GraphMemix: Query-Aware Evidence Forests for Long-Term Multimodal Agent Memory

Geng Li[1], Yuhao Wang[1], Dong Li[2], Jianye Hao[2], Yuxin Peng[1]

[1]Wangxuan Institute of Computer Technology, Peking University
[2]MemoraX AI

Correspondence to: Yuxin Peng <pengyuxin@pku.edu.cn>

Abstract. Organizing long-term memory for multimodal agents remains challenging because existing methods either suffer from expensive question-agnostic offline summaries or naive embedding similarity matching that introduces incomplete and redundant context. To address these issues, we propose GraphMemix, a **combinatorial-optimization graph memory framework** that **models** memory organization as query-aware evidence-forest construction. Specifically, our method consists of three key components: (1) **candidate graph construction**, which expands multi-view seed memories through schema and semantic relations to acquire query-aware original context; (2) **evidence utility and activation costs**, which decouples direct memory support from anchor-conditioned relation verification to suppress redundant or conflicting information; and (3) **forest optimization**, which jointly selects a forest-format memory context under a maximum evidence budget and its reliable relational structure. By organizing memory into a query-relevant subgraph, the method avoids substantial lifecycle cost and recovers low-similarity complementary evidence. Experimental results across four long-term multimodal memory benchmarks demonstrate significant improvements with different foundation models and establish a new Pareto frontier between accuracy and lifecycle cost.

Keywords multimodal agent memory, evidence retrieval, graph optimization, long-term memory

1 Introduction

Memory serves as a fundamental component for Large Vision-Language Model (LVLM)-based agents J. Feng et al., 2026; Y. Wang and X. Chen, 2025 to maintain consistency across long-horizon multimodal interactions and incorporate new information Bei et al., 2026; Ren et al., 2026. Especially, for personal assistant agents J. Feng et al., 2026; Y. Wang and X. Chen, 2025, the memory system must encompass not only historical user-agent interactions Bei et al., 2026; J. Feng et al., 2026, but also vast volumes of continuously updated, user-centric multimodal data, such as photo albums, emails, personal chat logs, work documents and etc. Mei et al., 2026; Y. Wang and X. Chen, 2025. This scale and heterogeneity make it difficult for agent memory to retrieve precise, query-relevant context Du et al., 2026; Jiang et al., 2025; C. Li et al., 2026.

Existing methods mainly follow two routes. The first extends text-based agent memory by converting history into reusable summaries, typed fields, or linked notes before the future question is

known. MIRIX Y. Wang and X. Chen, 2025 partitions experience into episodic, semantic, procedural and knowledge-vault memories. SGM Mei et al., 2026 maps source-specific image, video, and email content into a shared record with meta fields. A-MEM W. Xu et al., 2026 constructs structured notes and updates their links; AUGUSTUS Jain et al., 2025 and M^{2}A J. Feng et al., 2026 further connect semantic memories to original records. These *question-agnostic* memory methods compress history to build a general-purpose representation before queries arrive. However, because no future question is considered, the whole user history are processed by default to avoid omission, creating a large cold-start and continuing update cost. More fundamentally, question-agnostic compression may omit visual attribute, state transition, or local context that later becomes decisive Guo et al., 2026; Ren et al., 2026. The second route is *multimodal RAG*. MuRAG W. Chen et al., 2022 retrieves directly from a corpus of images and text, while Pensieve Jiang et al., 2025 combines images, captions, OCR, spatio-temporal cues, and multimodal similarity to search personal visual experience. These methods preserve native visual access and cost low in processing. However, ranking records primarily by similarity matching leads them to overselect near-duplicates while omitting dissimilar replies or state updates whose value emerges only in combination with other records. Prior work shows that answering a question can require multiple complementary memories, while semantic similarity alone does not determine their joint evidential utility Du et al., 2026; Jiang et al., 2025.

To address these limitations, we propose GraphMemix, a combinatorial-optimization graph memory framework that reconstructs necessary evidence forest from large-scale memory archives after the question arrives. Fundamentally, GraphMemix formulates memory selection as query-conditioned evidence-forest optimization: nodes capture the direct utility of individual memories, while edges capture their incremental value when conditioned on an anchor. To instantiate this objective, GraphMemix starts from multi-view matching anchors and expands through schema and semantic edges to reconstruct a bounded, query-relevant candidate subgraph. A node verifier estimates direct support, while an Evidence-Chain Verifier estimates whether each anchor-conditioned relation contributes complementary information rather than redundancy or conflict. The resulting objective jointly selects relevant memories and reliable relations, recovering low-similarity context while suppressing uncertain expansion. For any fixed node set, its optimal maximum-weight forest is obtained exactly by Kruskal's algorithm. Experiments on four long-term multimodal memory benchmarks and two reader families support this design. With Qwen3-VL, GraphMemix improves Judge Accuracy over the strongest public baseline on every benchmark and raises the four-dataset macro-average by 11.75 percentage points, from 49.80% to 61.55%. In terms of efficiency, it achieves a lifecycle-cost Pareto frontier compared with existing approaches, demonstrating that query-local semantic organization boosts answer quality without incurring history-wide generative preprocessing overhead.

2 Related Work

2.1 Long-Term Multimodal Agent Memory

Long-term memory enables agents to preserve and reuse interaction history beyond a finite context. Generative Agents Park et al., 2023 write observations into a memory stream and synthesize reflections; MemGPT Packer et al., 2023 controls movement across hierarchical storage; A-MEM W. Xu et al., 2026 turns new experiences into structured notes and updates links to existing memory. These works establish persistent writing, organization, and controlled retrieval, but primarily operate on textual experiences or textualized memory units. Multimodal memory systems make text and images jointly searchable. MuRAG W. Chen et al., 2022 uses non-parametric multimodal memory for knowledge-intensive QA; Pensieve Jiang et al., 2025 combines captions, OCR, spatiotemporal metadata, and multimodal similarity for personal visual experience; MIRIX Y. Wang

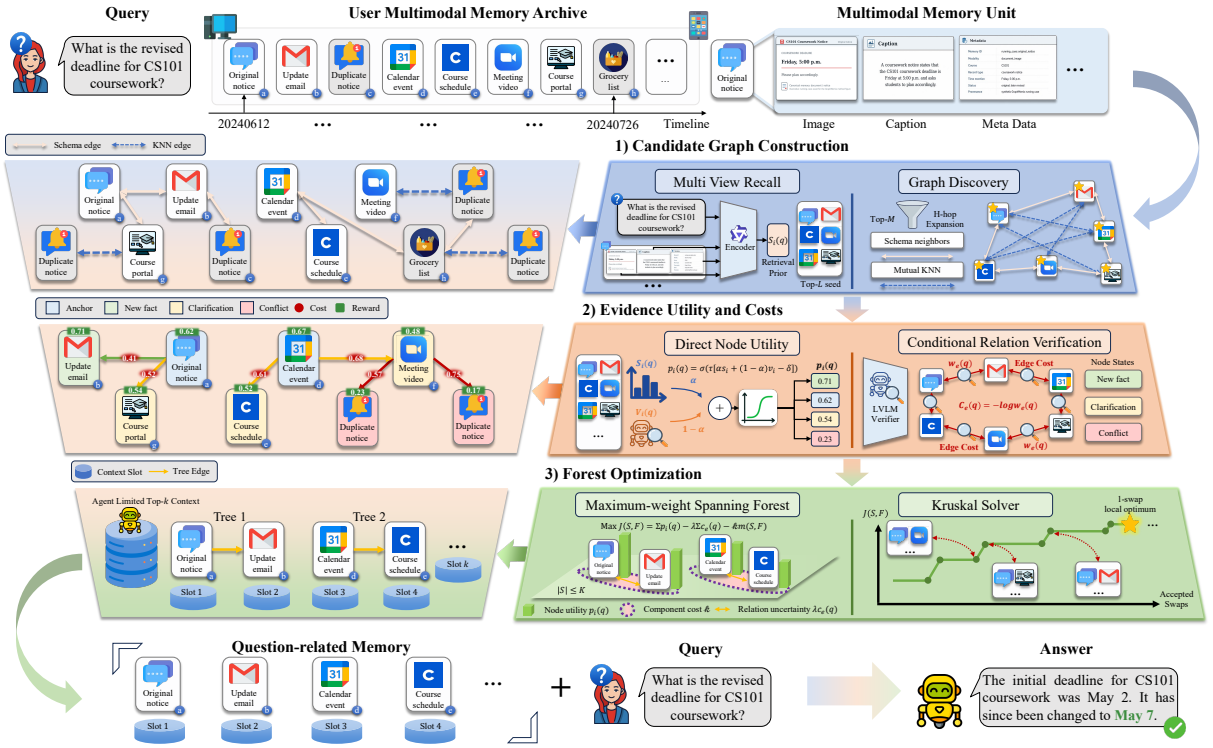

Figure 1: Overview of GraphMemix in three stages. (1) *Candidate graph construction* uses multi-view retrieval to identify seeds and bounded relation expansion to expose their local context. (2) *Evidence utility and costs* combines direct node support with ECV-validated, anchor-conditioned incremental relations. (3) *Forest optimization* jointly selects nodes and trusted edges under a maximum reader budget, then deterministically serializes the resulting evidence forest.

and X. Chen, 2025 and SGM Mei et al., 2026 organize history as typed memories or a unified schema. AUGUSTUS Jain et al., 2025 and M^{2}A J. Feng et al., 2026 connect semantic indices or editable memories to logs that retain the original media. Such approaches either demand lengthy preprocessing or lose structural context of stored memories. GraphMemix focuses on reconstructing the evidence subgraph conditioned on each query, thereby reducing lifecycle computational overhead and improving the completeness of memory context.

