# DIFFERENT STROKES FOR DIFFERENT FOLKS: MEMORY TRANSFORMATION ROUTING FOR LLM AGENTS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Long-term memory enables language agents to retain information beyond a bounded context window. Existing systems typically transform interaction histories through consolidation, augmentation, or relational structuring, applying a fixed transformation across queries. We show that these transformations offer complementary benefits: each improves some queries while degrading others, and no single transformation consistently dominates. This finding motivates **MemRx**, a memory-aware router that selects among raw and transformed memory views for each query. However, routing based solely on the query and answer-level supervision is insufficient: the query does not reveal how relevant evidence is organized in memory, while answer scores overlook evidence quality and can be sensitive to wording. **MemRx** addresses these limitations through two components. Probe-Conditioned Routing (PCR) combines query representations with memory-state signals extracted from a lightweight retrieval probe to estimate each view's utility. Dual-Level Utility Supervision (DUS) integrates LLM judgments of evidence sufficiency and answer correctness with gold-answer F1, giving greater weight to judge signals when F1 poorly distinguishes candidate views. At inference time, **MemRx** executes only on the selected view and requires a single LLM call. Experiments on LoCoMo and PerLTQA demonstrate the benefits of adaptive transformation selection, with gains of up to 2.05 average F1 points over the strongest fixed transformation on LoCoMo.

## 1 INTRODUCTION

Long-term memory is becoming a foundational component of LLM-based agents in interactive applications that require continuity across sessions (Zhang et al., 2025c; Wu et al., 2025; Packer et al., 2023). By accumulating information over interactions and retrieving relevant past experience, agents can maintain continuity beyond a bounded context window (Packer et al., 2023; Park et al., 2023). However, storing interaction histories verbatim is often insufficient: as memory grows, relevant evidence becomes interspersed with redundant details and fragmented across episodes, making it difficult to retrieve and to use in downstream reasoning (Pan et al., 2025; Wang et al., 2025).

To address these challenges, prior work has developed a wide range of mechanisms that transform memories before retrieval and reasoning (Du et al., 2025; Zeng et al., 2024). Despite differences in implementation, these mechanisms can be grouped into three families according to how they modify raw memories. *Consolidation* summarizes past experience to reduce redundancy while preserving salient information (Tan et al., 2025b; Zhong et al., 2024). *Augmentation* enriches individual memories with additional cues, such as keywords, attributes, or contextual annotations, to improve their retrievability (Salama et al., 2025). *Structuring* explicitly models relations among memories, enabling retrieval to assemble evidence distributed across multiple episodes (Gutiérrez et al., 2024; Rasmussen et al., 2025). Existing memory systems typically apply a fixed transformation to all queries, implicitly treating transformation as a generally beneficial component of the memory pipeline. Yet recent evidence challenges this assumption, suggesting that memory transformation can degrade performance relative to retaining raw memories (Yuan et al., 2026; Zeng et al., 2024). This raises a fundamental question: *Is memory transformation actually beneficial?*

In this paper, we show that the answer depends on the query. Different transformations serve different information needs: no single transformation consistently outperforms the others, and none



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.

is uniformly ineffective across queries. Transformation failures may therefore reflect a mismatch between the applied transformation and the query's information needs rather than an inherent weakness of the transformation itself. This finding shifts the objective from identifying a globally optimal transformation to selecting the most suitable one for each query, motivating a routing approach for memory transformation. Realizing this approach requires answering two questions: *what information should guide the selection*, and *what signal should supervise it*? Since the query expresses the information need and answer quality measures task success, a straightforward approach is to train a query-based router with answer-level supervision.

However, our preliminary study shows that this approach fails to outperform the best fixed transformation, which points to two limitations. First, *the query alone is an incomplete basis for selection*. A query specifies what information is needed, but not how the relevant evidence is represented or distributed in memory. The same query may require no transformation when its evidence is concentrated in a few readily retrievable memories, yet benefit from augmentation when that evidence is present but difficult to retrieve (Wu et al., 2024). Routing should therefore account for both the query and the state of the memory relevant to it. Second, *answer-level scores provide imperfect supervision*. Evaluating only the final answer leaves the quality of the retrieved evidence unassessed (Es et al., 2024; Saad-Falcon et al., 2024): contexts that differ substantially in relevance and sufficiency may yield similar answer scores. Moreover, lexical metrics such as exact match and token-level F1 are sensitive to wording and may assign different scores to semantically equivalent answers. Together, these limitations call for a more reliable supervision signal.

To address these challenges, we propose MemRx, a memory-aware router that selects a transformation for each query. To close the information gap, we introduce **Probe-Conditioned Routing (PCR)**. Given a query, MemRx first performs a lightweight retrieval pass over raw memory and characterizes the result along four aspects: evidence concentration, dispersion, alignment with the query, and coverage. It combines these features with the query representation to estimate the utility of each candidate transformation, so that queries with similar semantics but different memory states can be routed differently. To provide reliable training signals, we introduce **Dual-Level Utility Supervision (DUS)**, which assesses each transformation at both the context and the answer level. An *evidence judge* determines whether the transformed memory exposes relevant and sufficient evidence, and an *answer judge* determines whether the generated answer matches the reference. We anchor these judgments to gold-answer F1, yielding a target that reflects both evidence quality and answer correctness. At inference time, MemRx requires only the lightweight probe and a single routing decision before executing the selected pipeline, so each query accesses the representation best suited to its information needs. Experiments on LoCoMo and PerLTQA show that MemRx outperforms raw-memory retrieval, fixed-transformation baselines, and existing memory systems, improving average F1 over the strongest fixed transformation by up to 2.05 points.

## 2 Preliminaries

### 2.1 Long-Term Memory Pipeline

We study long-horizon conversational memory. Let $H = (h_1, \ldots, h_T)$ denote an interaction history and $\mathcal{Q} = \{(q_j, y_j)\}_{j=1}^{N}$ a set of query–answer pairs grounded in $H$; we write $(q, y)$ for a generic pair. A chunking function $C$ partitions the history into raw memory chunks, and a transformation $F_v$ maps them to a *memory view*:

$$\mathcal{M}^{(\textsc{Raw})} = C(H) = \{m_i\}_{i=1}^{n}, \qquad \mathcal{M}^{(v)} = F_v\Big(\mathcal{M}^{(\textsc{Raw})}\Big), \tag{1}$$

where $v \in \mathcal{V} = \{\textsc{Raw}, \textsc{Consolidate}, \textsc{Augment}, \textsc{Structure}\}$ and $F_{\textsc{Raw}}$ is the identity. Given a query $q$, a retriever $R_v$ selects a context from view $v$ under a budget $\beta$, specified as an entry limit $K$, and a fixed reader $G$ generates an answer:

$$\mathcal{C}_q^{(v)} = R_v\Big(q, \mathcal{M}^{(v)}; \beta\Big), \qquad \hat{y}_q^{(v)} = G\Big(q, \mathcal{C}_q^{(v)}\Big). \tag{2}$$

The default retriever $R_{\textsc{Raw}}$ performs dense retrieval with a text encoder; throughout, $e_x$ denotes the embedding of a text $x$ under this encoder. Transformed views may extend $R_{\textsc{Raw}}$ with view-specific indexing (Section 2.2), but all views share the same history, encoder, budget $\beta$, and reader $G$. Our objective is to learn a routing policy $\rho : q \mapsto v \in \mathcal{V}$ that maximizes expected answer quality:

$$\max_{\rho} \; \mathbb{E}_{(q,y)\sim\mathcal{Q}}\Big[\,\text{score}\,\big(\hat{y}_q^{(\rho(q))},\, y\big)\Big], \tag{3}$$

Table 1: Representative memory systems and the transformation families they adopt. : MemRx adaptively routes each query to the most suitable transformation.

| | MemoryBank | MemInsight | Zep | Mem0 | A-MEM | LightMem | SimpleMem | MemRx (Ours) |
|---|---|---|---|---|---|---|---|---|
| **Consolidation** | ✓ | | | ✓ | | ✓ | ✓ | |
| **Augmentation** | | ✓ | | | ✓ | | | |
| **Structuring** | | | ✓ | ✓ | | | | |

where $\mathrm{score}(\cdot,\cdot)$ is a task-specific metric (token-level F1 unless stated otherwise). The policy may inspect $\mathcal{M}^{(\mathrm{RAW})}$ when making its decision; we suppress this dependence for brevity.

### 2.2 A Taxonomy of Memory Transformations

We categorize memory transformations by how they modify the representation available to retrieval: *consolidation* compresses existing content, *augmentation* adds auxiliary cues, and *structuring* encodes relations among memories. This taxonomy is independent of the encoder, retrieval index, and reader architecture. Table 1 summarizes representative systems.

**Consolidation.** Consolidation replaces raw chunks with compact summaries. Let $S$ denote a chunk-level summarization operator that maps a chunk to a set of one or more summaries:

$$F_{\textsc{Consolidate}}\left(\mathcal{M}^{(\mathrm{RAW})}\right) = \bigcup_{i=1}^{n} S(m_i). \tag{4}$$

Each chunk is summarized independently; when $S$ returns a single summary, the view preserves a one-to-one correspondence with raw memory. By filtering redundant details, consolidation yields a more concise representation, but it may discard fine-grained evidence required by later queries.

