. An anonymized source-code archive is submitted as supplementary material.

REFERENCES

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

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. Qwen-vl: A frontier large vision-language model with versatile abilities. *CoRR*, abs/2308.12966, 2023.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan 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. *CoRR*, abs/2502.13923, 2025.

Jieneng Chen, Luoxin Ye, Ju He, Zhaoyang Wang, Daniel Khashabi, and Alan L. Yuille. Efficient large multi-modal models via visual context compression. In *Advances in Neural Information Processing Systems 37*, 2024a.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, volume 15139 of *Lecture Notes in Computer Science*, pp. 19–35. Springer, 2024b.

Zhengyao Fang, Pengyuan Lyu, Chengquan Zhang, Guangming Lu, Jun Yu, and Wenjie Pei. Prune redundancy, preserve essence: Vision token compression in vlms via synergistic importance-diversity. In *International Conference on Learning Representations*, pp. 51976–51999, 2026.

Chaoyou Fu, Peixian Chen, Yunhang Shen, Yulei Qin, Mengdan Zhang, Xu Lin, Jinrui Yang, Xiawu Zheng, Ke Li, Xing Sun, Yunsheng Wu, Rongrong Ji, Caifeng Shan, and Ran He. MME: A comprehensive evaluation benchmark for multimodal large language models. In *Advances in Neural Information Processing Systems 38*, 2025.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6325–6334, 2017.

Kai Huang, Hao Zou, Ye Xi, Bochen Wang, Zhen Xie, and Liang Yu. IVTP: instruction-guided visual token pruning for large vision-language models. In *European Conference on Computer Vision*, volume 15075 of *Lecture Notes in Computer Science*, pp. 214–230. Springer, 2024.

Kai Huang, Hao Zou, Bochen Wang, Ye Xi, Zhen Xie, and Hao Wang. Aircache: Activating inter-modal relevancy KV cache compression for efficient large vision-language model inference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 23958–23967. IEEE, 2025.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709. Computer Vision Foundation / IEEE, 2019.

Yutao Jiang, Qiong Wu, Wenhao Lin, Wei Yu, and Yiyi Zhou. What kind of visual tokens do we need? training-free visual token pruning for multi-modal large language models from the perspective of graph. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4075–4083. AAAI Press, 2025.

Youngeun Kim, Youjia Zhang, Huiling Liu, Aecheon Jung, Sunwoo Lee, and Sungeun Hong. Zoo-prune: Training-free token pruning via zeroth-order gradient estimation in vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 39572–39582. Computer Vision Foundation, 2026.

Ao Li, Yuxiang Duan, Jinghui Zhang, Congbo Ma, Yutong Xie, Gustavo Carneiro, Mohammad Yaqub, and Hu Wang. Transprune: Token transition pruning for efficient large vision-language model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 39529–39538. Computer Vision Foundation, 2026a.

Bohao Li, Rui Wang, Guangzhi Wang, Yuying Ge, Yixiao Ge, and Ying Shan. Seed-bench: Benchmarking multimodal llms with generative comprehension. *CoRR*, abs/2307.16125, 2023a.

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

Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 48(3):3530–3543, 2026b.

Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 292–305. Association for Computational Linguistics, 2023b.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26286–26296. IEEE, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. LLaVA-NeXT: Improved reasoning, ocr, and world knowledge, January 2024b. URL `https://llava-vl.github.io/blog/2024-01-30-llava-next/`.

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 *Advances in Neural Information Processing Systems 37*, 2024c.

Yuan Liu, Haodong Duan, Yuanhan Zhang, Bo Li, Songyang Zhang, Wangbo Zhao, Yike Yuan, Jiaqi Wang, Conghui He, Ziwei Liu, Kai Chen, and Dahua Lin. Mmbench: Is your multi-modal model an all-around player? In *European Conference on Computer Vision*, volume 15064 of *Lecture Notes in Computer Science*, pp. 216–233. Springer, 2024d.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems 35*, 2022.

Qwen Team. Qwen3-VL technical report. *CoRR*, abs/2511.21631, 2025.

Yuzhang Shang, Mu Cai, Bingxin Xu, Yong Jae Lee, and Yan Yan. Llava-prumerge: Adaptive token reduction for efficient large multimodal models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 22857–22867. IEEE, 2025.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8317–8326. Computer Vision Foundation / IEEE, 2019.

Dezhan Tu, Danylo Vashchilenko, Yuzhe Lu, and Panpan Xu. Vl-cache: Sparsity and modality-aware KV cache compression for vision-language model inference acceleration. In *International Conference on Learning Representations*. OpenReview.net, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pp. 5998–6008, 2017.

Zhongwei Wan, Ziang Wu, Che Liu, Jinfa Huang, Zhihong Zhu, Peng Jin, Longyue Wang, and Li Yuan. LOOK-M: look-once optimization in KV cache for efficient multimodal long-context inference. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, volume EMNLP 2024 of *Findings of ACL*, pp. 4065–4078. Association for Computational Linguistics, 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 *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 2485–2497. Association for Computational Linguistics, 2025.

Weizhi Wang, Yu Tian, Linjie Yang, Heng Wang, and Xifeng Yan. Open-qwen2vl: Compute-efficient pre-training of fully-open multimodal llms on academic resources. *CoRR*, abs/2504.00595, 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24727–24737, June 2026.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, Zhenda Xie, Yu Wu, Kai Hu, Jiawei Wang, Yaofeng Sun, Yukun Li, Yishi Piao, Kang Guan, Aixin Liu, Xin Xie, Yuxiang You, Kai Dong, Xingkai Yu, Haowei Zhang, Liang Zhao, Yisong Wang, and Chong Ruan. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *CoRR*, abs/2412.10302, 2024.

Long Xing, Qidong Huang, Xiaoyi Dong, Jiajie Lu, Pan Zhang, Yuhang Zang, Yuhang Cao, Conghui He, Jiaqi Wang, Feng Wu, and Dahua Lin. Conical visual concentration for efficient large vision-language models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 14593–14603. Computer Vision Foundation / IEEE, 2025.

Cheng Yang, Yang Sui, Jinqi Xiao, Lingyi Huang, Yu Gong, Chendi Li, Jinghua Yan, Yu Bai, Ponnuswamy Sadayappan, Xia Hu, and Bo Yuan. Topv: Compatible token pruning with inference time optimization for fast and low-memory multimodal vision language model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19803–19813. Computer Vision Foundation / IEEE, 2025a.

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 *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 19792–19802. Computer Vision Foundation / IEEE, 2025b.

Weihao Ye, Qiong Wu, Wenhao Lin, and Yiyi Zhou. Fit and prune: Fast and training-free visual token pruning for multi-modal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 22128–22136. AAAI Press, 2025.

Xiang Yue, Yuansheng Ni, Tianyu Zheng, Kai Zhang, Ruoqi Liu, Ge Zhang, Samuel Stevens, Dongfu Jiang, Weiming Ren, Yuxuan Sun, Cong Wei, Botao Yu, Ruibin Yuan, Renliang Sun, Ming Yin, Boyuan Zheng, Zhenzhu Yang, Yibo Liu, Wenhao Huang, Huan Sun, Yu Su, and Wenhu Chen. MMMU: A massive multi-discipline multimodal understanding and reasoning benchmark for expert AGI. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9556–9567. IEEE, 2024.