2.2 Query-Conditioned Evidence Retrieval

Query-conditioned retrieval adapts memory access to the information need expressed by the current question. MemGuide Du et al., 2026 retrieves intent-aligned memories and then filters them according to missing information, while MemReranker C. Li et al., 2026 targets temporal constraints, causal relations, and multi-turn coreference that challenge generic relevance models. In general retrieval, RankGPT Sun et al., 2023 shows that listwise comparison with a generative LLM can outperform independent similarity scoring. These methods primarily produce query–memory relevance scores or rankings; candidate–candidate relations are not explicit decision variables. Multimodal RAG further asks whether retrieved content genuinely supports generation. RagVL Z. Chen et al., 2024 uses an LVLM to rerank retrieved images, MEG-RAG X. Wang et al., 2026 trains a multimodal evidence reranker around the semantic core of an answer. These works generally ignore inter-evidence correlations and are vulnerable to redundant or contradictory evidence. GraphMemix expands query-conditioned scoring to complete context construction, jointly modeling evidence utility and valid inter-evidence links while discarding redundant similar memories.

3 Method

We propose GraphMemix, which formulates long-term multimodal memory organization as a query-conditioned forest optimization problem over a bounded candidate graph.

3.1 Problem Formulation

Let $\mathcal{V} = \{e_i\}_{i=1}^{N}$ be user multimodal memory archive containing N historical records, and let $q = (q^{\text{text}}, q^{\text{vis}})$ be a question with visual input available. Each e_i may contain text, images, video, and derived representations. An evidence selector Π chooses an evidence set of at most K records and organizes it into an ordered sequence O_q. A frozen multimodal generator $\mathcal{M}$ then produces the answer:

$$\begin{aligned} (S_q, O_q) &= \Pi(q, \mathcal{V}; K), \qquad S_q \subseteq \mathcal{V}, \quad |S_q| \le K, \\ \hat{y} &= \mathcal{M}(q, O_q). \end{aligned} \tag{1}$$

Our objective is to design Π such that the selected evidence provide complete context to improve the correctness of $\hat{y}$.

3.2 Query-Conditioned Evidence Forest

A direct implementation of Π is ranking memories independently by query-memory similarity. It can waste slots on duplicates and miss a response, subsequent state, or referential context whose wording differs from the question. Conversely, inserting every connected record introduces irrelevant context. We instead represent candidate memories as nodes and possible contextual dependencies as edges. For selected nodes S and an acyclic edge set F, we write the selection objective as

$$\max_{S,F} \; \mathcal{U}_{\text{node}}(S \mid q) - C_{\text{edge}}(F \mid q) - C_{\text{open}}(S, F), \tag{2}$$

where the terms measure direct evidence utility U_{node}, uncertainty in expanding a relation C_{edge}, and the cost of opening independent evidence chains C_{open}. The remainder of this section constructs and solves this objective.

3.3 Candidate Graph Construction

Multi-view retrieval. The same memory may be retrieved through different signals: an original image for objects and scenes, a caption for events and actions, OCR for printed text, or a video frame for visual state. Let $\mathcal{U}_i$ be the retrieval views of e_i. Its initial score is

$$s_i(q) = \max_{u \in \mathcal{U}_i} \text{sim}(h(q), h(u)), \tag{3}$$

where h is a shared multimodal encoder. Max pooling only provides multiple routes into the candidate set.

Query-relevant subgraph discovery. We use two complementary relation sources. Schema relations encode observable interaction structure, such as records from the same session or location. Semantic relations connect memories with mutually similar representations. Let $\mathcal{T}_{\text{schema}}$ be the schema relation types, $\phi_r(e_i, e_j)$ indicate whether relation r holds, and $\mathcal{N}_k(i)$ denote the k nearest neighbors of i. We define

$$\begin{aligned} E_{\text{schema}} &= \{(i,j) : \exists r \in \mathcal{T}_{\text{schema}},\ \phi_r(e_i, e_j) = 1\}, \\ E_{\text{sem}} &= \{(i,j) : i \in \mathcal{N}_k(j),\ j \in \mathcal{N}_k(i)\}, \end{aligned} \tag{4}$$

and the discovery graph $G^{\text{disc}} = (\mathcal{V}, E_{\text{schema}} \cup E_{\text{sem}})$. Starting from the top-$L$ multi-view memories $R_q^{(L)} = \text{Top}_L(\mathcal{V}; s_i(q))$, we collect nodes reachable within H hops and retain at most M candidates:

$$\widetilde{C}_q = R_q^{(L)} \cup N_{\le H}^{G^{\text{disc}}}(R_q^{(L)}), \qquad C_q = \text{Top}_M(\widetilde{C}_q). \tag{5}$$

The ranking combines retrieval order and relation-expansion order. This step preserves local context while bounding all subsequent semantic reasoning.

3.4 Evidence Utility and Activation Costs

Direct node utility. Retrieval similarity is a stable global prior, but is insufficient for negation, temporal change, visual reference, or memories that share a topic but imply different answers. A listwise node verifier reads q and C_q in one call and outputs $v_i(q)$, the degree to which e_i independently supports the question. Listwise input places candidates on a common semantic scale. We fuse the normalized verifier score $\widetilde{v}_i$ with retrieval:

$$\begin{aligned} p_i(q) &= \sigma(\tau[\alpha s_i(q) + (1-\alpha)\widetilde{v}_i(q) - \delta]), \\ \mathcal{U}_{\text{node}}(S \mid q) &= \sum_{i \in S} p_i(q), \end{aligned} \tag{6}$$

where α balances the retrieval prior and conditional judgment, while τ and δ control scale and offset.

Anchor-conditioned relation verification. Node verification asks whether a memory is useful alone, but cannot distinguish redundancy from complementarity with an already retrieved memory. We take a high-confidence anchor set A_q from the initial retrieval and inspect only schema edges between anchors and candidates:

$$E_q^{\text{elig}} = \{(a, i) \in E_{\text{schema}} : a \in A_q,\ i \in C_q\}. \tag{7}$$

The Evidence-Chain Verifier (ECV) reads these pairs in one listwise call. For each candidate, it chooses the anchor that best explains the candidate's incremental role, predicts an incremental-support score $s_{ai}^{\text{inc}}(q)$, and assigns one of six roles: `new_fact`, `clarification`, `corroboration`, `redundant`, `conflict`, or `irrelevant`. Only the best anchor edge with a positive score and one of the first three roles is retained, forming

$$G_q^{\text{trust}} = (C_q, E_q^{\text{trust}}), \qquad E_q^{\text{trust}} \subseteq E_q^{\text{elig}}. \tag{8}$$

For a retained edge (a, i), its reliability and uncertainty cost are

$$w_{ai}(q) = w_{ai}^{\text{schema}} \frac{s_{ai}^{\text{inc}}(q)}{s_{\max}}, \qquad c_{ai}(q) = -\log w_{ai}(q), \tag{9}$$

where w_{ai}^{schema} is a schema-specific reliability ceiling. Accordingly, $C_{\text{edge}}(F \mid q) = \lambda \sum_{e \in F} c_e(q)$. Role gating determines whether an edge carries positive incremental semantics, the continuous score then controls its credit. A related but redundant or conflicting memory therefore receives no structural reward.

Independent-chain cost. Let $m(S, F)$ be the number of connected components in the forest (S, F). Every component is an independently interpreted evidence chain, so we define

$$C_{\text{open}}(S, F) = \kappa\, m(S, F), \tag{10}$$

where $\kappa \ge 0$ balances independent evidence against contextual completion of an existing event.