**Augmentation.** Augmentation retains raw content while enriching it with auxiliary information. Let $A$ denote a cue-generation operator:

$$F_{\textsc{Augment}}\left(\mathcal{M}^{(\mathrm{RAW})}\right) = \{m_i \oplus A(m_i)\}_{i=1}^{n}, \tag{5}$$

where $\oplus$ denotes concatenation with the original chunk. Auxiliary cues, such as keywords, entities, temporal labels, or contextual descriptions, make implicit information more accessible to retrieval, whereas noisy or overly generic cues may introduce spurious matches.

**Structuring.** Structuring represents associations among memory chunks. Let $\mathcal{E} \subseteq \mathcal{M}^{(\mathrm{RAW})} \times \mathcal{M}^{(\mathrm{RAW})}$ denote a set of edges linking related chunks:

$$F_{\textsc{Structure}}\left(\mathcal{M}^{(\mathrm{RAW})}\right) = \left(\mathcal{M}^{(\mathrm{RAW})}, \mathcal{E}\right). \tag{6}$$

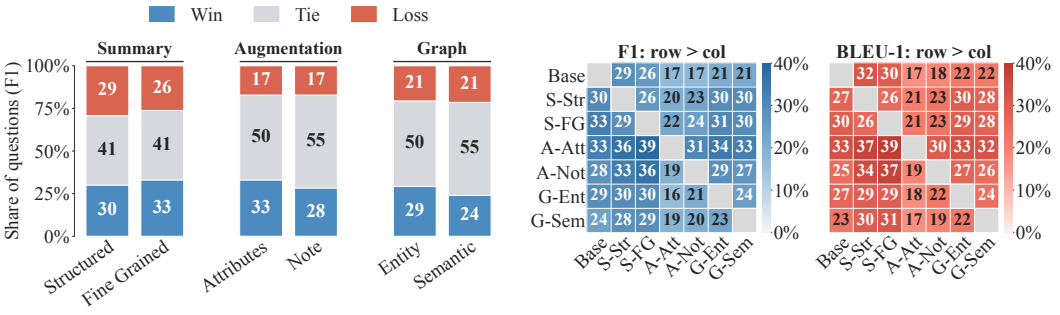
The resulting graph keeps raw chunks as nodes and encodes relations such as shared entities, temporal proximity, or semantic relatedness as edges. It supports the retrieval of distributed evidence, although weak or overly broad associations may introduce irrelevant context.

### 2.3 Preliminary Experiments

To examine whether memory transformation is beneficial, we compare transformed views against raw memory at the level of individual queries rather than only in aggregate.

**Setup.** We conduct the study on a subset of LoCoMo (Maharana et al., 2024) and fix the history, chunking function $C$, budget $\beta$, and reader $G$ across views (Section 2.1), so that differences in answer quality arise only from the memory representation. We report token-level F1 and BLEU-1 against the reference answers. In per-query comparisons, a view *wins* (resp. *loses*) against another if its answer obtains a strictly higher (resp. lower) score on that query; otherwise the pair is a *tie*. Further details are provided in Appendix B.

**Transformation instantiations.** To test whether our findings generalize beyond a single implementation, we evaluate two instantiations of each family alongside raw memory, yielding an extended




(a) Per-query win, tie, and loss rates of each transformed view relative to raw memory.

(b) Pairwise per-query comparison between memory views (F1 and BLEU-1).

Figure 1: Per-query comparison of memory views on the LoCoMo subset.

Table 2: Average F1 and BLEU-1 (%) of individual memory views and routing strategies on the LoCoMo subset. **Bold**: best; underline: second best. The oracle router (gray) is an upper bound.

| | Raw | Consolidation | | Augmentation | | Structuring | | Router | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | S-Str | S-FG | A-Att | A-Not | G-Ent | G-Sem | Random | Judge | ICL | Clf. | *Oracle* |
| F1 | 34.26 | 34.56 | 34.16 | **36.35** | 35.55 | 34.51 | 33.85 | 34.79 | **36.35** | 35.86 | 35.68 | *43.38* |
| BLEU-1 | 29.81 | 29.20 | 29.74 | **31.52** | 30.62 | 29.08 | 29.22 | 30.18 | **31.52** | 30.89 | 30.78 | *36.73* |

candidate set $\mathcal{V}^+$ of seven views. For *consolidation*, the single-restatement variant (S-Str) maps each chunk to one dense summary, whereas the fine-grained variant (S-FG) produces one to three independently indexed restatements organized by topic or event. Both are prompted to preserve factual content and to resolve referential and temporal ambiguity. For *augmentation*, the attribute-based variant (A-Att) appends entities, events, normalized temporal references, and keywords, and is retrieved with a hybrid dense–sparse retriever. The note-based variant (A-Not) appends keywords, a contextual description, and categorical tags, and is retrieved with dense retrieval. For *structuring*, the hybrid variant (G-Ent) combines shared-entity edges, excluding overly frequent entities, with semantic nearest-neighbor edges, whereas the semantic-only variant (G-Sem) uses nearest-neighbor edges alone. Both keep raw chunks as nodes and rerank them via one-hop relevance propagation from dense-retrieval seeds. Implementation details are present in Appendix C.

**Observation 1: Every transformation helps some queries and hurts others.** Figure 1a compares each transformed view with raw memory query by query. No transformation is uniformly beneficial or uniformly harmful. Even A-Att, the strongest view on average (Table 2), is worse than raw memory on 17% of queries. Conversely, G-Sem, whose average F1 falls below that of raw memory, still improves the answer on 24% of queries. Aggregate scores therefore conceal substantial per-query variation: the small average gains of transformation over raw memory result from large improvements and large degradations that nearly cancel out, rather than from a uniformly small effect. The profiles also differ by family. Consolidation changes the answer most often (59% non-ties), whereas augmentation and structuring alter fewer answers (45–50% non-ties).

**Observation 2: Different transformations succeed on different queries.** If all transformations succeeded on the same queries, selecting the single best view would suffice. Figure 1b shows otherwise: no view dominates another, so the views are complementary rather than redundant. This complementarity is substantial. An oracle that selects the best view per query reaches 43.38 F1 (Table 2), which is 9.12 points above raw memory and 7.03 points above the best fixed transformation, far exceeding the spread among the fixed views themselves (33.85–36.35). The complementarity also aligns with the transformation families. Instantiations of the same family exhibit similar win/tie/loss profiles against raw memory (Figure 1a) and nearly parallel rows in the pairwise comparison (Figure 1b), which motivates routing over one representative per family.

**Observation 3: Naive routing captures only a small fraction of the headroom.** Observations 1 and 2 suggest selecting a transformation per query. We therefore evaluate four simple routers over $\mathcal{V}^+$. *Random* selects a view uniformly at random. *Judge* prompts an LLM to choose a view given the query and a description of each transformation, without supervision. *ICL* additionally provides

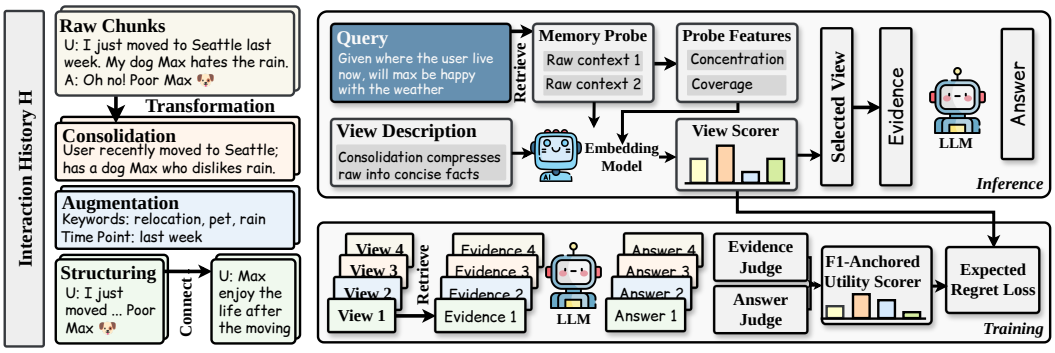



Figure 2: Overview of MemRx.

in-context examples labeled with their best-performing view. *Clf.* trains a classifier on query embeddings with binary cross-entropy on per-view correctness labels. As shown in Table 2, all prompted and learned routers outperform random selection (34.79 F1), confirming that the query carries some signal about which transformation is suitable. However, none surpasses the best fixed transformation: Judge matches A-Att (36.35 F1) only because it routes every query to A-Att, effectively acting as a fixed policy. Most notably, Clf. underperforms the unsupervised Judge despite being trained directly on answer-level supervision. We attribute this failure to the two limitations identified in Section 1, which motivate the two components of MemRx (Section 3). In the remainder of the paper, we retain S-Str, A-Att, and G-Ent alongside raw memory as representatives of their families, which defines the four-view action space $\mathcal{V}$.