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

Ce Zhang, Kaixin Ma, Tianqing Fang, Wenhao Yu, Hongming Zhang, Zhisong Zhang, Haitao Mi, and Dong Yu. Vscan: Rethinking visual token reduction for efficient large vision-language models. *Trans. Mach. Learn. Res.*, 2026, 2026.

Qizhe Zhang, Aosong Cheng, Ming Lu, Renrui Zhang, Zhiyong Zhuo, Jiajun Cao, Shaobo Guo, Qi She, and Shanghang Zhang. Beyond text-visual attention: Exploiting visual cues for effective token pruning in vlms. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 20857–20867. IEEE, 2025a.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in mllms. In *Advances in Neural Information Processing Systems 38*, 2025b.

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 *International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*. PMLR / OpenReview.net, 2025c.

A APPENDIX

A.1 IMPLEMENTATION DETAILS

This appendix describes the implementation details of SMPrune and DQVR used in the experiments. PersistRoute operates directly on the frozen VLM with training-free memory construction and routing.

A.1.1 SMPRUNE IMPLEMENTATION

Visual-token layout. For LLaVA-1.5 (Liu et al., 2024a), the original visual representation contains $N = 576$ tokens arranged on a 24×24 grid. SMPrune scores all N visual tokens and retains the top-M tokens for persistent visual-memory construction. Unless otherwise specified in the corresponding experiment, M is determined by the persistent-memory ratio used in that experiment rather than being fixed as part of the method definition.

For the other backbones, SMPrune follows their native visual-token layouts. LLaVA-NeXT contains a 24×24 global-view block and a 48×48 local-view block, with regional Session Sufficiency and local Evidence Removability computed independently within the two blocks. Qwen2.5-VL and Qwen3-VL use dynamically sized visual grids with a 2×2 spatial merge. Given a pre-merge patch grid of size $H \times W$, SMPrune operates on the resulting $(H/2) \times (W/2)$ grid, where the same regional and local criteria are applied.

Upstream visual salience. For the canonical LLaVA-1.5 implementation, the final vision self-attention map is

$$\mathbf{A}^{\text{vis}} \in \mathbb{R}^{B \times H \times (N+1) \times (N+1)},$$

where token 0 is the CLS token and tokens $1, \ldots, N$ are patch tokens. The image-only salience used by SMPrune is

$$a_i = \frac{1}{BH} \sum_{b=1}^{B} \sum_{h=1}^{H} A^{\text{vis}}_{b,h,0,i}, \qquad i = 1, \ldots, N. \tag{9}$$

Thus, the score averages over batches and attention heads and takes the attention from the CLS query to the patch keys. For LLaVA-1.5, $N = 576$. The salience is obtained from the image-only vision-tower forward pass and is therefore query-independent.

When vision attention is unavailable, the implementation uses the following fallback order. First, it uses the norm of the raw visual feature $\mathbf{r}_i$:

$$a_i = \|\mathbf{r}_i\|_2. \tag{10}$$

If the raw visual feature is also unavailable, it uses the norm of the projected visual feature $\mathbf{p}_i$:

$$a_i = \|\mathbf{p}_i\|_2. \tag{11}$$

The fallback order is therefore vision attention, raw visual-feature norm, and projected visual-feature norm, providing query-independent salience across supported implementations.

The CLS-to-patch definition above is specific to the canonical LLaVA-1.5 implementation. For LLaVA-NeXT (Liu et al., 2024b), we use the final vision-layer CLS-to-patch attention averaged across heads, preserving the global-view scores and mapping the four local-view scores to the 48×48 local raster. For Qwen2.5-VL (Bai et al., 2025), we use the received attention from the last full-attention vision block, averaged across heads and summed over queries, followed by 2×2 spatial-merge averaging and restoration to the merged visual-grid order. For Qwen3-VL (Qwen Team, 2025), we compute received attention from the final vision block in the same way, averaging across heads and summing over queries, and then average each 2×2 merge group to obtain salience scores aligned with the merged visual-token sequence. All salience signals are computed from the image-only vision encoder and remain query-independent. The subsequent S/R/F scoring, normalization, fusion, and Top-M selection are shared across backbones.

Session Sufficiency. Using the salience scores a_i above, the Session Sufficiency score is implemented as

$$p_i^S = \mathrm{RankNorm}\left(0.7\ \mathrm{RankNorm}(a_i) + 0.3\ \mathrm{RegionRankNorm}(a_i)\right). \tag{12}$$

The regional term is computed over a 6×6 spatial partition of the 24×24 visual-token grid and provides a soft regional coverage prior. Because this score does not use a conversation query, the selection remains query-agnostic.

Evidence Removability. We use a valid spatial eight-neighborhood $\mathcal{N}_8(i)$ and consider only neighbors that lie within the visual-token grid:

$$p_i^R = \mathrm{RankNorm}\left[1 - \max_{j \in \mathcal{N}_8(i)} \cos(\mathbf{x}_i, \mathbf{x}_j)\right]. \tag{13}$$

A larger score indicates that token i has fewer local substitutes and therefore has a higher removal risk.

Future-Query Resilience. For each token, let $\mathcal{T}_k(i)$ contain the k most similar non-self visual tokens. The implementation uses

$$k = \min(5, N-1), \qquad p_i^F = \mathrm{RankNorm}\left[1 - \frac{1}{k} \sum_{j \in \mathcal{T}_k(i)} \cos(\mathbf{x}_i, \mathbf{x}_j)\right]. \tag{14}$$

This score provides a query-independent global-uniqueness proxy, assigning higher scores to tokens with fewer globally similar alternatives.

Fusion and selection. The three normalized criteria are combined with

$$(w_S, w_R, w_F) = (0.5, 0.2, 0.3), \qquad \tau = 0.5, \tag{15}$$

using the deterministic retention score

$$F_i = \tau \log \sum_{m \in \{S,R,F\}} \exp\left(\frac{w_m p_i^m}{\tau}\right). \tag{16}$$

We retain the top-M tokens according to F_i, restore them to their original spatial order, and use them to construct the persistent visual memory.

A.1.2 DQVR Implementation

Layer grouping. LLaVA-1.5-7B has 32 decoder layers indexed from 0 to 31. We use four consecutive layer groups,

$$G_1 = [0, 7], \qquad G_2 = [8, 15], \qquad G_3 = [16, 23], \qquad G_4 = [24, 31],$$

with group-entry routing layers

$$\{r_g\}_{g=1}^4 = \{0, 8, 16, 24\}.$$

At the entry of each group, the current-query representation produces one visual selection shared by all layers in that group. Different groups may select different entries from the same persistent visual memory.

Query representation. At the entry layer r_g of each group, the current-query hidden states are available before the group's self-attention is evaluated. DQVR follows the frozen backbone's native path: current-query hidden state $\rightarrow$ layer input normalization $\rightarrow$ selection of current user-query token positions $\rightarrow$ native $q_{\mathrm{proj}} \rightarrow$ native positional encoding $\rightarrow$ query representation. The resulting $\mathbf{Q}_t^{(g)}$ is used to score the persistent visual keys before the current layer group performs its normal attention computation.