		ATM		Mem-Gallery		MemEye		H2HMem		Avg.
Method	Venue	J.Acc.	EM	J.Acc.	EM	J.Acc.	M.EM	J.Acc.	Recall	J.Acc.
A-MEM	NeurIPS'25	41.86	40.80	52.89	32.61	37.09	40.57	39.94	40.55	42.94
UniversalRAG	ACL'26	43.10	41.19	63.76	32.50	47.98	49.33	44.36	45.04	49.80
MemGuide	AAAI'26	48.47	47.80	48.10	32.73	41.40	44.61	39.94	35.81	44.48
LightMem	ICLR'26	22.70	24.52	38.28	23.20	36.23	39.76	31.04	27.82	32.06
VimRAG	ICML'26	33.81	32.76	48.33	16.13	28.25	28.23	17.89	17.01	32.07
GraphMemix	Ours	**55.27**	**53.83**	**76.33**	**36.76**	**53.64**	**54.25**	**60.96**	**54.70**	**61.55**
Gain over 2nd	–	+6.80	+6.03	+12.57	+4.03	+5.66	+4.92	+16.60	+9.66	+11.75

Table 1: End-to-end results with Qwen3-VL-8B-Instruct, judged by GPT-5-mini. J.Acc. denotes Judge Accuracy (%). Each benchmark additionally reports its representative native metric. Avg. is the macro-average over the four Judge Accuracies.

		ATM		Mem-Gallery		MemEye		H2HMem		Avg.
Method	Venue	J.Acc.	EM	J.Acc.	EM	J.Acc.	M.EM	J.Acc.	Recall	J.Acc.
A-MEM	NeurIPS'25	42.43	41.95	50.03	28.40	24.42	30.93	44.20	39.97	40.27
UniversalRAG	ACL'26	49.04	48.18	64.82	32.14	58.17	38.07	48.34	41.31	55.09
MemGuide	AAAI'26	54.98	53.07	48.33	22.85	30.40	36.66	47.98	41.97	45.43
LightMem	ICLR'26	24.33	25.48	33.72	17.01	20.70	30.59	32.80	27.79	27.89
VimRAG	ICML'26	22.51	21.36	46.99	14.14	31.81	34.64	39.00	33.67	35.08
GraphMemix	Ours	**58.05**	**57.47**	**81.47**	**35.65**	**66.68**	**43.06**	**63.47**	**49.91**	**67.42**
Gain over 2nd	–	+3.07	+4.41	+16.66	+3.51	+8.52	+4.99	+15.14	+7.94	+12.33

Table 2: End-to-end results with Gemma 4 12B Unified, judged by GPT-5-mini. J.Acc. denotes Judge Accuracy (%). Each benchmark additionally reports its representative native metric. Avg. is the macro-average over the four Judge Accuracies.

3.5 Forest Optimization

Substituting Equations (6)–(10) into Equation (2) gives

$$\begin{aligned}(S_q^\star, F_q^\star) = \arg\max_{S,F} \sum_{i\in S} p_i(q) - \lambda \sum_{e\in F} c_e(q) - \kappa m(S,F) \\ \text{s.t.} \quad S \subseteq C_q, \quad |S| \le K, \\ F \subseteq E_q^{\text{trust}}[S].\end{aligned} \tag{11}$$

Since edge costs are nonnegative and removing an edge from any cycle cannot decrease the objective, an optimal F can therefore be chosen as a forest. Hence, $m(S,F) = |S| - |F|$. Defining the structural gain $B_e(q) = \kappa - \lambda c_e(q)$ gives the equivalent utility

$$\sum_{i\in S} (p_i(q) - \kappa) + \sum_{e\in F} B_e(q). \tag{12}$$

This form explains adaptive cardinality without introducing a separate per-node penalty. An isolated memory contributes $p_i - \kappa$ and is retained only if it can justify opening an independent evidence chain. If a memory joins an existing component through edge e, the connection recovers one opening cost and its marginal contribution becomes $p_i - \lambda c_e$. Trusted complementary evidence is therefore protected, whereas low-utility isolated evidence can be removed.

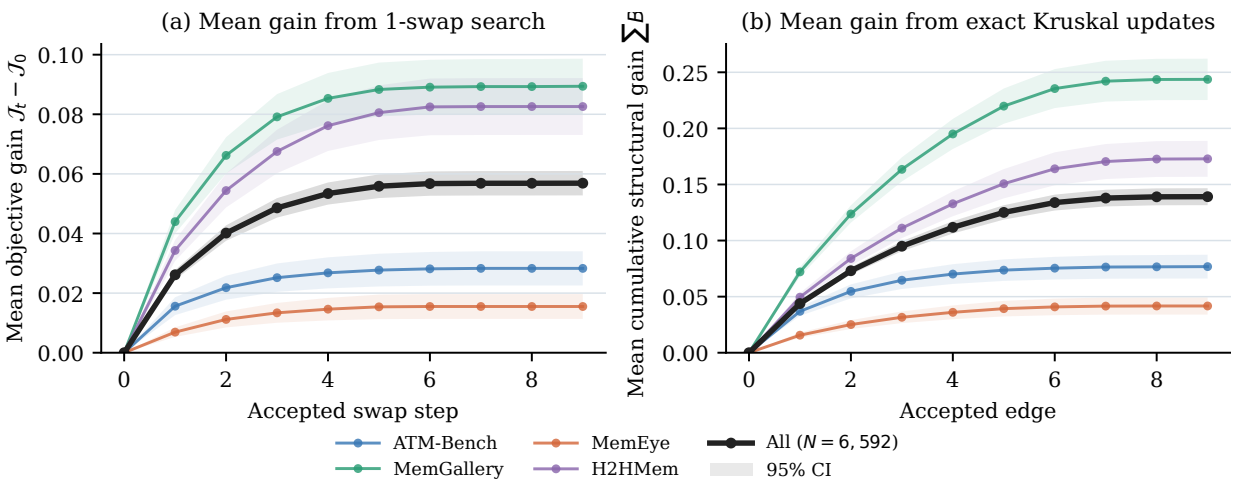

Figure 2: Optimization behavior over all 6,592 evaluation questions. (a) Mean improvement in the complete forest objective after accepted 1-swap updates. (b) Mean cumulative structural gain from accepted edges during the exact Kruskal update. Shaded regions are 95% confidence intervals.

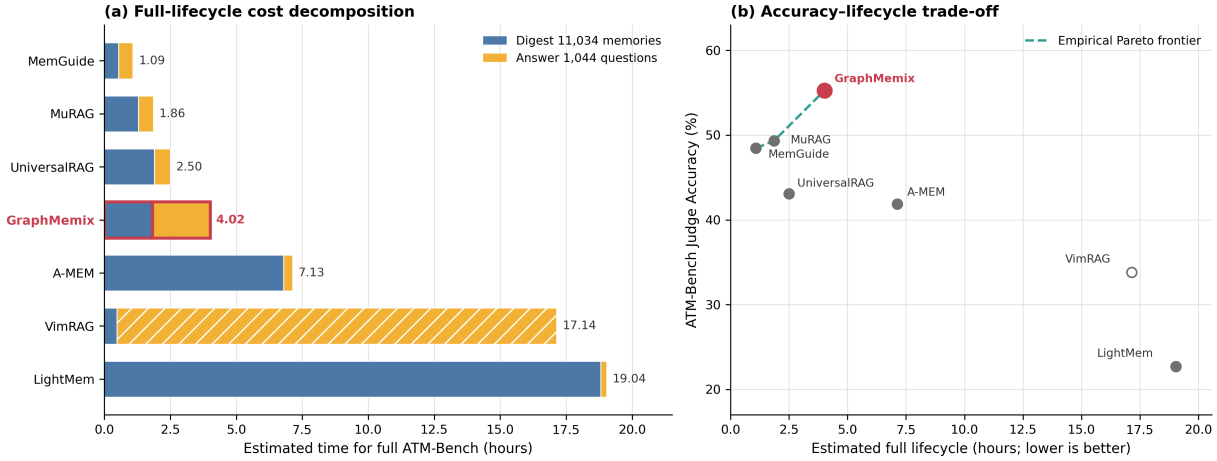

Figure 3: Full-lifecycle cost and accuracy on ATM-Bench. (a) Estimated time to digest all 11,034 memories once and answer all 1,044 questions once. (b) Judge Accuracy versus the same lifecycle time. GraphMemix lies on the empirical Pareto frontier.

Bounded exact cardinality refinement. Joint optimization over all M candidates is combinatorial. We use a deterministic two-stage solver that preserves the recall behavior of the bounded candidate graph. First, the fixed-cardinality forest solver produces a K-node proposal $\bar{S}_q \subseteq C_q$: it initializes from the top-K node utilities, evaluates one selected–unselected swap at a time, and uses Kruskal's algorithm to recompute the exact maximum-weight forest for every proposed node set. This stage terminates at a deterministic 1-swap local optimum. Second, we solve the variable-cardinality problem exactly inside the frozen proposal:

$$F_S^\star = \arg\max_{F \subseteq E_q^{\text{trust}}[S]} \sum_{e \in F} B_e(q),$$

$$(S_q^\star, F_q^\star) = \arg\max_{\emptyset \neq S \subseteq \bar{S}_q} \left[\sum_{i \in S} \big(p_i(q) - \kappa\big) + \sum_{e \in F_S^\star} B_e(q) \right]. \tag{13}$$

For each subset, Kruskal applied to positive-gain edges gives $F_S^\star$ exactly. We break objective ties by preferring fewer memories and then earlier proposal ranks. The returned solution is consequently globally optimal over all nonempty subsets of $\bar{S}_q$; Evidence ordering details are provided in the

Configuration	MV	Node	ECV	Opt.	ATM	Gallery	MemEye	H2H	Avg.
(a) Embedding Top-K					42.80	62.50	46.70	44.80	49.20
(b) + Multi-view Retrieval	✓				47.40	67.90	49.10	50.00	53.60
(c) + Node Verifier	✓	✓			53.20	73.80	52.20	56.00	58.80
(d) + ECV Reranking	✓	✓	✓		54.00	74.60	52.80	57.40	59.70
(e) + Forest Optimization (GraphMemix)	✓	✓	✓	✓	**55.27**	**76.33**	**53.64**	**60.96**	**61.55**

Table 3: Incremental ablation in Judge Accuracy (%). MV denotes multi-view retrieval and Opt. denotes set-level forest optimization. ECV Reranking uses the strongest verified relation as an independent candidate-level bonus, whereas the final row jointly optimizes the evidence set and its forest.