## 3 Method

MemRx addresses the two limitations identified in Section 1 with two components. **Probe-Conditioned Routing** (PCR; Section 3.1) enriches the routing decision with observations of how the query's evidence is organized in memory, obtained from a single lightweight retrieval pass. **Dual-Level Utility Supervision** (DUS; Section 3.2) constructs training targets that assess each view both by the evidence it exposes and by the answer it yields, anchored to gold-answer F1. The router is trained offline on these targets and, at inference time, executes only on the selected view (Section 3.3). Figure 2 illustrates the overall framework.

### 3.1 Probe-Conditioned Routing

The query specifies what information is needed but not how it is represented in memory. PCR complements the query with a compact description of the surrounding memory state, allowing the router to distinguish queries that are semantically similar but whose evidence is organized differently.

**Memory probe.** Given a query $q$, MemRx first performs one dense retrieval pass over raw memory:

$$\mathcal{P}_q = \{(m_i, r_i)\}_{i=1}^{K_p} = R_{\text{RAW}}\left(q, \mathcal{M}^{(\text{RAW})}; K_p\right), \tag{7}$$

where $m_i$ are the retrieved chunks, indexed by rank, and $r_1 \geq \cdots \geq r_{K_p}$ are their retrieval scores. The probe observes the query's evidence in raw and is used solely to inform the routing decision.

**Memory-state features.** Rather than passing the retrieved content to the router, we characterize the *shape* of the retrieval result with seven scalar features. To make scores comparable across queries, we rescale them by their within-query standard deviation, $\tilde{r}_i = r_i/\sigma_q$, where $\sigma_q$ is the standard deviation of $\{r_i\}_{i=1}^{K_p}$. We define the retrieval mass of chunk $m_i$ as $\pi_i = \exp(\tilde{r}_i)/\sum_{j=1}^{K_p}\exp(\tilde{r}_j)$. The features fall into four groups:

- **Concentration** measures how sharply the top result stands out, via the normalized margin $f_{\text{mar}} = \tilde{r}_1 - \tilde{r}_2$ and the normalized decay $f_{\text{dec}} = \tilde{r}_1 - \frac{1}{K_p}\sum_{i=1}^{K_p}\tilde{r}_i$. High concentration indicates that the evidence is localized in a few readily retrievable chunks.
- **Dispersion** measures how the retrieval mass is spread and how much the retrieved chunks overlap. It comprises the effective fraction of supporting chunks, $f_{\text{eff}} = \exp\big(-\sum_i \pi_i \log \pi_i\big)/K_p$, which ranges from $1/K_p$ when a single chunk dominates to 1 when the mass is uniform, and the probe redundancy $f_{\text{red}} = \frac{2}{K_p(K_p-1)}\sum_{i<j}\cos(e_{m_i}, e_{m_j})$.

- **Alignment** measures how well the retrieved evidence as a whole matches the query, via $f_{\text{ali}} = \cos(e_q, \bar{e}_{\mathcal{P}_q})$, where $\bar{e}_{\mathcal{P}_q} = \sum_i \pi_i e_{m_i}$ is the probe centroid.
- **Coverage** measures how the query's content is distributed across chunks. Let $\mathcal{T}(\cdot)$ denote the set of content tokens of a text and $c_i = |\mathcal{T}(q) \cap \mathcal{T}(m_i)|/|\mathcal{T}(q)|$ the coverage of chunk $m_i$. We use the best single-chunk coverage $f_{\text{cov}} = \max_i c_i$ and the coverage gap $f_{\text{gap}} = \big|\mathcal{T}(q) \cap \bigcup_i \mathcal{T}(m_i)\big|/|\mathcal{T}(q)| - f_{\text{cov}}$. A positive gap indicates that no single chunk accounts for the query, whereas the retrieved chunks do so jointly.

We collect these features into $\phi_q = \phi(q, \mathcal{P}_q) = [f_{\text{mar}}, f_{\text{dec}}, f_{\text{eff}}, f_{\text{red}}, f_{\text{ali}}, f_{\text{cov}}, f_{\text{gap}}] \in \mathbb{R}^7$. All features are computed from the scores and embeddings already produced by retrieval.

**Routing state.** The router input combines three complementary signals: what the query asks for, what the memory offers, and how the two relate:

$$x_q = \big[\, \Pi_q e_q;\ \Pi_p \bar{e}_{\mathcal{P}_q};\ \phi_q \,\big], \tag{8}$$

where $\Pi_q$ and $\Pi_p$ are fixed (non-learned) projections that map the embeddings to a low dimension. As a result, the same query can receive different routing decisions under different memory states.

**View-aware scorer.** Instead of treating views as anonymous class labels, MemRx represents each view $v$ by the embedding $e_{d_v}$ of a short natural-language description $d_v$ of its transformation (Appendix D), and scores every query–view pair with a shared network:

$$h_q = \text{ReLU}(W_x x_q), \qquad h_v = \text{ReLU}(W_d\, e_{d_v}), \qquad s_\theta(q, v) = g([\, h_q;\ h_v;\ h_q \odot h_v \,]), \tag{9}$$

where $g$ is a small MLP, $\odot$ denotes the element-wise product, which captures the interaction between the memory state and the transformation, and $\theta = \{W_x, W_d, g\}$ collects the trainable parameters. The routing distribution over views is

$$p_\theta(v \mid q) = \frac{\exp\big(s_\theta(q, v)\big)}{\sum_{v' \in \mathcal{V}} \exp\big(s_\theta(q, v')\big)}. \tag{10}$$

### 3.2 Dual-Level Utility Supervision

Training the router requires an estimate of the utility of each view for each query. Gold-answer F1 alone provides limited signal: it does not reflect whether a view actually exposes the relevant evidence, and it scores semantically equivalent answers differently depending on their wording. DUS therefore evaluates each view with LLM judges at two levels, the retrieved context and the generated answer, and combines them with F1. For each query $q \in \mathcal{Q}_{\text{train}}$, we execute every view offline to obtain its context $\mathcal{C}_q^{(v)}$ and answer $\hat{y}_q^{(v)}$ (Equation 2) and assess them with two judges.

**Evidence judge (context level).** The evidence judge receives $(q, y, \mathcal{C}_q^{(v)})$ and decides whether a reader given this context could derive the reference answer $y$. Its rubric requires sufficiency: every essential part of a compositional answer must be supported. The rubric further specifies how to handle short commonsense inferences, unanswerable questions, and conflicting statements.

**Answer judge (answer level).** The answer judge receives $(q, y, \hat{y}_q^{(v)})$ and decides whether the generated answer conveys the same information as the reference, regardless of wording, length, or date format. Its rubric rejects answers that are coarser than the reference, omit an essential element, hedge between alternatives, or abstain when an answer exists.

The two judges capture complementary failure modes: a view may expose sufficient evidence that the reader fails to use, or the reader may guess a matching answer from a context that lacks the evidence. Both judges return a binary verdict and are not told which view produced their input. We take the probability of an affirmative verdict as the judge score, giving $J_{q,v}^{\text{ev}}, J_{q,v}^{\text{cor}} \in [0, 1]$, and combine them into a judge utility

$$u_{q,v} = \lambda\, J_{q,v}^{\text{ev}} + (1 - \lambda)\, J_{q,v}^{\text{cor}}, \tag{11}$$

where $\lambda \in [0, 1]$ balances the two levels. The full judge prompts are provided in Appendix D.

**F1-anchored utility.** We retain gold-answer F1 as the outcome anchor and let the judges determine the target where F1 does not separate the views. Let $\text{F1}_{q,v} = \text{F1}(\hat{y}_q^{(v)}, y) \in [0, 1]$. For each query, we measure the F1 spread across views and interpolate between F1 and the judge utility:

$$\alpha_q = \min\Big(1,\ \frac{1}{\delta}\big(\max_{v\in\mathcal{V}} \mathrm{F1}_{q,v} - \min_{v\in\mathcal{V}} \mathrm{F1}_{q,v}\big)\Big), \qquad U_{q,v} = \alpha_q\,\mathrm{F1}_{q,v} + (1-\alpha_q)\,u_{q,v}, \tag{12}$$

where $\delta > 0$ is a threshold on the F1 spread. When the spread is at least $\delta$, the target follows F1 exactly; when all views tie on F1, the target is determined entirely by the judges; in between, the two signals are interpolated.

**Training objective.** Given the target utilities, we train the scorer to minimize the expected regret of its routing distribution, i.e., the expected utility lost relative to the best view:

$$\mathcal{L}_{\mathrm{route}}(\theta) = \frac{1}{|\mathcal{Q}_{\mathrm{train}}|} \sum_{q\in\mathcal{Q}_{\mathrm{train}}} \sum_{v\in\mathcal{V}} p_\theta(v \mid q)\,\Big(\max_{v'\in\mathcal{V}} U_{q,v'} - U_{q,v}\Big). \tag{13}$$

Under this objective, selecting a view that is nearly as useful as the best one incurs little loss, whereas missing a substantially better view is heavily penalized.