Persistent visual keys. The visual keys used for routing come from the persistent visual KV cache constructed after SMPrune. At group entry r_g, DQVR scores the current-query representation against all M cached persistent visual keys at that layer. The resulting Top-K indices are then used to gather the corresponding visual keys and values for every layer in G_g.

Query-key relevance. Let $\mathcal{P}_t$ denote the current-query token positions used for routing. For group g at turn t, the relevance of persistent visual entry i is computed as

$$s_{t,i}^{(g)} = \frac{1}{BH|\mathcal{P}_t|} \sum_{b,h,q} \mathrm{softmax}_i \left(\frac{\mathbf{Q}_{t,b,h,q}^{(g)} \mathbf{K}_{v,b,h,i}^{(g)\top}}{\sqrt{d_h}} \right). \tag{17}$$

The softmax is taken over the visual-entry index i for each query token and head. We then aggregate over query-token positions and attention heads and select the group-specific active index set

$$\mathcal{U}_t^{(g)} = \mathrm{TopK}\left(\{s_{t,i}^{(g)}\}_{i=1}^{M}, K \right).$$

Thus, each group performs one Top-K selection over the full M-entry persistent pool at its entry layer. The selected indices determine the active visual KVs used by that group.

Execution order. DQVR performs routing at the entry of each decoder group. At group g, the current hidden states provide the native query representation $\mathbf{Q}_t^{(g)}$, which is matched against the M persistent visual keys at layer r_g. The resulting relevance scores determine a group-specific Top-K index set $\mathcal{U}_t^{(g)}$. DQVR then gathers the corresponding visual keys and values for every layer in G_g, after which the group performs its normal attention computation using only these K active visual KVs.

The execution order within each group is therefore

$$\text{group-entry query representation} \;\rightarrow\; \text{routing scores over } M$$
$$\rightarrow\; \text{Top-}K \text{ selection} \;\rightarrow\; \text{physical gather of } K \;\rightarrow\; \text{layer-group attention with } K.$$

The groups are processed sequentially, so the hidden states entering a later group already incorporate the routed computation of preceding groups. For LLaVA-1.5-7B, routing is performed at layers 0, 8, 16, and 24 for groups G_1, G_2, G_3, and G_4.

Physical KV activation. Top-K selection is followed by physical gathering of the selected persistent entries. For a layer $\ell \in G_g$, the group-specific active visual KVs are

$$\widehat{\mathbf{K}}_t^{(\ell)} = \mathbf{K}_v^{(\ell)}[:,:,\mathcal{U}_t^{(g)},:], \qquad \widehat{\mathbf{V}}_t^{(\ell)} = \mathbf{V}_v^{(\ell)}[:,:,\mathcal{U}_t^{(g)},:].$$

The number of visual KVs participating in each layer group's attention is therefore K, while the original M-entry persistent visual memory remains intact. Entries not selected by one group remain available to other groups in the same turn and to routing decisions in later turns.

Routing frequency. DQVR refreshes the routing decisions once for each incoming query. During current-query processing, each group performs one Top-K selection at its entry layer. The resulting group-specific selections are then reused throughout the autoregressive generation of that turn and refreshed when the next query arrives.

Cache lifecycle. The visual memory constructed by SMPrune remains persistent across conversation turns. For each incoming query, DQVR derives one active index set for each layer group and gathers the corresponding visual KVs from the persistent pool. These group-specific active visual KVs are local to the current turn, while the underlying M-entry persistent visual memory remains unchanged. Conversation history follows the same textual-context and cache protocol for all evaluated methods.

Complexity. For turn t, let A_t be the number of current appended query/prompt tokens, Q_t the number of query tokens used for routing, S_t the number of generated response tokens, and G the number of layer groups. Ignoring shared textual-attention terms, the leading visual-KV costs are

$$\mathcal{C}_{\mathrm{route}}(t) = \mathcal{O}\left(GHQ_t M d_h\right), \tag{18}$$
$$\mathcal{C}_{\mathrm{prefill}}(t) = \mathcal{O}\left(LHA_t K d_h\right), \tag{19}$$
$$\mathcal{C}_{\mathrm{decode}}(t) = \mathcal{O}\left(LHS_t K d_h\right). \tag{20}$$

Routing and current-query processing are interleaved across layer groups: each group-entry query representation is scored against the M persistent visual keys once, after which that group's attention uses only the selected K visual KVs. The M-dependent term therefore appears only in the group-level routing computation, while visual-KV participation during current-query processing and autoregressive generation scales with K.

A.1.3 PERSISTROUTE PSEUDOCODE

Algorithm 1 summarizes SMPrune and DQVR with the execution order described above. The image is encoded once, and each query is processed only when its conversation turn begins. The persistent visual cache remains intact throughout the conversation.

Algorithm 1 PersistRoute: SMPrune and DQVR

Require: Image I; frozen VLM; budgets M and $K \leq M$; layer groups $\{G_g, r_g\}_{g=1}^{G}$.
Output: A response a_t to each incoming query q_t.

Stage I: SMPrune — construct persistent visual memory
Obtain visual tokens $\mathbf{X}_v$ and image-only salience from I
Compute p_i^S, p_i^R, and p_i^F using Eqs. 2–4
Fuse the criteria into F_i using Eq. 5; select the top-M indices
Restore the selected visual tokens to their original spatial order
Construct persistent visual KVs $\{\mathbf{K}_v^{(\ell)}, \mathbf{V}_v^{(\ell)}\}_{\ell=0}^{L-1}$
Stage II: DQVR — selectively route visual KVs per turn
Initialize the conversation state
for each incoming query q_t **do**
  for each layer group g in order **do**
    Obtain $\mathbf{Q}_t^{(g)}$ from the current hidden states at group-entry layer r_g
    Score all M persistent visual keys using Eq. 6
    Select $\mathcal{U}_t^{(g)} \leftarrow \mathrm{TopK}(\{s_{t,i}^{(g)}\}_{i=1}^{M}, K)$
    Gather the selected visual KVs for every $\ell \in G_g$ using Eq. 8
    Process the current-query states through G_g using the routed visual KVs
  end for
  Autoregressively decode a_t while reusing $\{\mathcal{U}_t^{(g)}\}_{g=1}^{G}$
  Update the conversation state with (q_t, a_t); retain all M persistent entries
end for

B ADDITIONAL EXPERIMENTAL DETAILS

B.1 MULTI-TURN PROTOCOL.

The three-turn MT-VQA-v2 (Goyal et al., 2017) and MT-GQA (Hudson & Manning, 2019) benchmarks follow the construction protocol of MetaCompress (Wang et al., 2026). For the Full-Session setting, all questions associated with the same image are grouped into one variable-length session. Previous model-generated answers are retained as conversation history, while ground-truth answers are used only for evaluation and future questions remain unavailable during inference.

B.2 EFFICIENCY PROFILING PROTOCOL

We measure practical efficiency on the VQA-v2 Full-Session track. The main comparison in Table 2 uses a decoding-time visual-KV ratio of $K/N = 10\%$, with Base left uncompressed. We further evaluate $K/N \in \{5\%, 10\%, 30\%\}$ in the extended efficiency comparison shown in Fig. 5. For CDPruner, VisionZip, and ZOO-Prune, we evaluate both Static Reuse and Turn-wise Refresh, while MetaCompress is reported with Static Reuse. PersistRoute retains a persistent-memory ratio of $M/N = 50\%$ and uses DQVR to route K visual KVs from the persistent memory at each turn.