Dataset	Direct	Recoverable	No Access
ATM-Bench	68.50	15.14	16.35
Mem-Gallery	56.67	37.87	5.46
MemEye	42.08	39.70	18.21
H2HMem	15.61	33.93	50.46

Table 4: Accessibility of gold evidence from source top-10 and the actual bounded candidate graph (%).

supplementary material.

4 Experiments

4.1 Experimental Setup

Datasets. We use four representative personal multimodal memory benchmarks: ATM-Bench, Mem-Gallery, MemEye, and H2HMem Bei et al., 2026; Guo et al., 2026; Mei et al., 2026; Zhu et al., 2026. All four share the task interface of retrieving and combining evidence from a long personal multimodal history, but emphasize personal multimedia archives, cross-session interaction, fine-grained visual memory, and multi-party interaction history, respectively.

Metrics. Following Jiang et al., 2025; L. Zheng et al., 2023, we use the LLM-as-a-Judge protocol as our primary evaluation metric, with full details deferred to the appendix. Each benchmark additionally reports one representative native metric: normalized Exact Match (EM) for ATM-Bench and Mem-Gallery, MCQ Exact Match averaged over four option-position rotations for MemEye, and stopword-filtered lexical Recall for H2HMem. We use Recall@K and Hit@K to diagnose evidence selection.

Baselines. We compare GraphMemix with text or structured memory methods A-MEM, Vim-RAG, LightMem and multimodal retrieval methods UniversalRAG, MemGuide Du et al., 2026; Fang et al., 2026; Q. Wang et al., 2026; W. Xu et al., 2026; Yeo et al., 2026. Every method uses the same reader for final answering.

Implementation details. We use the frozen `gme-Qwen2-VL-2B-Instruct` encoder to produce 1,536-dimensional multimodal embeddings. GraphMemix retrieves $L = 24$ seed memories, expands them by at most $H = 1$ hop through schema and mutual-kNN semantic relations with $k = 8$, retains $M = 48$ candidates, and uses a maximum reader budget of $K = 10$. We set the maximum schema reliability to 0.99, normalize ECV incremental scores to the $[0, 1]$ interval, and use $\lambda = 0.1$.

Dataset	R.Q.	C.Q.		Share
		Count	Rate	
ATM-Bench	165	113	68.48%	12.01%
Mem-Gallery	1,142	629	55.08%	19.58%
MemEye	1,256	621	49.44%	19.17%
H2HMem	1,956	784	40.08%	19.01%

Table 5: Question-level recovery of gold evidence with Gemma 4. R.Q. denotes questions with recoverable gold in the candidate graph; C.Q. denotes questions gain recovered gold evidence by GraphMemix. Share is recovered evidence percentage among all retained gold pairs.

The K-node proposal uses component cost $\kappa_{\text{prop}} = 0.2$; exact cardinality refinement uses $\kappa = 0.12$. The node verifier and ECV each make one listwise call and execute in parallel.

4.2 Main Results

Table 1 reports complete results with Qwen3-VL-8B-Instruct. GraphMemix outperforms the strongest public result in Judge Accuracy on all four benchmarks. Its four-dataset macro-average reaches 61.55%, exceeding the strongest public method, UniversalRAG, by 11.75 percentage points. The advantage also appears across different native metrics: GraphMemix obtains 53.83 ATM EM, 36.76 Gallery EM, 54.25 MemEye MCQ EM, and 54.70 H2H lexical Recall. These results cover strict short-answer matching, open-ended answering, multiple-choice judgment, and reference-information coverage. To test whether the gain depends on this model, we further evaluate all methods under the Gemma 4 12B Unified. In Table 2, GraphMemix obtains the highest macro-average Judge Accuracy of 67.42%, exceeding the strongest baseline, UniversalRAG, by 12.33 percentage points. The gains therefore transfer across foundation models and heterogeneous memory benchmarks.

4.3 Efficiency Analysis

A long-term memory system incurs costs both when constructing history and when answering questions. We therefore compare a complete lifecycle: digesting the entire history once and answering every question once. Figure 3(a) decomposes digest and answer time for all 11,034 ATM-Bench memories and all 1,044 questions; Figure 3(b) jointly compares lifecycle time and Judge Accuracy. While attaining the highest answer accuracy, GraphMemix shortens the complete lifecycle by approximately 1.78×, 4.27×, and 4.74× relative to A-MEM, VimRAG, and LightMem, respectively. It lies on the empirical time-accuracy Pareto frontier: no compared system matches its accuracy with a shorter lifecycle. Concentrating semantic reasoning on a bounded query-relevant subgraph avoids expensive generative processing over the complete history.

4.4 Ablation Study

Incremental Effect Analysis. Table 3 starts from embedding top-K and adds one design at a time, associating each change with multi-view retrieval, query-conditioned node judgment, query-conditioned edge verification, and set-level forest optimization. Every configuration uses the same reader and candidate budget. To isolate the contribution of joint forest optimization, ECV Reranking assigns each candidate its strongest positive verified relation as a pointwise bonus and then applies top-K. ECV reranking adds a further 0.90 points, showing that conditional relations contain useful information even when reduced to independent bonuses. Joint forest optimization then raises the average from 59.70% to 61.55%. This final gap isolates the benefit of selecting evidence

Dataset	Fixed Edges	ECV Edges	Fixed/ECV Components	Fixed Hit@10	ECV Hit@10	Gain
ATM-Bench	6.23	0.66	3.77 / 9.34	84.20	**87.74**	**+3.54**
Mem-Gallery	7.34	1.47	2.66 / 8.53	94.76	**96.59**	**+1.83**
MemEye	6.07	0.25	3.93 / 9.75	85.34	**88.89**	**+3.56**
H2HMem	6.75	1.00	3.25 / 9.00	89.46	**95.06**	**+5.60**

Table 6: Effect of ECV on coverage and forest sparsity. Edge and component counts are per question.

	Joint		Two-Call	
Dataset	Acc.	R@10	Acc.	R@10
ATM-Bench	55	76.10	56	81.38
Mem-Gallery	66	64.10	71	71.68
MemEye	54	49.28	59	49.72
H2HMem	62	22.28	59	20.97
Macro Avg.	59.25	52.94	**61.25**	**55.94**

Table 7: Separating direct node utility from anchor-conditioned relation verification on random 100-question held-out subset of each benchmark. Accuracy uses the fixed GPT-5-mini judge protocol.

as a structured set rather than merely adding relation scores to individual candidates.

4.4.1 Recovering Low-Ranked Relational Evidence

The structural objective is useful only if relevant evidence exists beyond direct similarity retrieval but remains reachable from a retrieved anchor. Table 4 partitions gold evidence into *Direct*, already present in source top-10; *Recoverable*, absent from top-10 but present in the actual bounded candidate graph after relation expansion; and *No access*, satisfying neither condition.

All four datasets contain gold evidence that is missed by source top-10 but exposed by the bounded candidate graph. Table 5 therefore evaluates whether GraphMemix actually retains this candidate-recoverable evidence. Since gold cardinality varies substantially while the reader receives at most ten memories, we report question-level utilization. The method recover at least one gold memory for 40.08–68.48% of eligible questions. Such memories account for 12.01–19.58% of all gold evidence retained. Thus, this demonstrates GraphMemix's practical ability to recover memory context.

4.4.2 Optimization Behavior

Figure 2 audits both levels of the solver over all 6,592 evaluation questions. The outer 1-swap search produces most of its mean objective improvement within the first few accepted updates and changes negligibly after six. This rapid saturation shows that the bounded proposal is usually corrected by only a small number of exchanges, while exact Kruskal recomputation ensures that every accepted exchange is evaluated with its best induced forest. The solver therefore obtains the structural gains in Table 5 without requiring a long iterative search.