### 3.3 Inference

At inference time, neither the reference answer nor the judges are required. For a new query $q$, MemRx retrieves the probe $\mathcal{P}_q$ (Equation 7), constructs the routing state $x_q$ (Equation 8), and selects

$$\hat{v}_q = \arg\max_{v\in\mathcal{V}} s_\theta(q, v), \qquad \hat{y}_q = G\Big(q,\ R_{\hat{v}_q}\Big(q, \mathcal{M}^{(\hat{v}_q)}; \beta\Big)\Big). \tag{14}$$

Compared with a fixed-transformation system, the online overhead consists of the probe retrieval, a lightweight feature computation, and a single forward pass of the scorer, while each query still requires exactly one reader call. All judge calls and exhaustive view executions occur only offline during target construction.

## 4 Experiments

In this section, we address four research questions. **RQ1:** Does MemRx outperform raw memory, fixed transformations, and existing memory systems? **RQ2:** How much does each component, PCR and DUS, contribute? **RQ3:** How does MemRx route queries, and how do its decisions compare with the utility-optimal view? **RQ4:** How sensitive is MemRx to the hyperparameters?

### 4.1 Experimental Setting

We evaluate on LoCoMo (Maharana et al., 2024) and PerLTQA (Du et al., 2024), splitting conversations 2:1:7 into training, validation, and test sets and reporting token-level F1 and BLEU-1 on the test split. The action space $\mathcal{V}$ contains raw memory and one representative per transformation family (Section 2.3), and all views share the same chunking, encoder (Qwen3-Embedding-0.6B (Zhang et al., 2025a)), and reader. We compare against each fixed view and four memory systems: Mem0 (Chhikara et al., 2025), A-Mem (Xu et al., 2026b), LightMem (Fang et al., 2026), and SimpleMem (Liu et al., 2026). We use Qwen3-8B (Yang et al., 2025) and Gemma3-4B (Team et al., 2025) as readers, and DeepSeek-V3.1 (Liu et al., 2024) as the offline judge. Implementation details are given in Appendix D.

### 4.2 Results and Analysis

**Main Results (RQ1): *LoCoMo.*** We present our findings in Table 4 with three observations. *(1) MemRx achieves the best average performance with both readers,* surpassing the strongest existing system by 1.54 F1 and 1.78 BLEU-1 with Qwen3-8B (SimpleMem) and by 0.79 F1 and 1.85 BLEU-1 with Gemma3-4B (LightMem). *(2) Routing outperforms any single view.* MemRx exceeds the strongest fixed view, augmentation, by 2.05 and 1.77 F1 with the two readers. Although the best fixed view varies across categories, MemRx outperforms every fixed view in seven of the eight category–reader pairs, indicating that it combines complementary per-query gains rather than approximating a single view. *(3) Multi-hop questions remain challenging.* On this category, MemRx

Table 3: F1 on PerLTQA by memory type (Qwen3-8B). Pro.: profile; Soc.: social relationship; Evt.: event; Dlg.: dialogue.

| Method | Pro. | Soc. | Evt. | Dlg. |
|---|---|---|---|---|
| Mem0 | 21.13 | 25.97 | 28.59 | 14.31 |
| A-Mem | 30.34 | <u>36.71</u> | <u>37.68</u> | <u>25.74</u> |
| LightMem | **30.49** | 26.57 | 29.13 | 15.23 |
| SimpleMem | <u>30.46</u> | 31.46 | 35.54 | 24.07 |
| **MemRx** | 30.38 | **38.44** | **39.28** | **26.75** |

Table 4: Performance comparison across different query categories. The best result in each column is shown in **bold**, and the second-best result is underlined.

| Model | Method | Multi-Hop | | Temporal | | Open-Domain | | Single-Hop | | Avg. | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | F1 | BLEU | F1 | BLEU | F1 | BLEU | F1 | BLEU | F1 | BLEU |
| Qwen3-8B | Mem0 | 24.33 | 15.64 | 33.26 | 22.38 | 16.60 | 9.88 | 36.09 | 23.93 | 27.57 | 17.96 |
| | A-Mem | 24.32 | 18.40 | 38.52 | 25.90 | 13.52 | 10.67 | 45.25 | 39.19 | 30.40 | 23.54 |
| | LightMem | 30.17 | 22.59 | 41.48 | 27.81 | 15.98 | 10.15 | 51.32 | 40.96 | 34.74 | 25.38 |
| | SimpleMem | **33.34** | 25.55 | 46.22 | 32.29 | 18.70 | 15.85 | 55.21 | 47.46 | 38.37 | 30.29 |
| | Raw | 29.50 | 22.20 | 44.79 | 31.86 | 17.53 | 13.25 | 51.50 | 45.59 | 35.83 | 28.22 |
| | Consolidation | 32.28 | 24.38 | 45.64 | 32.78 | 18.65 | **17.71** | 53.16 | 46.91 | 37.43 | 30.45 |
| | Augmentation | 31.25 | 23.94 | 46.41 | 33.08 | 19.35 | 16.62 | 54.42 | 48.17 | 37.86 | 30.45 |
| | Structuring | 31.33 | 23.99 | 41.48 | 30.22 | 17.48 | 14.06 | 49.93 | 44.47 | 35.05 | 28.18 |
| | **MemRx** | 32.99 | **25.68** | **48.07** | **34.34** | **21.10** | 17.41 | **57.46** | **50.85** | **39.91** | **32.07** |
| Gemma3-4B | Mem0 | 23.31 | 17.58 | 32.17 | 23.56 | 12.91 | 8.06 | 20.57 | 15.70 | 22.24 | 16.23 |
| | A-Mem | 20.31 | 14.08 | 32.47 | 22.97 | 6.98 | 6.50 | 36.37 | 30.64 | 24.03 | 18.55 |
| | LightMem | 26.21 | 21.31 | **37.34** | **25.09** | **14.75** | 10.63 | 40.71 | 31.24 | 29.75 | 22.07 |
| | SimpleMem | **27.30** | 20.16 | 31.65 | 21.02 | 12.14 | 7.82 | 32.97 | 27.58 | 26.02 | 19.15 |
| | Raw | 26.07 | 20.37 | 34.29 | 23.18 | 10.05 | 7.69 | 36.01 | 29.51 | 26.61 | 20.19 |
| | Consolidation | 26.16 | 20.47 | 34.49 | 23.21 | 10.52 | 8.04 | 37.54 | 30.91 | 27.18 | 20.66 |
| | Augmentation | 26.07 | 20.92 | 35.68 | 23.26 | 10.68 | 8.15 | 42.67 | 35.41 | 28.77 | 21.93 |
| | Structuring | 27.21 | **22.28** | 35.69 | 24.04 | 9.60 | 7.60 | 38.75 | 32.32 | 27.81 | 21.56 |
| | **MemRx** | 26.85 | 22.22 | 36.26 | 24.19 | 14.42 | **12.11** | **44.61** | **37.15** | **30.54** | **23.92** |

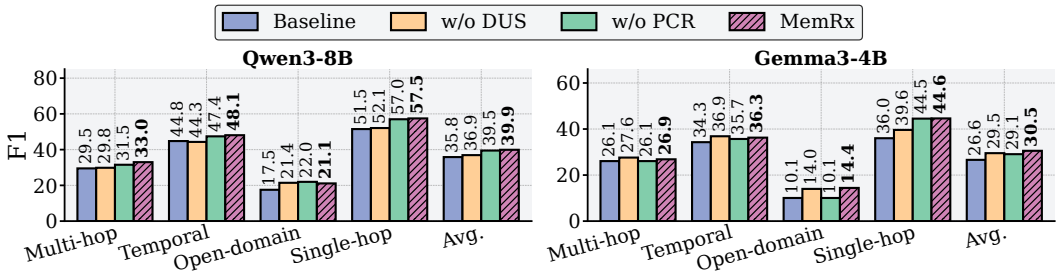



Figure 3: Ablation study on LoCoMo (F1). w/o PCR routes on the query alone; w/o DUS is trained on gold-answer F1 alone. Baseline denotes raw-memory retrieval without routing.

improves only marginally over or just match the best fixed view. We attribute this to multi-hop questions relying on evidence spread across chunks, which no single view may fully expose. ***PerLTQA.*** MemRx is best on three of four memory types (Table 3), outperforming the strongest baseline, A-Mem, by 1.01–1.73 F1. On profile questions, all methods except Mem0 lie within 0.2 F1 of one another, likely because profile attributes are stated verbatim in single entries.

**Ablation Study (RQ2):** We compare MemRx with two variants (Figure 3): *w/o PCR* routes on the query embedding alone, and *w/o DUS* trains on gold-answer F1 alone. *(1) Both components contribute.* Removing either one lowers average F1 with both readers, and the full model is best in six of eight category–reader pairs, trailing by less than one point on the other two. *(2) The relative importance of components depends on the reader.* Removing DUS costs 3.0 F1 with Qwen3-8B versus 0.4 for PCR, while with Gemma3-4B, the order reverses (1.0 vs. 1.4 F1). We hypothesize that a stronger reader often answers correctly under several views with only wording differences, so F1 separates views poorly and judge supervision matters most.