For LLaVA-1.5-7B, the original visual representation contains $N = 576$ tokens, with $M = 288$ persistent entries under $M/N = 50\%$; the 5%, 10%, and 30% budgets correspond to $K = 29$, 58, and 173. For LLaVA-NeXT-7B, $N = 2880$ and $M = 1440$, with corresponding active visual-KV budgets of $K = 144$, 288, and 864. CDPruner, VisionZip, ZOO-Prune, and MetaCompress follow their native compression procedures under the same target K/N budgets.

All methods use the same precision, attention backend, decoding configuration, warm-up procedure, and CUDA-synchronization timing protocol. We report TTFT, accumulated Prefill latency,

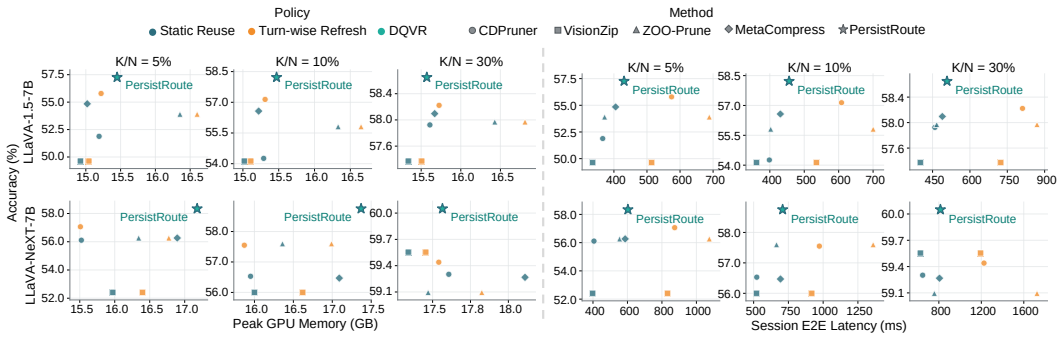

Figure 5: Extended performance–efficiency comparison on VQA-v2 Full-Session across LLaVA-1.5-7B and LLaVA-NeXT-7B. We report accuracy against peak GPU memory (left) and session end-to-end latency (right) at decoding-time visual-KV ratios $K/N \in \{5\%, 10\%, 30\%\}$. Static Reuse and Turn-wise Refresh denote the corresponding visual-update policies for the pruning baselines, while PersistRoute uses DQVR for turn-wise selective routing.

accumulated Decode latency, Session E2E latency, peak GPU memory, and Accuracy. TTFT is measured from the first-turn input to the first generated token; the remaining latency metrics cover the complete multi-turn session. Experiments are conducted on a server equipped with two NVIDIA GeForce RTX 4090 D GPUs and CUDA 12.4. LLaVA-1.5-7B is profiled on a single GPU, while LLaVA-NeXT-7B is profiled on two GPUs. Within each backbone, all compared methods, visual-update policies, and visual-KV budgets use the same GPU allocation.

B.3 Hyperparameter Selection Protocol

All default hyperparameters are selected using Full-Session data constructed from the VQA-v2 training split, which is disjoint from the VQA-v2 validation split used for the reported evaluation. Specifically, this development set is used to determine the SMPrune fusion weights $(w_S, w_R, w_F) = (0.5, 0.2, 0.3)$, the Session Sufficiency mixing weight $\alpha = 0.7$, the 6×6 regional partition, the global-neighbor count $k = 5$, the fusion temperature $\tau = 0.5$, and the default DQVR configuration of four layer groups. Once selected, these hyperparameters are fixed for all reported experiments without benchmark- or backbone-specific tuning.

C Additional Efficiency Results

Figure 5 extends the efficiency comparison across visual-KV budgets. Across both backbones, PersistRoute lies on the performance–efficiency Pareto frontier at every evaluated budget ($K/N \in \{5\%, 10\%, 30\%\}$) for both accuracy–latency and accuracy–memory trade-offs. Overall, PersistRoute exhibits a consistent Pareto-efficient profile across different visual-KV budgets.

D Additional Ablation Results

D.1 Controlled Routing Analysis

Protocol. We isolate the routing policy on LLaVA-1.5-7B using the 4,061-session MT-GQA benchmark. All variants first construct the same 50% SMPrune-constructed persistent visual memory ($N = 576$, $M = 288$), and each routing rule selects from this same persistent visual pool. We evaluate decoding-time visual-KV ratios $K/N \in \{5\%, 10\%, 30\%\}$, corresponding to $K \in \{29, 58, 173\}$. All variants share the same layer-group execution, physical gather, generation, and cache protocol; only the $M \to K$ routing rule changes.

Routing rules. Random samples an active subset from the persistent visual pool. Static-Q1 reuses the DQVR selection obtained from Turn 1 on later turns. Global Query-Aware Top-K computes one

current-query Top-K subset shared across layer groups, whereas DQVR computes a group-specific Top-K subset for each incoming query.

Table 6: Turn-wise controlled routing comparison on MT-GQA with LLaVA-1.5-7B. All variants share the same 50% persistent visual memory, decoding-time visual-KV ratio K/N, and routed execution protocol; only the routing rule differs. Entries report accuracy (%). Bold entries mark the best value within each ratio and turn, including ties.

Visual-KV Ratio K/N	Routing Rule	Turn 1	Turn 2	Turn 3	Avg.
5%	Random	57.82	58.83	61.05	59.23
	Static-Q1	**59.52**	59.09	61.06	59.89
	Global Query-Aware Top-K	59.21	60.42	62.74	60.79
	DQVR	**59.52**	**61.17**	**63.31**	**61.33**
10%	Random	58.90	60.69	62.02	60.54
	Static-Q1	**60.28**	59.19	61.53	60.33
	Global Query-Aware Top-K	59.64	61.54	63.61	61.60
	DQVR	**60.28**	**61.96**	**63.97**	**62.07**
30%	Random	59.39	60.20	62.89	60.83
	Static-Q1	**60.75**	60.25	62.76	61.25
	Global Query-Aware Top-K	60.39	62.15	64.18	62.24
	DQVR	**60.75**	**62.35**	**64.84**	**62.65**

Turn-wise interpretation. Static-Q1 and DQVR are identical at Turn 1 by construction. After the first turn, DQVR gains 2.08–2.77 points at Turn 2 and 2.08–2.44 points at Turn 3, showing that the advantage emerges as the query changes. DQVR also consistently exceeds the global query-aware control by 0.41–0.54 points in average accuracy, indicating that group-specific routing provides an additional benefit beyond current-query conditioning alone.

D.2 Effect of Layer-Group Granularity

We further examine the number of layer groups used by DQVR on MT-GQA with LLaVA-NeXT-7B. All variants use the same persistent-memory ratio $M/N = 50\%$ and active visual-KV ratio $K/N = 12.5\%$; only the number of DQVR layer groups is varied.

Table 7: Effect of DQVR layer-group granularity on MT-GQA with LLaVA-NeXT-7B. All settings use $M/N = 50\%$ and $K/N = 12.5\%$. DQVR overhead is reported per turn.