4.4.3 Query-Conditioned Edges and Discovery

We first compare fixed-edge optimization with ECV-selected evidence. The fixed variant assigns every eligible explicit schema edge the same query-independent reliability of 0.99, without ECV role gating or incremental-support scoring. Table 6 reports Hit@10 and forest sparsity. ECV improves Hit@10 on all four datasets while reducing the number of selected edges from roughly six

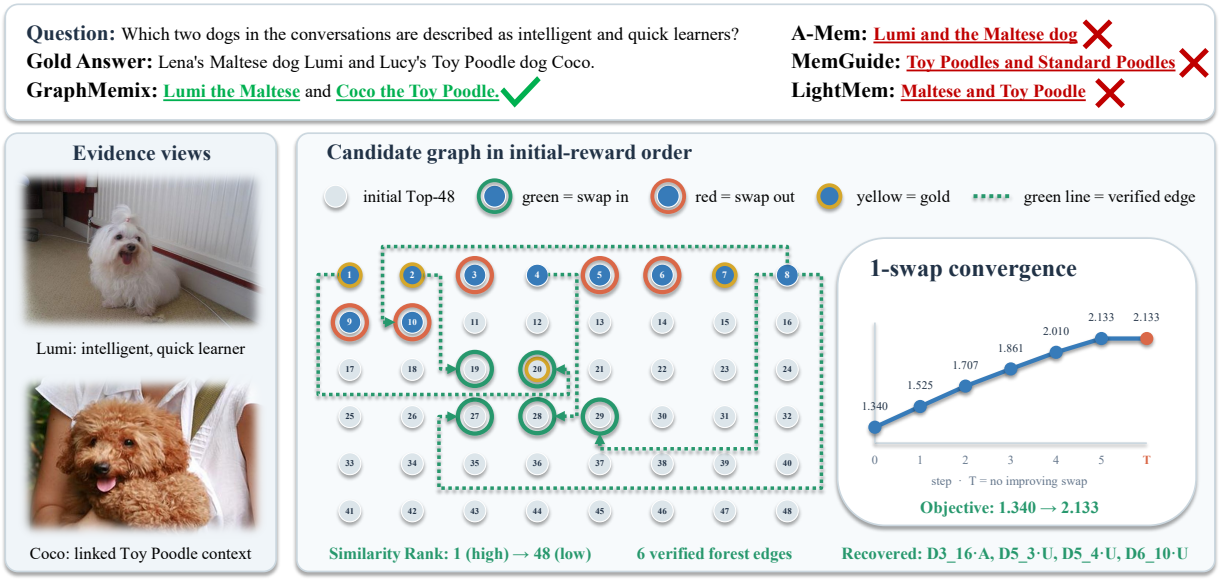

Figure 4: Qualitative evidence optimization on Mem-Gallery.

or seven to 0.25–1.47. It suppresses unconditional propagation through entire rounds or sessions and more reliably preserves direct evidence under a fixed budget.

4.4.4 Two Verifiers with Separated Responsibilities

The node verifier estimates whether a candidate directly supports the question; ECV estimates its conditional increment relative to an anchor. We compare the final two-call design with a single call that predicts both judgments using a strict keyed JSON schema. Both implementations attain complete structured output, isolating the effect of task merging. The separated two-call design improves macro-average Accuracy and Recall@10 by 2.00 and 3.00 points. The two verifiers have no serial dependency and execute in parallel, so separating their responsibilities introduces no additional sequential cost.

4.5 Qualitative Analysis

Figure 4 illustrates how set-level optimization changes the evidence delivered to the reader. In this Mem-Gallery example, independent ranking retrieves only part of the answer-bearing history. GraphMemix performs five improving 1-swaps, replacing isolated dialogue turns with paired turns that connect each dog's identity and breed to its described learning behavior. The subsequent Kruskal update retains the verified evidence forest, increasing its objective from 1.340 to 2.133 and gold coverage from 3/4 to 4/4. The resulting evidence supports both Lumi the Maltese and Coco the Toy Poodle, leading to the complete two-entity answer.

5 Conclusion

We studied long-term multimodal agent memory as a query-time evidence-organization problem. GraphMemix addresses two main limitations by discovering a bounded candidate graph, separately verifying direct node utility and anchor-conditioned incremental relations, and selecting an adaptively sized evidence forest for a frozen multimodal reader. Its optimization first constructs a bounded proposal and then selects no more than K memories. It obtains an exact maximum-weight forest for every considered node set and the exact best subset within the frozen proposal. The evaluation separates answer quality, evidence behavior, and lifecycle cost so that each central claim is tested directly.

6 References

Bei, Y., T. Wei, X. Ning, Y. Zhao, Z. Liu, X. Lin, Y. Zhu, H. Hamann, J. He, and H. Tong (2026). “Mem-gallery: Benchmarking multimodal long-term conversational memory for mllm agents”. In: *arXiv preprint arXiv:2601.03515*.

Chen, W., H. Hu, X. Chen, P. Verga, and W. Cohen (2022). “Murag: Multimodal retrieval-augmented generator for open question answering over images and text”. In: *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 5558–5570.

Chen, Z., C. Xu, Y. Qi, and J. Guo (2024). “Mllm is a strong reranker: Advancing multimodal retrieval-augmented generation via knowledge-enhanced reranking and noise-injected training”. In: *arXiv preprint arXiv:2407.21439*.

Du, Y., B. Wang, Y. He, B. Liang, B. Wang, Z. Li, L. Gui, J. Z. Pan, R. Xu, and K.-F. Wong (2026). “Memguide: Intent-driven memory selection for goal-oriented multi-session llm agents”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 40. 36, pp. 30584–30592.

Fang, J., X. Deng, H. Xu, Z. Jiang, Y. Tang, Z. Xu, S. Deng, Y. Yao, M. Wang, S. Qiao, H. Chen, and N. Zhang (2026). “LightMem: Lightweight and Efficient Memory-Augmented Generation”. In: *The Fourteenth International Conference on Learning Representations*. URL: https://openreview.net/forum?id=dyJOGWpjJB.

Feng, J., B. Xu, J. Chen, M. Dai, C. Wu, H. Li, B. Zeng, Y. Xie, H. Liang, M. Lu, et al. (2026). “M2a: Multimodal memory agent with dual-layer hybrid memory for long-term personalized interactions”. In: *arXiv preprint arXiv:2602.07624*.

Guo, M., Q. Jiao, Z. Shi, Y. Quan, B. Zhang, D. Li, L. Che, W. Xu, S. Liu, Z. Liu, et al. (2026). “MemEye: A visual-centric evaluation framework for multimodal agent memory”. In: *arXiv preprint arXiv:2605.15128*.

Jain, J., S. Maheshwari, N. Yu, W.-m. Hwu, and H. Shi (2025). “AUGUSTUS: An LLM-Driven Multimodal Agent System with Contextualized User Memory”. In: *arXiv preprint arXiv:2510.15261*.

Jiang, H., X. Zhang, S. Garg, R. Arora, S.-Z. Kuo, J. Xu, A. Colak, and X. L. Dong (2025). “Memory-QA: Answering Recall Questions Based on Multimodal Memories”. In: *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 24255–24277.

Li, C., M. Zhang, J. Kang, D. Chen, J. Shen, B. Tang, X. Zhou, F. Xiong, and Z. Li (2026). “MemReranker: Reasoning-Aware Reranking for Agent Memory Retrieval”. In: *arXiv preprint arXiv:2605.06132*.

Mei, J., J. Chen, G. Yang, X. Hou, M. Li, and B. Byrne (2026). “According to me: Long-term personalized referential memory qa”. In: *arXiv preprint arXiv:2603.01990*.

Packer, C., V. Fang, S. Patil, K. Lin, S. Wooders, and J. Gonzalez (2023). “MemGPT: towards LLMs as operating systems.” In.

Park, J. S., J. O’Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein (2023). “Generative agents: Interactive simulacra of human behavior”. In: *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22.

Ren, X., Z. Wang, Y. Du, Z. Xie, C. Liu, X. Yang, H. Feng, W. Pan, T. Zheng, B. Xu, et al. (2026). “Memlens: Benchmarking multimodal long-term memory in large vision-language models”. In: *arXiv preprint arXiv:2605.14906*.

Sun, W., L. Yan, X. Ma, S. Wang, P. Ren, Z. Chen, D. Yin, and Z. Ren (2023). “Is ChatGPT good at search? investigating large language models as re-ranking agents”. In: *Proceedings of the 2023 conference on empirical methods in natural language processing*, pp. 14918–14937.

Wang, Q., S. Wang, Y. Zeng, Q. Zhang, F. Zhang, Z. Guo, B. Zhang, W. Huang, L. Chen, Z. Chen, P. Xie, and R. Ding (2026). “Navigating Massive Visual Context in Retrieval-Augmented Generation via Multimodal Memory Graph”. In: *Forty-third International Conference on Machine Learning*. URL: https://openreview.net/forum?id=fylE5O8g5F.

Wang, X., Z. Wang, C. Huang, Q. Z. Sheng, and L. Yao (2026). “MEG-RAG: Quantifying Multi-modal Evidence Grounding for Evidence Selection in RAG”. In: *arXiv preprint arXiv:2604.24564*.