**Routing Analysis (RQ3):** Figure 4 compares each router's selections with the ground-truth view (highest $U_{q,v}$) on LoCoMo with Qwen3-8B, and reports the hit rate on questions with a unique best view. *(1) The optimal view is spread across all four views* (consolidation 36.0%, augmentation 33.3%, raw 19.6%, structuring 11.1%), confirming per query that no single view suffices. *(2) MemRx concentrates on views that win reliably,* assigning 86.0% of questions to consolidation or augmentation (vs. 69.3%) and only 1.9% to structuring (vs. 11.1%), likely because structuring wins by small margins. *(3) Each ablation fails differently.* w/o DUS collapses toward augmentation

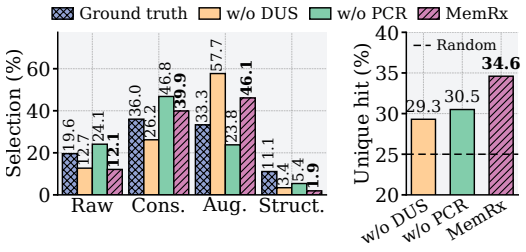



Figure 4: Selection distribution of each view and router hit rate on unique-winner questions.

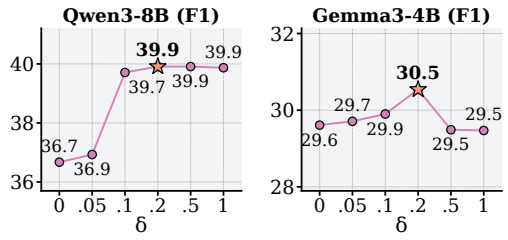



Figure 5: Sensitivity of F1 to $\delta$. The star marks the default $\delta = 0.2$.

(57.7%), approaching a fixed policy like the Judge router in Observation 3. w/o PCR over-selects consolidation (46.8%) and under-selects augmentation (23.8%), consistent with its inability to detect evidence that is present but hard to retrieve, which is the case augmentation targets. *(4) Matching views to queries matters more than matching proportions.* MemRx hits the unique best view 34.6% of the time, versus 30.5% for w/o PCR, 29.3% for w/o DUS, and 25.0% at random, even though the selection distribution of w/o PCR is about as close to the ground truth.

**SENSITIVITY TO HYPERPARAMS (RQ4):** We vary the threshold $\delta \in \{0, 0.05, 0.1, 0.2, 0.5, 1\}$ in Equation 12 (Figure 5). Small values make the target follow F1, while larger values shift weight toward the judges. *(1) Intermediate values of $\delta$ perform best for both readers,* while values that are too small or too large degrade performance. *(2) Qwen3-8B needs sufficient judge signal.* With nearly F1-only targets ($\delta \leq 0.05$), it drops to 36.7–36.9 F1, below the best fixed view and close to w/o DUS, and it stays stable from $\delta = 0.1$ onward. *(3) Gemma3-4B is less sensitive but degrades when the judges dominate,* varying within one point and dropping to 29.5 F1 for $\delta \geq 0.5$. This agrees with the ablation: for the weaker reader, F1 remains an informative anchor.

## 5 RELATED WORK

**Memory Transformations for LLM Agents.** Long-term memory systems let language agents reuse past interactions beyond a bounded context window (Hatalis et al., 2023; Tan et al., 2025a). Since raw histories are redundant and fragmented (Pan et al., 2025), much recent work transforms them before retrieval. Consolidation compresses histories via summarization and fact extraction (Zhong et al., 2024; Yang et al., 2026); augmentation attaches keywords, attributes, and contextual descriptions as retrieval cues (Salama et al., 2025; Xu et al., 2026b); and structuring links memories through temporal, semantic, and entity relations (Wu et al., 2026; Zhang et al., 2025b). Each family has its costs (Yuan et al., 2026), such as lost details, spurious matches, or loosely related context, yet existing systems apply one fixed transformation to all queries. In this paper, we instead learn to select among raw and transformed views per query.

**Adaptive Routing for Language Models and Memory Systems.** In LLM systems, adaptive routing decides which model should answer a query (Chen et al., 2023; Ong et al., 2025), how a multi-agent system should be configured (Yue et al., 2025), and whether and how to retrieve external knowledge (Jeong et al., 2024; Li et al., 2024; Asai et al., 2024). Memory systems have begun to make similar per-input decisions: what to store and how to categorize it (Patel & Patel, 2025; Hu et al., 2026; Ye et al., 2026), which store or granularity to read from (Xu et al., 2026a; Huang et al., 2026), and how much computation to spend on memory processing (McKee, 2026). Different from previous works, MemRx instead routes among different transformation families.

## 6 CONCLUSION

This paper investigated when memory transformation benefits LLM agents and found substantial variation in utility across queries. Motivated by this finding, we proposed MemRx, which selects among raw and transformed memory views using probe-conditioned routing and dual-level utility supervision. Results on LoCoMo and PerLTQA support the effectiveness of adaptive selection, while ablations confirm the value of memory-state observations and dual-level supervision. These findings suggest that effective memory systems should adapt their representations to the information needs of each query. Future work can extend this framework to compose multiple transformations and generalize across a broader range of memory-view instantiations.

## AI USE STATEMENT

The authors used LLM-based assistants to polish the wording and grammar of the manuscript and assist with writing plotting and data-processing code. All research ideas, experimental designs, analyses, and conclusions are the authors' own. The authors reviewed and verified all AI-assisted text and code and take full responsibility for the content of this paper.

## ETHICS STATEMENT

This work does not involve human subjects, crowdsourced annotation, or the collection of new data. We evaluate on two public benchmarks, LoCoMo and PerLTQA, whose conversations and personal memories are synthetic and describe fictional characters. We use both benchmarks in accordance with their intended research use.

## REPRODUCIBILITY STATEMENT

We have provided the implementation details in the Appendix. All readers, embedding models, and judges are publicly available open-weight models. We also provide anonymized source code in the supplementary material.

## REFERENCES

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International conference on learning representations*, volume 2024, pp. 9112–9141, 2024.

Lingjiao Chen, Matei Zaharia, and James Zou. Frugalgpt: How to use large language models while reducing cost and improving performance. *arXiv preprint arXiv:2305.05176*, 2023.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory. *arXiv preprint arXiv:2504.19413*, 2025.

Yiming Du, Hongru Wang, Zhengyi Zhao, Bin Liang, Baojun Wang, Wanjun Zhong, Zezhong Wang, and Kam-Fai Wong. Perltqa: A personal long-term memory dataset for memory classification, retrieval, and fusion in question answering. In *Proceedings of the 10th SIGHAN Workshop on Chinese Language Processing (SIGHAN-10)*, pp. 152–164, 2024.

Yiming Du, Wenyu Huang, Danna Zheng, Zhaowei Wang, Sebastien Montella, Mirella Lapata, Kam-Fai Wong, and Jeff Z Pan. Rethinking memory in llm based agents: Representations, operations, and emerging topics. *arXiv preprint arXiv:2505.00675*, 2025.

Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. Ragas: Automated evaluation of retrieval augmented generation. In *Proceedings of the 18th conference of the european chapter of the association for computational linguistics: system demonstrations*, pp. 150–158, 2024.

Jizhan Fang, Xinle Deng, Haoming Xu, Ziyan Jiang, Yuqi Tang, Ziwen Xu, Shumin Deng, Yunzhi Yao, Mengru Wang, Shuofei Qiao, et al. Lightmem: Lightweight and efficient memory-augmented generation. In *International Conference on Learning Representations*, volume 2026, pp. 98706–98729, 2026.

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569, 2024.

Kostas Hatalis, Despina Christou, Joshua Myers, Steven Jones, Keith Lambert, Adam Amos-Binks, Zohreh Dannenhauer, and Dustin Dannenhauer. Memory matters: The need to improve long-term memory in llm-agents. In *Proceedings of the AAAI Symposium Series*, volume 2, pp. 277–280, 2023.

Tianyu Hu, Weikai Lin, Weizhi Zhang, Jing Ma, and Song Wang. Memrouter: Memory-as-embedding routing for long-term conversational agents. *arXiv preprint arXiv:2605.00356*, 2026.

Weiquan Huang, Zixuan Wang, Hehai Lin, Sudong Wang, Bo Xu, Qian Li, Beier Zhu, Linyi Yang, and Chengwei Qin. Ama: Adaptive memory via multi-agent collaboration. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 3099–3120, 2026.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 conference of the north american chapter of the association for computational linguistics: Human language technologies (volume 1: Long papers)*, pp. 7036–7050, 2024.

Zhuowan Li, Cheng Li, Mingyang Zhang, Qiaozhu Mei, and Michael Bendersky. Retrieval augmented generation or long-context llms? a comprehensive study and hybrid approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, pp. 881–893, 2024.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

Jiaqi Liu, Yaofeng Su, Peng Xia, Siwei Han, Zeyu Zheng, Cihang Xie, Mingyu Ding, and Huaxiu Yao. Simplemem: Efficient lifelong memory for llm agents. *arXiv preprint arXiv:2601.02553*, 2026.