Layer Groups	Accuracy (%) ↑	DQVR Overhead (ms/turn) ↓
1	60.238	6.921
2	60.906	7.112
4	**62.136**	**8.849**
8	62.192	13.284

Increasing the number of layer groups from one to four improves accuracy from 60.238% to 62.136%, showing the benefit of adapting visual routing across decoder depth. Increasing the granularity further to eight groups yields only a marginal additional gain, reaching 62.192%, while DQVR overhead increases from 8.849 to 13.284 ms per turn. The four-group default selected using the development protocol in Appendix B.3 provides a favorable balance between depth-adaptive routing and routing overhead in this analysis.

D.3 Persistent-Memory Constructor Comparison

Table 8 compares four persistent-memory constructors under a common downstream execution protocol. All variants use $M/N = 50\%$, identical DQVR, the same cache lifecycle and generation protocol, and the same decoding-time visual-KV ratio K/N. Panel (a) jointly reports accuracy, session end-to-end latency, and peak GPU memory, while panel (b) reports direct-reduction accuracy for reference.

Table 8: Controlled comparison of persistent-memory constructors on VQA-v2 Full-Session with LLaVA-1.5-7B. (a) All variants use $M/N = 50\%$, identical DQVR, the same cache lifecycle and generation protocol, and the same decoding-time visual-KV ratio K/N. Accuracy, session end-to-end latency, and peak GPU memory are measured under the same execution setting. (b) Direct reduction is shown for reference. Bold and underlined values denote the best and second-best results within each ratio and metric, respectively.

(a) Persistent-Memory Construction with DQVR

K/N	Constructor	Acc. ↑	E2E (ms) ↓	Peak Mem. (GB) ↓
5%	VisionZip	56.13	453.504	**15.297**
	PruMerge	56.03	**428.160**	<u>15.434</u>
	ZOO-Prune	<u>56.57</u>	<u>430.507</u>	16.330
	SMPrune	**57.26**	430.733	15.447
12.5%	VisionZip	56.31	484.153	**15.333**
	PruMerge	56.30	**417.431**	<u>15.471</u>
	ZOO-Prune	<u>56.59</u>	<u>435.948</u>	16.330
	SMPrune	**58.43**	470.994	15.482
25%	VisionZip	<u>56.70</u>	<u>488.595</u>	**15.409**
	PruMerge	56.21	506.541	<u>15.548</u>
	ZOO-Prune	56.68	**479.871**	16.332
	SMPrune	**58.52**	497.628	15.560

(b) Direct Reduction Reference

Constructor	Direct Retention Ratio		
	5%	12.5%	25%
VisionZip	**54.22**	54.19	56.04
PruMerge	<u>54.06</u>	**55.94**	<u>57.11</u>
ZOO-Prune	53.51	<u>55.90</u>	**57.32**
SMPrune	49.36	55.03	57.10

Across the three decoding-time visual-KV ratios, SMPrune achieves the highest accuracy under the common protocol, reaching 57.26%, 58.43%, and 58.52% at 5%, 12.5%, and 25%, respectively. Session end-to-end latency varies across constructors and routing budgets, while peak memory remains close among VisionZip, PruMerge, and SMPrune. These results highlight persistent-memory construction as an important factor in task performance within a shared memory, routing, and execution configuration. Panel (b) reports the corresponding direct-reduction reference, illustrating how constructor behavior changes when its output is used directly as the active visual representation.

D.4 Factorized Ablation Efficiency Details

Static-Q1 selects the visual subset from the first query and reuses it across later turns, whereas DQVR selects a new subset from the persistent memory for every incoming query. The table below reports the complete factorized ablation record across the three decoding-time visual-KV ratios.

Across all three ratios, SMPrune substantially reduces session E2E latency and peak memory relative to the corresponding Full-memory variants. DQVR consistently improves accuracy over Static-Q1 under both memory settings, while introducing only a modest routing overhead in session latency. With SMPrune, the accuracy gains are 0.65, 0.91, and 0.23 points at 5%, 12.5%, and 25%, respectively, with only small increases in E2E latency. These results show that persistent-memory construction provides the main efficiency gain, while turn-wise selective routing further improves accuracy at limited additional cost.

D.5 Complete Cross-Backbone Results

Table 10 reports the complete cross-backbone results for LLaVA-1.5-13B, LLaVA-NeXT-7B, Qwen2.5-VL-7B, and Qwen3-VL-8B across MT-GQA, MT-VQA-v2, and ConvBench. Each entry lists performance at 5%, 12.5%, and 25% decoding-time visual-KV ratios, respectively. Base denotes the uncompressed model and is repeated across the three ratios for reference.

Across the four backbones, PersistRoute consistently delivers strong performance under reduced visual-KV budgets. At the 12.5% visual-KV ratio, it retains 96.63–99.17% of the Base performance

Table 9: Factorized ablation efficiency details on VQA-v2 Full-Session with LLaVA-NeXT-7B. Static-Q1 reuses the active subset selected from the first query, whereas DQVR refreshes the subset for each turn. Full denotes the 2880-token persistent memory; the 5%, 12.5%, and 25% settings activate 144, 360, and 720 visual KVs, respectively.

K/N	Persistent	Access	Acc. ↑	E2E (ms) ↓	Mem. (GB) ↓
5%	Full	Static-Q1	57.35	1808.13	24.998
	Full	DQVR	57.65	1838.71	24.996
	SMPrune	Static-Q1	57.69	**637.29**	**17.176**
	SMPrune	DQVR	**58.34**	652.03	**17.176**
12.5%	Full	Static-Q1	57.79	1867.82	25.110
	Full	DQVR	58.54	1896.63	25.112
	SMPrune	Static-Q1	58.25	**700.86**	**17.273**
	SMPrune	DQVR	**59.16**	716.89	17.274
25%	Full	Static-Q1	58.66	1909.85	25.307
	Full	DQVR	58.98	1932.72	25.307
	SMPrune	Static-Q1	59.67	**721.70**	**17.459**
	SMPrune	DQVR	**59.90**	738.09	17.460

Table 10: Complete cross-backbone results under 5%, 12.5%, and 25% decoding-time visual-KV ratios. Each entry reports performance in the order 5% / 12.5% / 25%. Base denotes the uncompressed model and is repeated across ratios for reference. Bold values indicate the best result among the compared compressed methods at each ratio.