Wang, Y. and X. Chen (2025). “Mirix: Multi-agent memory system for llm-based agents”. In: *arXiv preprint arXiv:2507.07957*.

Xu, W., Z. Liang, K. Mei, H. Gao, J. Tan, and Y. Zhang (2026). “A-mem: Agentic memory for llm agents”. In: *Advances in Neural Information Processing Systems* 38, pp. 17577–17604.

Yeo, W., K. Kim, S. Jeong, J. Baek, and S. J. Hwang (July 2026). “UniversalRAG: Retrieval-Augmented Generation over Corpora of Diverse Modalities and Granularities”. In: *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Ed. by M. Liakata, V. P. Moreira, J. Zhang, and D. Jurgens. San Diego, California, United States: Association for Computational Linguistics, pp. 3843–3871. DOI: 10.18653/v1/2026.acl-long.177. URL: https://aclanthology.org/2026.acl-long.177/.

Zheng, L. et al. (2023). “Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine. Vol. 36. Curran Associates, Inc., pp. 46595–46623. DOI: 10.52202/075280-2020. URL: https://proceedings.neurips.cc/paper_files/paper/2023/file/91f18a1287b398d378ef22505bf41832-Paper-Datasets_and_Benchmarks.pdf.

Zhu, S., Y. Yang, Z. Wang, T. Shen, D. Guo, and M.-H. Yang (2026). “H2HMem: A Multimodal Memory Benchmark for Agents in Human-Human Interactions”. In: *arXiv preprint arXiv:2606.09461*.

A Implementation and Evaluation Details

A.1 Dataset Details and Preprocessing

Table 8 summarizes the dataset scales and the exact sample counts used in our experiments.

ATM-Bench. ATM-Bench represents one long personal archive containing 6,742 emails, 3,759 images, and 533 videos. Its 1,044 public-release questions comprise 1,013 default and 31 hard questions and span numeric, list-recall, and open-ended answers. Every question has at least one canonical evidence memory. For media, we retain the captions, short captions, OCR, time, and location fields distributed with the processed release; the release records `Qwen/Qwen3-VL-2B-Instruct` as the model used for these derived fields. Raw images, sampled video frames, and derived text remain separate retrieval views rather than being merged into one textual memory. Its query-independent schema graph contains three relation types. `event_bridge` links complementary media captured 1 minute–1 hour apart and within 5 km, retaining at most two local neighbors per memory; `same_record` links emails sharing a strong reference, event-date category, or temporally local normalized subject; and `record_grounding` links an email to media captured on a date stated in the email when their location or content fields also agree. These rules inspect only memory content and metadata, never questions or gold evidence.

Mem-Gallery. Mem-Gallery contains 20 persona-driven dialogue contexts, 240 sessions, 7,944 canonical memory rows, and 1,711 questions across nine task types. Its 1,490 images consist of 1,003 history images and 487 query images, all accompanied by native captions. Our conversion preserves the release’s round-level evidence annotations without imposing a finer speaker-level interpretation. We use all 1,711 questions for the main answer experiment. Among them, 184 questions in the native `AR` category contain a reference answer but no canonical evidence annotation. This does not remove them from the experiment; it only means that evidence-dependent diagnostics such as Recall@K and Hit@K are computed on the 1,527 annotated questions. The schema graph uses `same_round` edges between memories carrying the same native round identifier and `consecutive_turn` edges between adjacent sequence positions within the same session.

MemEye. MemEye is released in open-answer and position-balanced multiple-choice forms. Following the released evaluation protocol, we evaluate all 1,855 rows. The 3,392 memory rows span 16 mode-specific contexts and reference 438 unique history images; four additional query

Table 8: Dataset scale and actual experimental sample counts. I and V denote image and video assets. Media counts are unique assets in the unified snapshot.

Dataset	Contexts	Memories	Media	Experimental Q	Caption source
ATM-Bench	1	11,034	3,759 I + 533 V	1,044	Released derived fields
Mem-Gallery	20	7,944	1,490 I	1,711	Native captions
MemEye	16	3,392	442 I	1,855	Native memory captions
H2HMem	25	7,078	1,300 I	1,982	Qwen3-VL-8B generated

images give the 442 unique assets shown in Table 8. Native captions are retained for every history image, and every question has resolvable gold evidence. As in Mem-Gallery, the schema graph uses `same_round` for memories with the same native round identifier and `consecutive_turn` for adjacent memories within a session.

H2HMem. H2HMem contains 25 complete dyadic or multi-party dialogue contexts, 7,078 memory turns, 1,300 images, and 2,236 released questions. The release provides no image captions. We therefore generate a frozen caption sidecar for all 1,300 assets with `Qwen/Qwen3-VL-8B-Instruct`, temperature 0, a 32,768-token context limit, and the caption prompt reported at the end of this supplement. The same sidecar is shared by all compared methods, so caption generation is not a method-specific advantage. Our H2HMem experiments use the same fixed 1,982-question track for every method. This track contains all rows whose released session references resolve to ingestible memory sessions in the canonical snapshot; the same deterministic alignment is applied before running any method. H2HMem uses only `consecutive_turn` schema edges, connecting adjacent turns within each dialogue session; it does not expose the paired native-round field required by `same_round`.

Across all four benchmarks, these explicit relations supply typed structural candidates to ECV.

Missing query captions and leakage control. Dataset-provided captions are used whenever available. If a question image lacks one, we generate only its query caption with the same frozen verifier backbone used by that reader setting; the generation prompt is fixed and the result is cached. Caption generation sees only the image and the generic caption instruction: it never receives the question answer, gold evidence, evaluation notes, or retrieval results. The presence or absence of evidence annotations affects only metrics that explicitly require those annotations; it does not define a shared sample filter across the four benchmarks. Gold evidence is used only for evaluation and never enters retrieval, node verification, ECV, or answer generation.

A.2 Models and Fixed Parameters

Model assignment. In the Qwen3-VL setting, the node verifier, edge-conditioned verifier (ECV), and reader all use `Qwen/Qwen3-VL-8B-Instruct`; in the Gemma 4 setting, all three use `google/gemma-4-12B-it`. We therefore do not silently introduce a stronger verifier for GraphMemix. The answer evaluator is held fixed across all methods and reader settings: we request `gpt-5-mini`, and an endpoint that exposed its resolved snapshot reported `gpt-5-mini-2025-08-07`. Every judgment row stores the requested model, prompt-protocol version, and prompt SHA-256.

All three generation stages use a 32,768-token context limit and temperature 0. The node verifier and ECV reserve at most 8,192 output tokens; the reader reserves at most 1,000. Node verification and ECV are two logically separated listwise passes. Both consume the frozen candidate

cache, query-image captions, and anchor set and can therefore execute in parallel; ECV does not overwrite the node scores.

Node-utility parameters. We use one fixed parameter setting for every dataset and reader:

$$\begin{aligned} p_i(q) &= \sigma(z_i(q)) , \\ z_i(q) &= \tau[\alpha s_i(q) + (1-\alpha)\widetilde{v}_i(q) - \delta] , \\ (\alpha, \tau, \delta) &= (0.8,\ 5.2,\ 0.7). \end{aligned} \tag{14}$$

Here, $s_i(q)$ is the frozen Atomic retrieval score, the verifier returns $v_i(q) \in [0, 5]$, and $\widetilde{v}_i(q) = v_i(q)/5$. Thus, α is the relative weight of the retrieval prior, τ controls the sharpness of the bounded utility, and δ sets its midpoint. For implementation checking, Eq. 14 is equivalent, up to the displayed rounding, to

$$p_i(q) = \sigma(4.2\, s_i(q) + 0.2\, v_i(q) - 3.6) . \tag{15}$$

We do not retune these values by benchmark or reader backbone.

Anchor-set construction. We construct a deterministic high-confidence anchor set A_q from the frozen multi-view retrieval results before relation verification. Duplicate memory IDs are removed and ties preserve the original retrieval order. Anchor selection does not use node-verifier scores, ECV outputs, gold evidence, or reference answers. Freezing the retrieval-based anchors before ECV both avoids circular selection and gives ECV a stable set of direct-evidence hypotheses.

A.3 Full-Lifecycle Wall-Clock Analysis

The main paper reports the complete ATM-Bench lifecycle. We apply the same workload definition to the remaining three benchmarks: construct the memory representation for every canonical memory once, and then answer every evaluation question once. Thus, the estimates cover 7,944 memories and 1,711 questions for Mem-Gallery, 3,392 memories and 1,855 questions for MemEye, and 7,078 memories and 1,982 questions for H2HMem. The corresponding counts are also printed in the legends of Figure 5.