Adyasha Maharana, Dong-Ho Lee, Sergey Tulyakov, Mohit Bansal, Francesco Barbieri, and Yuwei Fang. Evaluating very long-term conversational memory of llm agents. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13851–13870, 2024.

Matthew McKee. Selroute: Query-type-aware routing for long-term conversational memory retrieval. *arXiv preprint arXiv:2604.02431*, 2026.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E Gonzalez, Mohammed Kadous, and Ion Stoica. Routellm: Learning to route llms from preference data. In *International Conference on Learning Representations*, volume 2025, pp. 34433–34448, 2025.

Charles Packer, Vivian Fang, Shishir_G Patil, Kevin Lin, Sarah Wooders, and Joseph_E Gonzalez. Memgpt: towards llms as operating systems. 2023.

Zhuoshi Pan, Qianhui Wu, Huiqiang Jiang, Xufang Luo, Hao Cheng, Dongsheng Li, Yuqing Yang, Chin-Yew Lin, H Vicky Zhao, Lili Qiu, et al. Secom: On memory construction and retrieval for personalized conversational agents. In *International Conference on Learning Representations*, volume 2025, pp. 91851–91885, 2025.

Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. Generative agents: Interactive simulacra of human behavior. In *Proceedings of the 36th annual acm symposium on user interface software and technology*, pp. 1–22, 2023.

Daivik Patel and Shrenik Patel. Engram: Effective, lightweight memory orchestration for conversational agents. *arXiv preprint arXiv:2511.12960*, 2025.

Preston Rasmussen, Pavlo Paliychuk, Travis Beauvais, Jack Ryan, and Daniel Chalef. Zep: a temporal knowledge graph architecture for agent memory. *arXiv preprint arXiv:2501.13956*, 2025.

Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. Ares: An automated evaluation framework for retrieval-augmented generation systems. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 338–354, 2024.

Rana Salama, Jason Cai, Michelle Yuan, Anna Currey, Monica Sunkara, Yi Zhang, and Yassine Benajiba. Meminsight: Autonomous memory augmentation for llm agents. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 33124–33140, 2025.

Haoran Tan, Zeyu Zhang, Chen Ma, Xu Chen, Quanyu Dai, and Zhenhua Dong. Membench: Towards more comprehensive evaluation on the memory of llm-based agents. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 19336–19352, 2025a.

Zhen Tan, Jun Yan, I-Hung Hsu, Rujun Han, Zifeng Wang, Long Le, Yiwen Song, Yanfei Chen, Hamid Palangi, George Lee, et al. In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8416–8439, 2025b.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025.

Qingyue Wang, Yanhe Fu, Yanan Cao, Shuai Wang, Zhiliang Tian, and Liang Ding. Recursively summarizing enables long-term dialogue memory in large language models. *Neurocomputing*, 639:130193, 2025.

Di Wu, Hongwei Wang, Wenhao Yu, Yuwei Zhang, Kai-Wei Chang, and Dong Yu. Longmemeval: Benchmarking chat assistants on long-term interactive memory. *arXiv preprint arXiv:2410.10813*, 2024.

Yaxiong Wu, Sheng Liang, Chen Zhang, Yichao Wang, Yongyue Zhang, Huifeng Guo, Ruiming Tang, and Yong Liu. From human memory to ai memory: A survey on memory mechanisms in the era of llms. *arXiv preprint arXiv:2504.15965*, 2025.

Zhaofen Wu, Hanrong Zhang, Fulin Lin, Wujiang Xu, Xinran Xu, Yankai Chen, Henry Peng Zou, Shaowen Chen, Weizhi Zhang, Xue Liu, et al. Gam: Hierarchical graph-based agentic memory for llm agents. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 34647–34664, 2026.

Derong Xu, Yi Wen, Pengyue Jia, Yingyi Zhang, Wenlin Zhang, Yichao Wang, Huifeng Guo, Ruiming Tang, Xiangyu Zhao, Enhong Chen, et al. From single to multi-granularity: Toward long-term memory association and selection of conversational agents. In *International Conference on Learning Representations*, volume 2026, pp. 91581–91617, 2026a.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents. *Advances in Neural Information Processing Systems*, 38:17577–17604, 2026b.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Chengyuan Yang, Zequn Sun, Wei Wei, and Wei Hu. Beyond static summarization: Proactive memory extraction for llm agents. *arXiv preprint arXiv:2601.04463*, 2026.

Bowen Ye, Yongchao Xu, Zhijian Li, Xiang Yin, Junkai Ma, and Wenzhao Li. Coevo-mem: Coevolving retrieval policy and memory bank for llm agents. *arXiv preprint arXiv:2608.01739*, 2026.

Boqin Yuan, Yue Su, and Kun Yao. Diagnosing retrieval vs. utilization bottlenecks in llm agent memory. *arXiv preprint arXiv:2603.02473*, 2026.

Yanwei Yue, Guibin Zhang, Boyang Liu, Guancheng Wan, Kun Wang, Dawei Cheng, and Yiyan Qi. Masrouter: Learning to route llms for multi-agent systems. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15549–15572, 2025.

Ruihong Zeng, Jinyuan Fang, Siwei Liu, and Zaiqiao Meng. On the structural memory of llm agents. *arXiv preprint arXiv:2412.15266*, 2024.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025a.

Yujie Zhang, Weikang Yuan, and Zhuoren Jiang. Bridging intuitive associations and deliberate recall: Empowering llm personal assistant with graph-structured long-term memory. In *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 17533–17547, 2025b.

Zeyu Zhang, Quanyu Dai, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. A survey on the memory mechanism of large language model-based agents. *ACM Transactions on Information Systems*, 43(6):1–47, 2025c.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pp. 19724–19731, 2024.

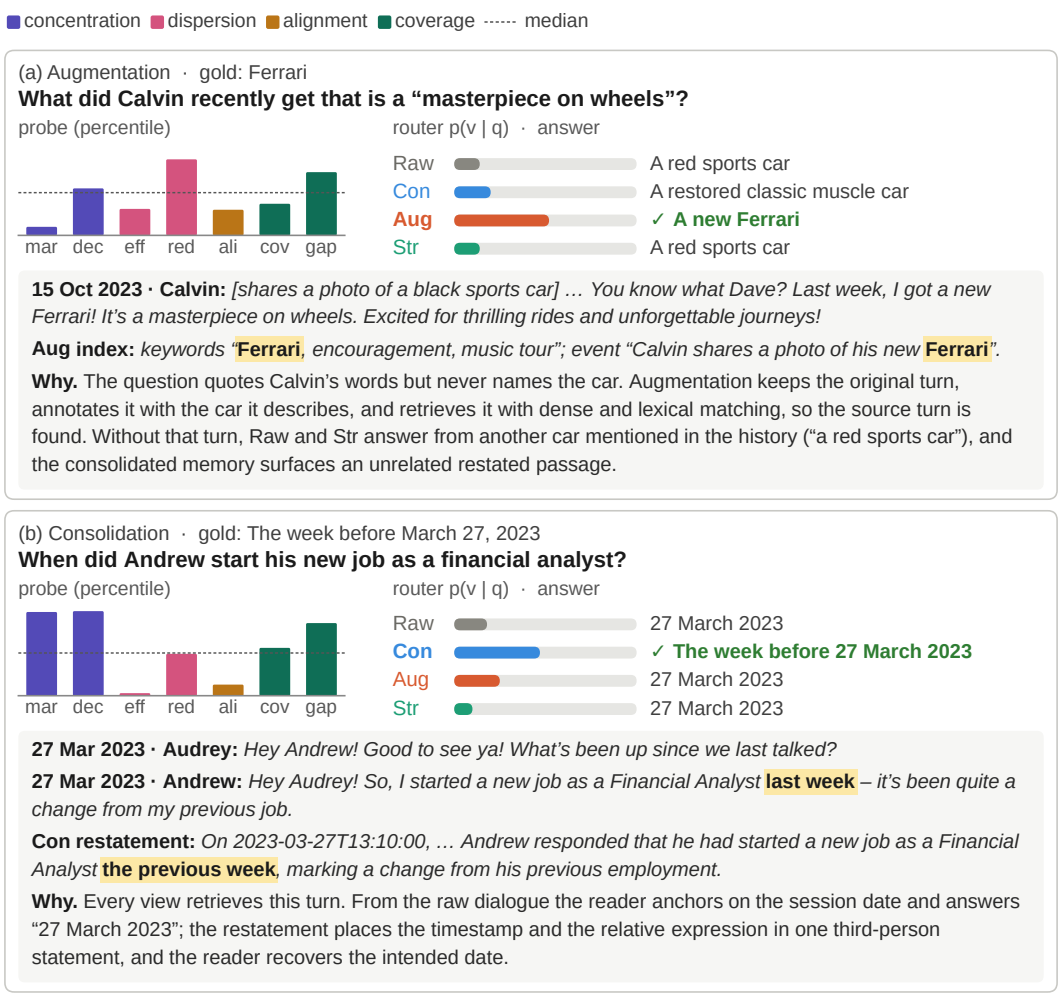



Figure 6: Cases in which MemRx selects the view that yields the correct answer. (a) Augmentation retrieves a turn that the query paraphrases but never names. (b) Consolidation resolves a relative temporal expression that every view retrieves but only the restatement makes explicit. Left: probe features as percentiles (dashed line: median). Right: $p_\theta(v \mid q)$ and the answer under each view.