Backbone	Method	MT-GQA	MT-VQA-v2	ConvBench
LLaVA-1.5-13B	Base	63.3090 / 63.3090 / 63.3090	80.1129 / 80.1129 / 80.1129	5.7192 / 5.7192 / 5.7192
	VisionZip	55.9140 / 59.5420 / 60.7732	68.9860 / 74.1610 / 77.5598	4.5061 / 4.6794 / 5.0260
	CDPruner	59.5256 / 61.0359 / 62.0947	73.3749 / 76.1708 / 78.1825	4.6794 / 4.8527 / 5.1993
	ZOO-Prune	59.5994 / 61.0769 / 61.9059	74.2666 / 76.7981 / 78.2564	4.6794 / 5.0260 / 5.1993
	PersistRoute	**62.1604 / 62.6611 / 62.7103**	**77.8569 / 78.9046 / 79.4231**	**5.0260 / 5.3726 / 5.5459**
LLaVA-NeXT-7B	Base	62.6529 / 62.6529 / 62.6529	84.7947 / 84.7947 / 84.7947	9.0121 / 9.0121 / 9.0121
	VisionZip	56.1356 / 58.8033 / 60.9866	75.0773 / 80.7440 / 83.2560	7.1057 / 7.4523 / 7.2790
	CDPruner	59.6159 / 61.3067 / 62.3656	79.6733 / 82.7307 / 84.1227	7.4523 / 7.2790 / 7.7990
	ZOO-Prune	59.5502 / 60.8963 / 62.2261	80.0667 / 82.7840 / 84.1253	6.9324 / 7.4523 / 7.7990
	PersistRoute	**61.1426 / 62.1358 / 62.5051**	**82.7280 / 84.1640 / 84.7373**	**7.7990 / 8.1456 / 8.4922**
Qwen2.5-VL-7B	Base	60.8553 / 60.8553 / 60.8553	89.3933 / 89.3933 / 89.3933	22.7036 / 22.7036 / 22.7036
	VisionZip	53.3366 / 55.2491 / 58.5324	73.0320 / 80.7013 / 85.5080	14.8469 / 17.0999 / **19.8151**
	CDPruner	48.7975 / 54.7402 / 57.4735	66.0800 / 74.7480 / 80.3333	11.4962 / 14.5003 / 16.5222
	ZOO-Prune	51.7278 / 57.7198 / 58.6919	73.8187 / 81.9547 / 85.9507	13.4027 / 16.5222 / 18.5442
	PersistRoute	**57.9578 / 58.8033 / 59.2547**	**85.2773 / 86.9147 / 87.5920**	**16.4645 / 19.2952** / 19.6996
Qwen3-VL-8B	Base	62.9073 / 62.9073 / 62.9073	88.9200 / 88.9200 / 88.9200	23.4683 / 23.4683 / 23.4683
	VisionZip	55.2343 / 57.2110 / 60.6061	72.7453 / 80.3740 / 85.1552	15.4669 / 17.7258 / 20.2825
	CDPruner	50.3929 / 56.6160 / 59.4114	65.5301 / 74.4422 / 79.8379	11.8734 / 14.9886 / 17.1786
	ZOO-Prune	53.5120 / 59.7760 / 60.6709	73.4678 / 81.4907 / 85.5156	13.8741 / 17.1786 / 19.1888
	PersistRoute	**60.8307 / 61.1918 / 61.2493**	**85.9973 / 86.7667 / 86.9893**	**18.1295 / 20.9451 / 21.5040**

on MT-GQA and 97.23–99.26% on MT-VQA-v2. Across backbones and visual-KV ratios, PersistRoute maintains consistently high compressed performance, with particularly stable results on MT-GQA and MT-VQA-v2. These results further support the generalization of persistent-memory construction and turn-wise selective routing across different VLM architectures.

D.6 Complete Persistent-Memory and DQVR-Ratio Sweep

Figure 6 extends the persistent-memory and DQVR-ratio sweep with a zero-routing control, $K = 0$ ($K/M = 0\%$). The ratio K/M is measured relative to the persistent memory, rather than the original visual-token sequence. The nonzero-routing columns correspond to the 3D visualization in Fig. 4: over $K/M = 5\%$–75%, accuracy is more sensitive to persistent-memory capacity M/N than to the routing ratio.

The $K = 0$ column provides a zero-routing boundary condition in which no persistent visual KVs are activated for current-turn layer-group attention. Across the evaluated persistent-memory ratios, increasing K/M from 0% to 5% improves accuracy by 5.92–10.71 percentage points. In comparison, further increasing the routing ratio from 5% to 75% produces a total accuracy range of 0.57–2.05 points at fixed M/N. This contrast shows that direct routed visual access provides a substantial benefit, while additional routed capacity beyond the initial 5% produces comparatively smaller changes over the evaluated nonzero routing range.

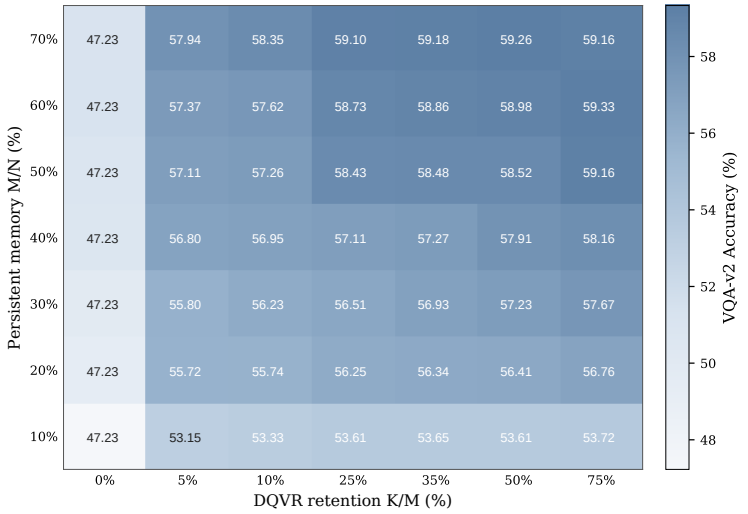

Figure 6: Complete persistent-memory and DQVR-ratio sweep on VQA-v2 Full-Session with LLaVA-1.5-7B. Each cell reports accuracy for a persistent-memory ratio M/N and a DQVR routing ratio K/M measured relative to the persistent memory. The nonzero-routing columns correspond to Fig. 4; the additional $K = 0$ column is a zero-routing control with no persistent visual KVs activated for current-turn layer-group attention.

D.7 SMPrune Criterion Ablation

Evaluation protocol. We evaluate the SMPrune criterion composition on LLaVA-1.5-7B using the same full-session GQA and VQA-v2 setting as the persistent-memory quality analysis. The original visual sequence contains $N = 576$ tokens, and every variant constructs a fixed $M = 288$ persistent visual memory, corresponding to a 50% retention ratio. DQVR is disabled so that only the persistent memory construction criteria are compared. The evaluation includes 398 GQA sessions with 12,578 questions and 822 VQA-v2 sessions with 4,354 questions. The fusion temperature is fixed at $\tau = 0.5$.

Let S, R, and F denote Session Sufficiency, Evidence Removability, and Future-Query Resilience, respectively. The full formulation uses weights $(w_S, w_R, w_F) = (0.5, 0.2, 0.3)$. The S-only variant uses $(1, 0, 0)$, while the leave-one-out variants use $(0, 0.4, 0.6)$ for w/o S, $(0.625, 0, 0.375)$ for w/o R, and $(0.7142857143, 0.2857142857, 0)$ for w/o F; the remaining weights are proportionally renormalized. We reuse the persistent-memory quality diagnostic definitions in Appendix E.1.

Table 11: Full SMPrune criterion ablation on LLaVA-1.5-7B under a fixed persistent-memory ratio $M/N = 50\%$ with DQVR disabled. Values are percentages except Future Drop, which is reported in percentage points. Utility denotes Utility Capture; higher is better except for Future Drop.