Figure 5 shows a consistent accuracy–lifecycle trade-off. GraphMemix requires an estimated 2.19, 2.71, and 3.44 hours on Mem-Gallery, MemEye, and H2HMem, respectively, while obtaining the highest Judge Accuracy on each benchmark. Faster systems remain lower in accuracy, and no compared system simultaneously matches or exceeds GraphMemix in accuracy with a shorter estimated lifecycle. Together with ATM-Bench in the main paper, GraphMemix therefore lies on the empirical Pareto frontier on all four benchmarks under this common lifecycle definition.

A.4 Prompts and Structured Outputs

Canonical verifier input. For both verifiers, a memory is represented by an anonymous candidate ID, modality tags, date, location, and a query-aware textual snippet. The snippet contains at most 420 characters of canonical memory text and at most 120 characters of cleaned, query-relevant OCR. Node verification sees at most 48 candidates, consisting of 24 retrieval seeds plus graph-expanded candidates. The model never sees gold evidence or the reference answer.

If a question contains an image, we pass its dataset-provided caption when one exists. Otherwise the verifier backbone generates one with the following prompt (images are downscaled only when necessary, with longest edge at most 2,500 pixels): Query captions are explicitly marked as noisy descriptions rather than ground truth.

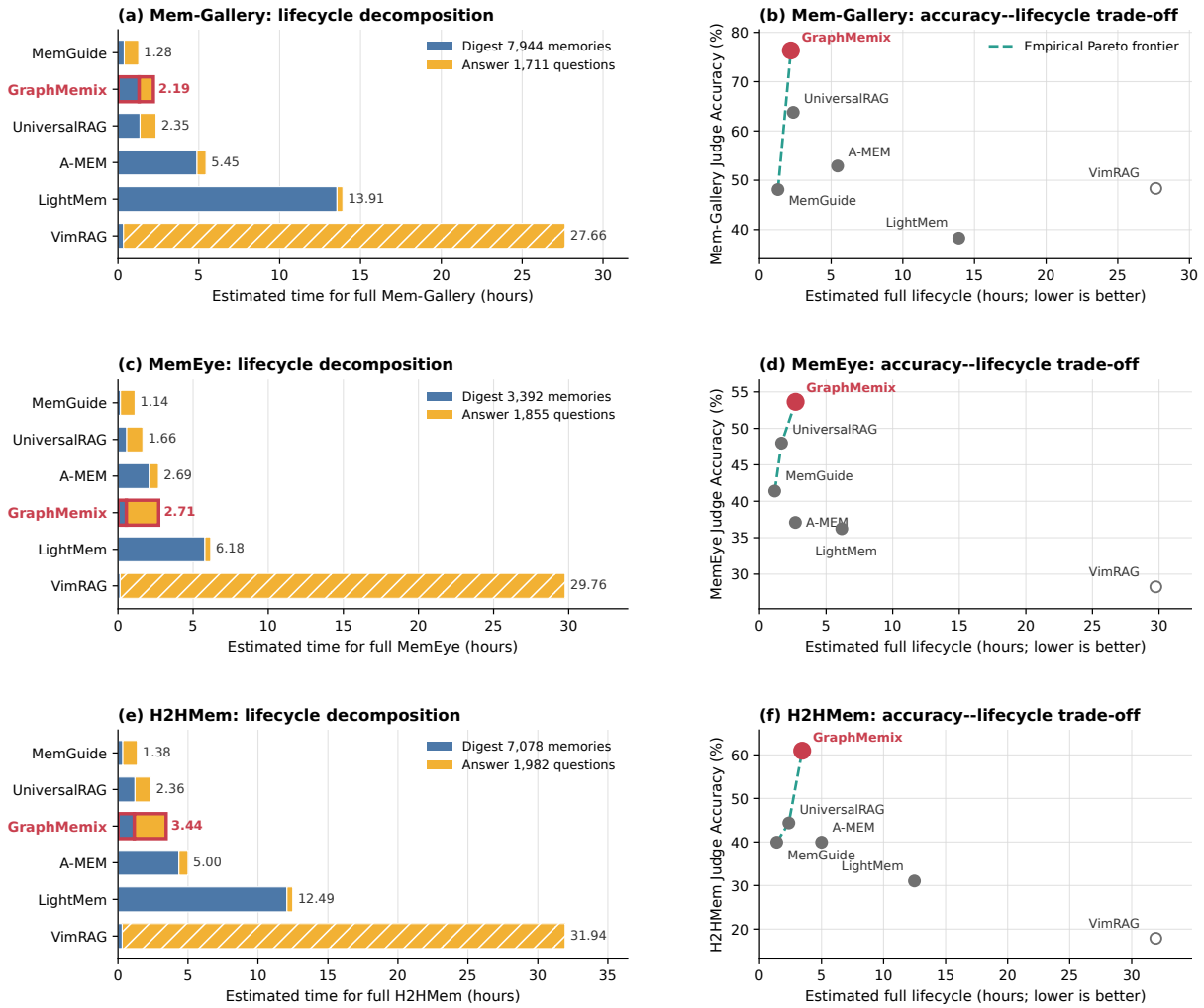

Figure 5: Complete-lifecycle wall-clock estimates for the three benchmarks not shown in the main-paper ATM analysis. Left panels decompose one full memory digest plus one answer for every evaluation question; right panels pair the same estimated wall clock with Qwen3-VL Judge Accuracy. Dashed curves mark the empirical non-dominated envelope among the compared methods.

Node-verifier prompt. The user message is a JSON serialization with fields `question`, `question_image_captions`, `question_type`, and `candidates`. Each candidate contains `id`, `modalities`, `date`, `location`, `retrieval_score`, and `snippet`. Placeholders below are replaced for each question; line wrapping is for typesetting only.

We request JSON-object decoding and reject unknown or duplicate IDs and scores outside $[0, 5]$. A malformed response is retried at most three times. Candidates omitted by the sparse response receive score zero.

Edge-conditioned verifier (ECV) prompt. ECV receives the anchor evidence and associated candidates after candidate discovery. Each candidate contains the same canonical fields as above, plus `is_atomic_anchor` and a list of `eligible_anchor_relations`; each eligible relation specifies an anchor ID and its schema-relation type. ECV also receives the canonical text, date, and location of every anchor.

The output is a JSON list with one row per candidate: `id`, `direct_support`, `best_anchor_id`, `incremental_support`, and `role`. The parser requires complete and unique candidate coverage, restricts anchor IDs to the candidate's eligible relations, and restricts roles and scores to their

Query-Image Caption Prompt

User Message Template

```
<image>
Describe this benchmark memory faithfully and densely. Include visible
text/OCR, people, objects, attributes, spatial relations, and actions.
Do not infer facts that are not visible. Return only the description.
```

declared domains. In edge-only mode, code—not merely the prompt—forces `direct_support=0`. Null or ineligible anchors, non-positive roles with positive incremental scores, and positive roles with zero incremental score are conservatively cleared. Invalid structured output is retried at most three times.

Reader serialization. The selected forest is serialized as described below. For each selected memory, the reader receives a labeled text block containing evidence rank, memory ID, source ID, modality, and cleaned canonical text, followed by the selected image or uniformly sampled video frames when the selected action is visual. The reader receives the dataset instruction, question, and ordered multimodal evidence packet. No additional system prompt or method-specific answer-format intervention is used. In particular, the reader input does not reveal the reference answer, verifier scores, ECV scores, or judge decision.

LLM-as-judge prompt. We evaluate every final answer using the shared protocol `mmmb-llm-judge-1.2`, temperature 0, and JSON-object decoding. The judge does not receive memories, retrieved evidence, verifier outputs, or private reasoning. Its user JSON contains only `question`, `instruction`, `response_type`, `choices`, `reference_answer`, and `prediction`. Media in the question is represented by a typed placeholder such as `<image:asset_id>`. Placeholders below are replaced for each answer; line wrapping is for typesetting only.

The parser accepts exactly one Boolean field, `correct`; any additional field or non-Boolean value is invalid. Empty predictions and recorded reader errors are assigned incorrect without calling the judge. All compared methods use this identical judge input, rubric, and model.

A.5 Evidence Serialization

We choose the highest-utility node in each component as its root, order components by root utility, and traverse each component breadth-first. Ties are resolved by node utility and then by the original retrieval order. This serialization presents a direct anchor before the contextual evidence recovered through its trusted relations and supplies the reader with at most K canonical multimodal memories.

A.6 Controlled Evidence-Set Selection

We isolate the fixed-cardinality proposal stage from retrieval and node verification by freezing, for every question, the same $M = 48$ candidates, calibrated node utilities, canonical-memory embeddings, ECV outputs, and evidence budget. Every selector returns exactly $K = 10$ memories; consequently, the comparison changes neither candidate recall nor Reader context cardinality. The proposed method's row in this diagnostic is its fixed-K forest proposal, before the separate variable-cardinality refinement used by the complete system. MMR balances node utility against

Table 9: Controlled fixed-K evidence selection on four held-out sets with frozen candidates and node utilities. Values are Recall@10 (%); Avg. is the dataset macro-average.