## A Case Studies

Figures 6 and 7 show four LoCoMo test questions with Qwen3-8B as the reader. Each panel shows the seven probe features $\phi_q$ as percentiles (left; dashed line: median), the routing distribution $p_\theta(v \mid q)$ with the answer under each view (right; ✓ marks a match with the reference), and the evidence turns together with, where relevant, the entry in the winning view that exposes the answer (bottom).

**(a) Evidence present but hard to retrieve.** The query quotes Calvin (“masterpiece on wheels”) without naming the car. The probe shows a very low margin, high redundancy, and a large coverage gap: no retrieved chunk stands out, the chunks overlap heavily, and none accounts for the query alone. This is the pattern augmentation targets. The attribute annotations add “Ferrari” as a keyword and an event, so the hybrid retriever surfaces the source turn; without it, the reader answers with a different car mentioned elsewhere.

**(b) Evidence retrieved but misinterpreted.** Margin and decay are near the top of their distributions: the evidence sits in a single, easily retrieved turn, and every view retrieves it. The failure is one of interpretation: from the raw dialogue, the reader anchors on the session date and ignores “last week.” The consolidated restatement combines the timestamp and the relative expression in

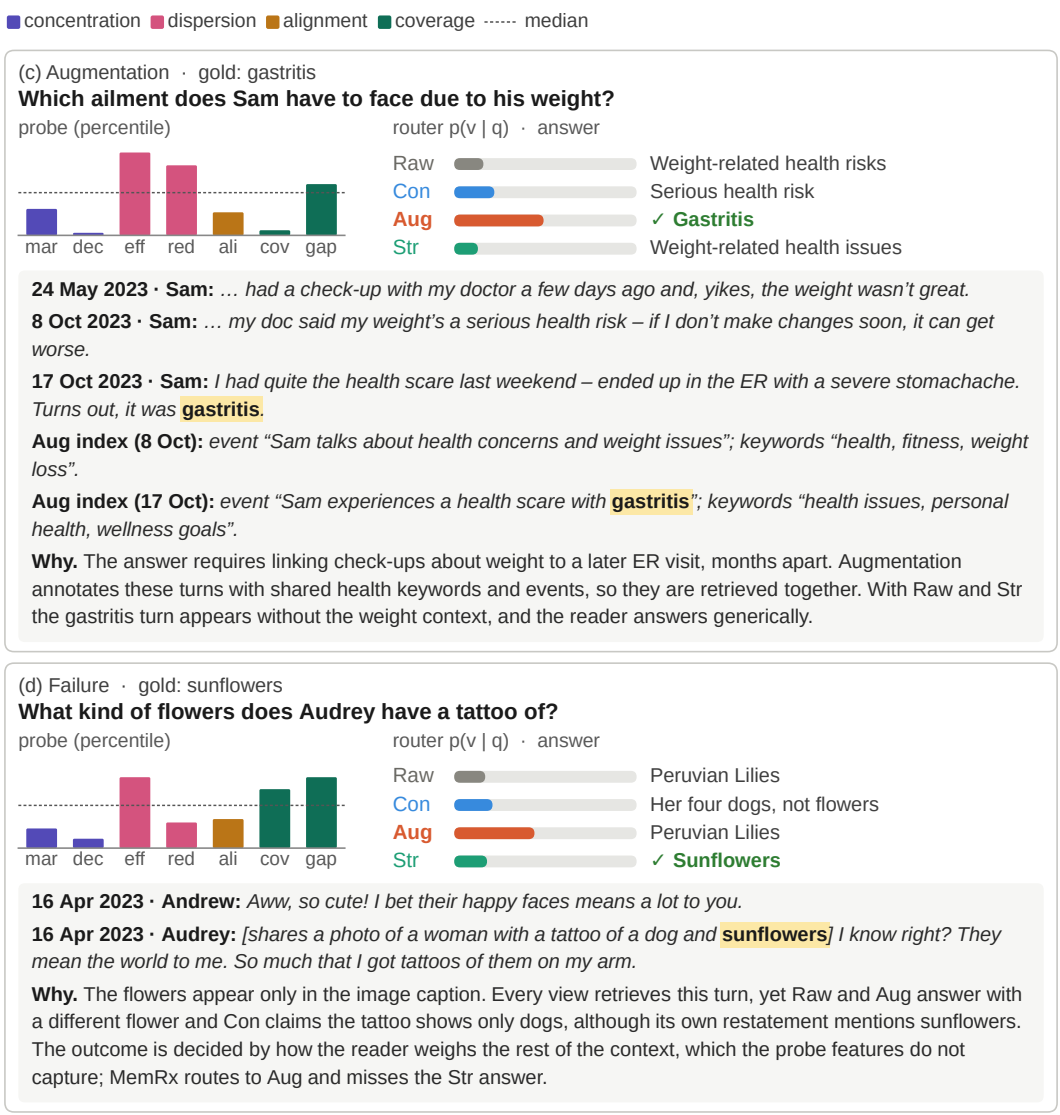



Figure 7: Further cases. (c) Augmentation links turns about the same health issue that occur months apart. (d) A failure: every view retrieves the evidence turn, the outcome depends on how the reader uses the remaining context, and MemRx selects augmentation although only structuring answers correctly. Layout as in Figure 6.

one third-person statement, from which the reader recovers the correct date. When retrieval is not the bottleneck, a view helps through how it represents the evidence, and MemRx places most of its mass on consolidation.

**(c) Evidence dispersed across sessions.** The answer requires linking check-ups on Sam's weight in May and October to an emergency visit for gastritis later in October. The probe shows a very high effective fraction and very low single-chunk coverage: retrieval mass is spread across many chunks, none of which covers the query. Augmentation annotates these turns with shared health-related events and keywords, so they are retrieved together; under the raw and structuring views, the gastritis turn appears without its weight context and the reader answers generically. Although dispersed evidence is the case structuring targets, shared annotations also act as implicit links, so the information needs served by the three families overlap.

**(d) Failure: the outcome is decided by the reader.** The answer ("sunflowers") appears only in an image caption, and every view retrieves that turn. Yet the raw and augmentation views yield a different flower, and consolidation claims the tattoo shows only dogs even though its own restatement mentions sunflowers. The probe features describe the shape of retrieval, not how the reader uses the context, so they cannot identify the view under which the reader succeeds. MemRx selects augmentation and assigns low probability to structuring, the only correct view, consistent with the routing analysis (Section 4.2): structuring wins are narrow and it is rarely selected. Adding reader-side signals to the routing state is a natural extension.

# B Preliminary Study Details

**Setup.** The preliminary study uses the validation conversation of LoCoMo with Qwen3-8B as the reader. All seven views in $\mathcal{V}^+$ share the setup in Appendix C.1 and are constructed as in Appendices C.2–C.4. Win, tie, and loss rates (Figure 1a) use token-level F1; the pairwise comparison (Figure 1b) reports F1 and BLEU-1. The oracle selects, per query, the view in $\mathcal{V}^+$ with the highest F1 and thus upper-bounds any router over this set.

**Router baselines.** *Random* selects a view uniformly from $\mathcal{V}^+$. *Judge* gives DeepSeek-V3.1 the query and a one-sentence description of each view and asks for the most suitable view. *ICL* adds to the Judge prompt 8 training queries, each labeled with its best view. *Clf.* is a two-layer MLP over Qwen3-Embedding-0.6B query embeddings with one sigmoid output per view, trained with binary cross-entropy on per-view correctness labels from the training split; at test time it selects the view with the highest predicted probability of being correct, breaking ties in favor of raw memory.

# C Memory Views

This appendix specifies how each view is constructed and retrieved; all prompts are listed in Appendix C.5. In the prompt listings, ⟨*name*⟩ marks a slot filled at run time, and all other text, including JSON braces, is passed verbatim.

## C.1 Shared Setup

**Raw chunks.** The interaction history is segmented into windows of five dialogue turns with a one-turn overlap. Each turn is rendered as `[`⟨*timestamp*⟩`]` ⟨*speaker*⟩`:` ⟨*utterance*⟩, and turns are joined by newlines. Each chunk records its source turn range and the date of its first turn; derived entries inherit this metadata, so every entry in every view traces back to the same source turns. All transformations make one construction call per chunk.

**Models.** Memory construction uses Qwen3-8B with thinking mode disabled, served through an OpenAI-compatible endpoint. The reader is Qwen3-8B or Gemma3-4B (Section 4). Qwen3-Embedding-0.6B encodes entries and queries, and dense retrieval ranks entries by cosine similarity to the query.