Variant	GQA Acc. ↑	VQA-v2 Acc. ↑	Macro Acc. ↑	WTR ↑	Utility Capture ↑	CSR ↑	NCRP ↑	Future Drop ↓
S only	59.52	58.18	58.85	92.71	56.78	63.19	95.29	0.73
w/o S	59.82	57.88	58.85	91.21	57.42	64.36	**95.44**	0.85
w/o R	59.59	58.45	59.02	91.41	57.59	64.65	95.38	0.49
w/o F	59.59	58.29	58.94	91.21	57.01	63.75	95.31	0.68
Full	**60.16**	**58.82**	**59.49**	**92.74**	**57.75**	**65.04**	95.40	**0.48**

Criterion-level results. Table 11 reports the complete criterion ablation using the existing persistent-memory diagnostics. The full formulation achieves the best GQA accuracy, VQA-v2 accuracy, Macro Accuracy, WTR, Utility Capture, CSR, and Future Drop. Removing S causes the largest Macro Accuracy loss (0.64 percentage points), while removing F causes the largest leave-one-out losses in Utility Capture and CSR. Across the eight reported metrics, the full formulation leads on seven, while the w/o S variant obtains a marginally higher NCRP (95.44 vs. 95.40). Together, these results indicate complementary contributions from the three criteria.

Signal complementarity. The criterion scores are correlated but not redundant, with average image-level Spearman correlations of 0.433 (S–R), 0.555 (S–F), and 0.703 (R–F). Removing S, R, or F yields average selected-token Jaccards of 0.646, 0.903, and 0.841 with the full formulation, respectively. This overlap pattern further indicates that the three criteria provide related but distinct signals for constructing persistent visual memory.

D.8 Effect of SMPrune Hyperparameters

We examine the main structural hyperparameters of SMPrune on a fixed 200-session development subset constructed from the VQA-v2 training split under the Full-Session setting. The subset contains 1,026 questions and is fixed across all configurations. All variants use the same persistent-memory ratio $M/N = 50\%$ with DQVR disabled; only one hyperparameter is varied at a time.

Table 12: Effect of SMPrune hyperparameters on a fixed 200-session development subset constructed from the VQA-v2 training split under the Full-Session setting. All settings use $M/N = 50\%$ with DQVR disabled. Bold values denote the selected default configuration.

Hyperparameter	Setting	Accuracy (%) ↑
Region grid	4×4	56.43
	6×6	**56.92**
	8×8	55.95
Global neighbor k	3	56.53
	5	**56.92**
	10	56.63
Fusion temperature τ	0.25	56.73
	0.5	**56.92**
	1.0	56.53

Varying the global-neighbor count from $k = 3$ to 10 changes accuracy by at most 0.39 percentage points relative to the default $k = 5$, while varying the fusion temperature from $\tau = 0.25$ to 1.0 changes accuracy by at most 0.39 points relative to $\tau = 0.5$. The spatial partition has a larger effect: using 4×4 and 8×8 region grids reduces accuracy by 0.49 and 0.97 points relative to the default 6×6 grid. These results show that SMPrune remains stable under moderate changes to the global-neighbor count and fusion temperature, while the spatial granularity has a comparatively larger influence on persistent-memory construction.

E Persistent-Memory Quality Analysis

E.1 Metric Definitions

The following metrics are offline diagnostics for analyzing persistent-memory quality. Several use future conversation turns or ground-truth answers to estimate the utility of removed visual evidence and are computed only for post-hoc evaluation, outside SMPrune and DQVR inference.

Let s index a multi-turn session, t index a turn, and i index a visual token. For a fixed method, let $q_{s,i}$ denote the token-retention value, with $q_{s,i} = 1$ for a retained token and $q_{s,i} = 0$ for a removed token. Let $u_{s,t,i} \geq 0$ denote the non-negative offline oracle utility of visual token i at turn t. We use $\mathrm{Acc}^{m}_{s,t}$ and $\mathrm{Acc}^{\mathrm{full}}_{s,t}$ for the method and Full Cache accuracy, respectively. Let T_s be the set of turns in session s and $\mathcal{S}$ the evaluated session set.

The oracle utility is obtained offline. For each non-overlapping 2×2 visual-token group, we delete the group and compute

$$\Delta\mathrm{NLL} = \mathrm{NLL}_{\mathrm{deleted}} - \mathrm{NLL}_{\mathrm{full}}, \tag{21}$$

using the ground-truth answer. The resulting non-negative group utility is mapped to the corresponding source tokens to obtain $u_{s,t,i}$. For LLaVA-1.5, the 24×24 visual-token grid yields 144 non-overlapping 2×2 groups. We define the critical-evidence mask by

$$C_{s,t,i} = \mathbf{1}[u_{s,t,i} > 0.05]. \tag{22}$$

Worst-Turn Retention (WTR). For each session turn, we define

$$r_{s,t} = \begin{cases} \mathrm{Acc}^{m}_{s,t}/\mathrm{Acc}^{\mathrm{full}}_{s,t}, & \mathrm{Acc}^{\mathrm{full}}_{s,t} > 0, \\ \mathbf{1}[\mathrm{Acc}^{m}_{s,t} = \mathrm{Acc}^{\mathrm{full}}_{s,t}], & \text{otherwise.} \end{cases} \tag{23}$$

The reported WTR is aggregated as

$$\mathrm{WTR} = \min_{b} \left(\frac{1}{|Q_b|} \sum_{(s,t) \in Q_b} r_{s,t} \right), \tag{24}$$

where b ranges over the exact Turn 1–9 buckets and the session-balanced Turn 10+ bucket, and Q_b is the corresponding set of session–turn pairs. Full Cache is the reference; higher values are better.

Utility Capture. For each session turn, Utility Capture is

$$\mathrm{UC}_{s,t} = \frac{\sum_i u_{s,t,i} q_{s,i}}{\sum_i u_{s,t,i} + \epsilon}, \qquad \epsilon = 10^{-12}. \tag{25}$$

The final metric is

$$\mathrm{UC} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{1}{|T_s|} \sum_{t \in T_s} \mathrm{UC}_{s,t}. \tag{26}$$

The numerator is the retained offline oracle utility and the denominator is the total oracle utility. Higher values are better, with utility derived from the offline group-deletion $\Delta\mathrm{NLL}$ oracle.

Critical Survival Retention (CSR). Using the critical mask in Eq. 22, we define

$$\mathrm{CSR}_{s,t} = \frac{\sum_i C_{s,t,i} q_{s,i}}{\sum_i C_{s,t,i}}. \tag{27}$$

The final metric is

$$\mathrm{CSR} = \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \frac{1}{|T_s|} \sum_{t \in T_s} \mathrm{CSR}_{s,t}. \tag{28}$$

Turns with no critical token are skipped using the implementation's NaN handling. Higher values are better, with critical evidence defined by the offline $\Delta\mathrm{NLL}$ oracle.

Non-Critical Removal Precision (NCRP). Let the deleted retention mass be

$$d_{s,i} = 1 - q_{s,i}. \tag{29}$$

The mass-based Non-Critical Removal Precision (NCRP) diagnostic is

$$\mathrm{NCRP}_{s,t} = \frac{\sum_i (1 - C_{s,t,i}) d_{s,i}}{\sum_i d_{s,i}}. \tag{30}$$

Its numerator is the deleted non-critical retention mass and its denominator is the total deleted retention mass. NCRP is therefore computed over retention mass rather than discrete token counts. The reported value is averaged from turns to sessions and then by dataset macro-average; higher values are better.