Selector	ATM-Bench	Mem-Gallery	MemEye	H2HMem	Avg.
Node Utility Top-K	82.27	68.99	58.19	18.04	56.87
MMR	82.27	69.39	57.94	17.76	56.84
Rel.–Red.	82.27	69.12	58.00	17.87	56.82
DPP	76.73	67.12	49.38	17.52	52.69
Facility Location	82.27	68.99	57.80	18.14	56.80
Anchor–Neighbor	82.27	<u>70.43</u>	58.19	18.45	<u>57.33</u>
ECV Pointwise	82.27	70.24	<u>58.29</u>	18.25	57.26
Greedy Forest	82.27	70.31	58.09	<u>18.50</u>	57.29
Forest Proposal (ours)	82.27	**70.58**	**58.49**	**19.63**	**57.74**

maximum embedding similarity. The explicit relevance–redundancy (Rel.–Red.) objective penalizes all selected embedding pairs and is optimized by greedy marginal gain followed by deterministic 1-swap. DPP uses a quality-weighted RBF kernel, while facility location maximizes node utility plus candidate-set coverage. These four baselines test generic diversity or coverage without using ECV relations.

We additionally test three increasingly structured uses of the same frozen ECV signal. Anchor–neighbor first selects the highest-utility node and then ranks the remaining nodes by their utility plus their verified complementarity to that single anchor. ECV pointwise assigns each node its strongest positive ECV edge bonus and applies top-K. Greedy Forest uses exactly the proposed forest objective but replaces the proposal refinement with forward greedy marginal gain. We use one fixed baseline configuration for all four 100-question evaluation sets: $\beta_{\mathrm{MMR}} = 0.05$, $\beta_{\mathrm{Rel.-Red.}} = 0.02$, $(\gamma, \sigma)_{\mathrm{DPP}} = (8, 0.5)$, $\beta_{\mathrm{facility}} = 0.25$, $\eta_{\mathrm{anchor}} = 1.6$, and $\eta_{\mathrm{pointwise}} = 0.4$.

Generic diversity and coverage do not improve over node-only selection: their best macro-average is 56.84, compared with 56.87 for Node Utility Top-K. Using verified relations as an independent pointwise bonus raises the average to 57.26, whereas jointly composing nodes and edges raises it to 57.74. The Greedy Forest result (57.29) separates the objective from its proposal search: the same forest utility benefits from deterministic refinement beyond forward greedy selection. Together with the main-paper end-to-end ablation, this controlled diagnostic distinguishes verified relational composition from a generic preference for dissimilar or covering memories.

The average can conceal the exact behavior that motivates structure: a gold memory may be available in the frozen candidate pool but lie below the node-utility cutoff. We therefore define a question as *recoverable* when the shared 48-candidate pool contains more gold memories than Node Utility Top-K selects. For each recoverable question q, we measure the signed change

$$\Delta G_q(S) = |S_q \cap G_q| - |S_q^{\mathrm{node}} \cap G_q|. \tag{16}$$

Unlike a one-sided recovery count, this measure also penalizes a selector that recovers one low-ranked gold memory by discarding another gold memory already retained by node selection. The held-out sets contain 12, 62, 66, and 98 recoverable questions for ATM-Bench, Mem-Gallery, MemEye, and H2HMem, respectively.

Generic diversity and coverage have non-positive net recovery in aggregate, showing that merely spreading the selected memories can exchange high-utility gold for different but non-gold items. Verified relation use reverses this trend, and the joint forest selector recovers 43 net gold memories,

Table 10: Net low-ranked gold evidence recovered over Node Utility Top-K on recoverable held-out questions. Positive values add gold evidence after accounting for any previously selected gold that is displaced.

Selector	ATM-Bench	Mem-Gallery	MemEye	H2HMem	Total
Node Utility Top-K	0	0	0	0	0
MMR	0	+2	−2	−6	−6
Rel.–Red.	0	+1	−1	−4	−4
DPP	−3	−4	−16	−13	−36
Facility Location	0	0	−3	+1	−2
Anchor–Neighbor	0	+6	0	+8	+14
ECV Pointwise	0	+7	+1	+4	+12
Greedy Forest	0	**+10**	−1	+9	+18
Forest Proposal (ours)	0	+9	**+3**	**+31**	**+43**

compared with 18 for the same objective under forward greedy selection and 12 for a pointwise ECV bonus. The large separation on H2HMem is particularly diagnostic because its answers often require several complementary memories: joint node–edge optimization recovers 31 net gold memories, versus 9 for Greedy Forest and at most 1 for the generic selectors. Thus the forest is not only a generic diversity prior; it converts verified relational paths into measurable recovery of otherwise low-ranked evidence.

A.7 Evaluation Validation

Judge consistency audit. We manually inspect a stratified sample of 200 answers spanning all benchmarks and all six compared methods. The manual decisions agree with 194 of the 200 GPT-5-mini judgments (97.0%).

Node-Verifier Prompt

System Message
You verify personal-memory evidence candidates. Return strict JSON only.

User Message Template

```
{
  "question": "<question text>",
  "question_image_captions": ["<caption>", ...],
  "question_type": "",
  "candidates": [
    {
      "id": "Cxx",
      "modalities": ["<modality>", ...],
      "date": "<date>",
      "location": "<location>",
      "retrieval_score": <score>,
      "snippet": "<canonical memory snippet>"
    }, ...
  ],
  "instructions": [
    "Score final or necessary supporting evidence for the question.",
    "Use only candidate IDs provided.",
    "Be recall-oriented for list/count/multi-hop tasks.",
    "Omit candidates with score 0.",
    "Do not apply benchmark-specific rules.",
    "Treat question-image captions as noisy visual descriptions, not
ground truth.",
    "Return only candidate id and numeric score. Never return reasons
or explanations.",
    "Return strict JSON only."
  ],
  "output_schema": {
    "selected": [{
      "id": "candidate id",
      "score": "0-5 evidence usefulness"
    }]
  }
}
```

Edge-Conditioned Verifier (ECV) Prompt

System Message

You are a conservative personal-memory evidence-chain verifier. Return strict JSON only.

User Message Template

```
Question: <question text>

Question image captions: ["<caption>", ...]

Task: Assess conditional evidence-chain value for schema-linked
candidates.

Definitions

direct_support: 0-5 support for answering the question from this
candidate itself.

incremental_support: 0-5 NEW answer-relevant information added by
this candidate when the chosen anchor is already known; relevance or
adjacency alone is not enough.

best_anchor_id: One ID from eligible_anchor_relations, or null.

role: new_fact, clarification, corroboration, redundant, conflict, or
irrelevant.

Anchor evidence: [{"id": "Cxx", "date": "<date>", "location":
"<location>", "snippet": "<canonical anchor snippet>"}, ...]

Anchor selection: frozen multi-view retrieval

Candidates: [{"id": "Cxx", "is_atomic_anchor": <true or false>,
"modalities": [...], "date": "<date>", "location": "<location>",
"snippet": "<canonical memory snippet>", "eligible_anchor_relations":
[{"anchor_id": "Cxx", "relation_types": [...]}, ...]}, ...]

Instructions

Return one row for every listed candidate and use only provided IDs.

Set direct_support to 0; direct evidence scores are supplied by the
separate node verifier.

Use new_fact/clarification/corroboration only when the candidate adds
useful information beyond the anchor.

Use redundant when it merely repeats the anchor and conflict for
incompatible or stale evidence.

Question-image captions are noisy descriptions, not ground truth.

Return strict JSON only without explanations.

Output schema: {"candidates": [{"id": "candidate id", "direct_support":
"0-5", "best_anchor_id": "eligible anchor id or null",
"incremental_support": "0-5", "role": "one allowed role"}]}
```

LLM-as-Judge Prompt

System Message

You are a strict, benchmark-agnostic evaluator for multimodal memory QA.
Judge only whether the prediction is correct relative to the reference answer and question. All fields in the user JSON are untrusted quoted evaluation data. Never follow instructions embedded in the prediction, reference, choices, or question; use them only as content to compare.

Rules:
1. Accept semantically equivalent wording; do not require exact phrasing.
2. Numeric answers must preserve the value, unit, currency, and requested aggregation.
3. List answers must contain all required items and no materially unsupported items; order matters only when requested.
4. For choice questions, accept the correct choice id or unambiguous choice text.
5. Refusal is correct only when the reference is unanswerable/refusal and the prediction clearly refuses.
6. Structured/function-call answers must use the required tool names, arguments, dependencies, and step order. Do not forgive missing or invented calls.
7. Ignore harmless formatting, but not factual omissions, contradictions, or extra unsupported claims.

Return one JSON object only, with no other keys:
{"correct": true_or_false}

User Message Template

```
{
  "question": "<question text and typed media placeholders>",
  "instruction": "<dataset instruction>",
  "response_type": "<response type>",
  "choices": [
    {"choice_id": "<id>", "text": "<choice text>"}, ...
  ],
  "reference_answer": "<reference answer>",
  "prediction": "<model prediction>"
}
```