Restoration AUC. For the restoration analysis, the restored percentages are

$$p \in \{0, 10, 20, 40, 60, 80, 100\}.$$

Starting from the 50% pruned persistent representation, deleted visual tokens are restored in descending order of the offline worst-turn oracle utility. For method m, let

$$g_m(p) = \mathrm{mean}_s \left(\mathrm{Acc}^m_{s,p} - \mathrm{Acc}^m_{s,0} \right). \tag{31}$$

The Restoration AUC is computed by trapezoidal integration:

$$\mathrm{RAUC}_m = \sum_{j=1}^{6} \frac{p_j - p_{j-1}}{100} \frac{g_m(p_{j-1}) + g_m(p_j)}{2}. \tag{32}$$

The analysis uses a fixed set of 245 sessions. The horizontal axis is the restored percentage of deleted visual tokens, and the vertical axis is the accuracy gain over the base pruned memory, in percentage points. The factor $1/100$ normalizes the restored percentage to $[0, 1]$, so RAUC is also reported in percentage points. Lower RAUC indicates less recoverable evidence among the deleted tokens under this protocol.

Future Drop. For sessions with at least two turns, we define the later-turn accuracy drop as

$$\mathrm{FD}_s = \frac{1}{|T_s| - 1} \sum_{t=2}^{|T_s|} \left(\mathrm{Acc}^{\mathrm{full}}_{s,t} - \mathrm{Acc}^m_{s,t} \right). \tag{33}$$

The final metric is

$$\mathrm{FD} = \frac{1}{|\mathcal{S}_{\geq 2}|} \sum_{s:\, |T_s| \geq 2} \mathrm{FD}_s. \tag{34}$$

Here, "future" denotes Turns 2 and later. Full Cache is the reference and lower values are better.

Offline persistent-memory quality diagnostics

Metric	Reference / Oracle	Direction
WTR	Full Cache	↑
Utility Capture	offline ΔNLL utility	↑
CSR	offline critical-evidence oracle	↑
NCRP	offline critical-evidence oracle	↑
Restoration AUC	offline restoration oracle	↓
Future Drop	Full Cache	↓

E.2 Evaluation Protocol and Results

Evaluation protocol. We analyze SMPrune with the persistent-memory quality diagnostics defined above. All metrics are computed with the same persistent-memory ratio and generation and evaluation protocol.

For this analysis, we use LLaVA-1.5-7B with $576 \rightarrow 288$ visual tokens under the same persistent-memory ratio $M/N = 50\%$ for all methods, disable DQVR, and evaluate 1,220 shared-image multi-turn sessions with 16,932 questions, including 398 GQA sessions with 12,578 questions and 822 VQA-v2 sessions with 4,354 questions. Query-conditioned baselines construct a fixed persistent memory from the first-turn query and reuse it in later turns.

Overall quality profile. Table 13 compares persistent pruning strategies under the same $M/N = 50\%$ persistent-memory ratio. SMPrune achieves the highest WTR, Utility Capture, and CSR, with a Restoration AUC of 0.24, compared with 1.09–2.74 for the evaluated baselines. At the same time, CDPruner obtains the highest Macro Accuracy and the lowest Future Drop, while VisionZip achieves the highest NCRP. These results show that aggregate task accuracy alone does not fully characterize the quality of a persistent visual memory. SMPrune performs strongly across session stability, retained evidence, and removability metrics.

Session Sufficiency. SMPrune achieves the highest WTR of 92.74, slightly exceeding CDPruner at 92.62. This indicates that the retained memory remains more stable in the weakest evaluated turn bucket.

Table 13: Persistent-memory quality analysis on LLaVA-1.5-7B with the same 50% persistent-memory ratio. All methods construct a 288-token persistent visual memory from the original 576 visual tokens. DQVR is disabled to isolate the quality of persistent visual-memory construction. Restoration AUC uses the normalized integration in Eq. 32.

Method	Macro Acc. ↑	WTR ↑	Utility Capture ↑	CSR ↑	NCRP ↑	Restoration AUC ↓	Future Drop ↓
Random	58.67	89.82	49.96	50.19	94.72	2.74	1.38
FastV	57.51	86.29	50.07	51.24	95.14	2.15	2.95
CDPruner	**59.80**	<u>92.62</u>	55.13	58.86	95.30	1.37	**0.37**
PruMerge	59.57	92.11	53.98	56.43	95.20	<u>1.09</u>	0.75
VisionZip	59.18	91.13	50.69	53.51	**95.52**	1.11	0.71
ZOO-Prune	<u>59.77</u>	91.60	<u>56.15</u>	<u>61.81</u>	95.35	1.42	<u>0.42</u>
SMPrune	59.49	**92.74**	**57.75**	**65.04**	<u>95.40</u>	**0.24**	0.48

Evidence Removability. SMPrune achieves the highest Utility Capture and CSR, reaching 57.75 and 65.04, respectively. VisionZip obtains the highest NCRP at 95.52, while SMPrune remains close at 95.40. Together, these results indicate that SMPrune preserves more useful and critical visual evidence under the same persistent-memory ratio.

Future-Query Resilience. SMPrune achieves the lowest Restoration AUC of 0.24, compared with 1.09 for PruMerge, the strongest baseline on this metric. This agrees with the lower restoration-gain curve in Fig. 1(c), indicating less recoverable evidence among SMPrune's removed tokens under the same restoration protocol. CDPruner achieves the lowest Future Drop at 0.37, showing that the two measurements capture complementary aspects of future-turn robustness.

F VISUALIZATION

We further visualize the behavior of PersistRoute with LLaVA-1.5-7B in Fig. 7. For all examples, SMPrune first constructs a fixed persistent visual memory with $M/N = 50\%$ ($N = 576$, $M = 288$), after which DQVR routes $K = 15$ visual tokens for each incoming query. The persistent memory remains unchanged across turns, while the routed visual evidence varies with the current query.

G LIMITATIONS

Our study has several limitations. First, Future-Query Resilience is a query-independent proxy for preserving distinctive visual evidence rather than a formal guarantee that all information required by unseen future queries will be retained. Persistent pruning may therefore still remove evidence that becomes important later in a conversation. Second, our evaluation mainly focuses on shared-image multi-turn VQA settings. Broader validation on longer conversations, more diverse multimodal tasks, and additional VLM families would further establish the generality of the framework. Finally, DQVR reduces the number of active visual KVs used by each layer group, and its practical benefit may depend on factors such as visual-sequence length, query length, and response length.



Figure 7: Qualitative visualization of PersistRoute with LLaVA-1.5-7B. SMPrune first constructs a fixed persistent visual memory with $M/N = 50\%$ ($M = 288$ from $N = 576$), after which DQVR routes $K = 15$ visual tokens for each incoming query. Colors indicate routing frequency, pruned regions denote visual tokens removed during persistent-memory construction, and red boxes highlight the visual regions relevant to each query. Across conversation turns, PersistRoute reuses the same persistent memory while adapting visual access according to the current query.