**Retrieval and answer generation.** Each view retrieves at most $K = 20$ entries, which are passed to the reader in retrieval order, the $i$-th rendered as `[Context` $i$`]` followed by `Content:` ⟨*text*⟩. All views share the reader prompt (adapted from SimpleMem), sampled at temperature 0.1 with at most 1,024 output tokens. The `answer` field of the returned JSON is the prediction; if no parseable JSON is obtained within three attempts, the raw reply is used. For questions without a reference answer (LoCoMo category 5), the reference is set to "Not mentioned in the conversation."

## C.2 Consolidation

Consolidation replaces raw text with LLM-generated restatements, which serve as both the retrieval index and the reader context. Both variants require complete coverage, full names instead of pronouns, and absolute ISO 8601 timestamps instead of relative expressions; these are generation requirements, not guarantees of lossless compression. Construction uses temperature 0.1 and up to

three attempts per chunk, succeeding if at least one non-empty restatement is returned; otherwise the raw chunk is kept as a single entry, so no source material is dropped.

- **Single-restatement (S-Str).** Each chunk maps to one dense, self-contained restatement, so the view has as many entries as raw memory. This is the consolidation representative in $\mathcal{V}$.
- **Fine-grained (S-FG).** Each chunk maps to one to three restatements; a chunk covering distinct topics may be split, provided the restatements jointly cover it. After deduplication, each restatement is indexed as a separate entry.

### C.3 AUGMENTATION

Augmentation keeps one entry per chunk and appends LLM-generated annotations, one line per non-empty field (e.g. `Entities: a; b; c`). The annotated text is used only for indexing; the reader sees the original chunk, so annotations change what is retrieved but not what is read. Both prompts receive the chunk's conversation date and forbid unstated facts. Construction uses temperature 0 and up to two attempts, succeeding if a JSON object with at least one non-empty field is returned; otherwise the entry is left unannotated.

**Attribute-based (A-Att).** The model extracts entities (people, places, organizations, objects, titles), events (what happened and who was involved), temporal references (rewritten as absolute dates relative to the conversation date), and keywords (topical terms and synonyms that need not appear in the chunk). Retrieval fuses dense and BM25 rankings over the annotated text by weighted reciprocal-rank fusion,

$$s_{\text{attr}}(m, q) = \frac{0.7}{10 + r_{\text{dense}}(m, q)} + \frac{0.3}{10 + r_{\text{BM25}}(m, q)}, \tag{15}$$

with one-based ranks; an entry absent from a ranking receives no contribution from it. Each retriever contributes up to $\min\{N_v, \max(4K, 10)\}$ candidates, where $N_v$ is the number of entries, and the top $K$ under the fused score are returned. This is the augmentation representative in $\mathcal{V}$.

**Note-based (A-Not).** Following A-MEM, the model generates salient keywords, a one-sentence context, and categorical tags. The note is embedded jointly with the chunk, and retrieval is dense only.

### C.4 STRUCTURING

Structuring keeps the raw chunks and their embeddings and adds an undirected graph over chunks that is used only to rerank dense retrieval. The two variants differ in their edges.

**Edges.** *Entity edges* link two chunks that share a normalized entity (lowercased, non-alphanumerics replaced by spaces, whitespace collapsed). Entities are extracted per chunk (temperature 0, one attempt; a failed extraction yields none). To suppress hubs such as speaker names, an entity induces edges only if it occurs in between 2 and $\max\{2, \lfloor 0.3n \rfloor\}$ of the $n$ chunks. *Semantic edges* link each chunk to its three nearest neighbors by embedding cosine similarity, symmetrized so an edge exists whenever either chunk selects the other. The hybrid variant (G-Ent) uses both edge types and is the structuring representative in $\mathcal{V}$; the semantic variant (G-Sem) uses semantic edges only.

**One-hop propagation.** For query $q$, let $a_i$ be the cosine similarity between $q$ and chunk $m_i$, and let $\mathcal{S}_q$ be the ten highest-scoring chunks. Chunks are ranked by

$$s_{\text{graph}}(m_j, q) = a_j + \gamma \max(\{0\} \cup \{a_i : i \in \mathcal{S}_q,\ (i, j) \in \mathcal{E}\}), \qquad \gamma = 0.2. \tag{16}$$

Propagation is a single, non-iterated hop; chunks without an adjacent seed receive no bonus, and adjacent seeds may boost one another. Setting $\gamma = 0$ recovers dense retrieval exactly. The top $K$ chunks are passed to the reader as original text.

### C.5 PROMPTS

**Reader prompt (all views)**

```
[System] You are a professional Q&A assistant. Extract concise answers from context. You
     must output valid JSON format.

[User]
Answer the user's question based on the provided context.

User Question: ⟨question⟩

Relevant Context:
⟨assembled context⟩

Requirements:
1. First, think through the reasoning process
2. Then provide a very CONCISE answer (short phrase about core information)
3. Answer must be based ONLY on the provided context
4. All dates in the response must be formatted as 'DD Month YYYY' but you can output more
     or less details if needed
5. Return your response in JSON format

Output Format:
```json
{
  "reasoning": "Brief explanation of your thought process",
  "answer": "Concise answer in a short phrase"
}
```

Example:
Question: "When will they meet?"
Context: "Alice suggested meeting Bob at 2025-11-16T14:00:00..."

Output:
```json
{
  "reasoning": "The context explicitly states the meeting time as 2025-11-16T14:00:00",
  "answer": "16 November 2025 at 2:00 PM"
}
```

Now answer the question. Return ONLY the JSON, no other text.
```

**Consolidation: single-restatement (S-Str)**

```
[System] You are a professional information extraction assistant. Extract structured,
     unambiguous facts from conversations. Output valid JSON only.

[User]
Your task is to compress the following dialogues into a SINGLE memory entry that preserves
     every piece of information necessary to answer downstream questions.

[Current Window Dialogues]
⟨raw chunk⟩

[Requirements]
1. Must write timestamps to the lossless restatement with format YYYY-MM-DDTHH:MM:SS.
2. **Complete Coverage**:  Ensure ALL information in the dialogues is captured.
3. **Force Disambiguation**: Absolutely PROHIBIT using pronouns (he, she, it, they, this,
     that) and relative time (yesterday, today, last week, tomorrow). Use full names and
     absolute ISO 8601 timestamps inline.

[Output Format]
Return a JSON array with EXACTLY ONE element:

```json
[
  {
    "lossless_restatement": "Complete, dense, unambiguous restatement that includes every
     fact and time stamp from the window"
  }
]
```
```

```
Now process the above dialogues. Return ONLY the JSON array (length 1), no other
    explanations.
```

**Consolidation: fine-grained (S-FG; same system prompt, differs in requirement 4 and output length)**

```
Your task is to compress the following dialogues into memory entries that preserve every
    piece of information necessary to answer downstream questions.

[Current Window Dialogues]
⟨raw chunk⟩

[Requirements]
1. Must write timestamps to the lossless restatement with format YYYY-MM-DDTHH:MM:SS.
2. **Complete Coverage**:  Ensure ALL information in the dialogues is captured.
3. **Force Disambiguation**: Absolutely PROHIBIT using pronouns (he, she, it, they, this,
    that) and relative time (yesterday, today, last week, tomorrow). Use full names and
    absolute ISO 8601 timestamps inline.
4. **Splitting**: If the dialogues cover several distinct topics or events, you may split
    them into 2-3 entries, each self-contained and covering one topic. Together the
    entries must still cover ALL information.

[Output Format]
Return a JSON array with 1-3 elements:

'''json
[
  {
    "lossless_restatement": "Complete, dense, unambiguous restatement of one part of the
    window, including every relevant fact and time stamp"
  }
]
'''

Now process the above dialogues. Return ONLY the JSON array (length 1-3), no other
    explanations.
```

**Augmentation: attribute-based (A-Att)**

```
Annotate this conversation excerpt with attributes that would help find it again later.

Conversation date: ⟨date⟩

Excerpt:
⟨raw chunk⟩

Return a JSON object with exactly these fields (use [] when nothing applies):
{"entities": [], "events": [], "time": [], "keywords": []}

- entities: people, places, organizations, objects, titles mentioned.
- events: short phrases for what happened, with who (e.g. "Alice adopted a dog").
- time: time references rewritten as absolute dates using the conversation date (e.g.
    "last Saturday" -> "the Saturday before 9 June 2023").
- keywords: 3-8 topic words or synonyms someone might use to ask about this excerpt,
    including words that do not appear in it.

Do not add facts that are not stated. Return ONLY the JSON object.
```

**Augmentation: note-based (A-Not)**

```
Generate a structured analysis of the following conversation excerpt by:
1. Identifying the most salient keywords (focus on nouns, verbs and key concepts).
2. Writing one sentence of context: the main topic, the key points, and who is involved.
3. Creating relevant categorical tags (broad categories or themes, for classification).

Conversation date: ⟨date⟩

Excerpt:
⟨raw chunk⟩

Return a JSON object:
{"keywords": ["...", "..."], "context": "one sentence", "tags": ["...", "..."]}
```

```
Keep keywords ordered from most to least important and do not add facts that are not
    stated. Return ONLY the JSON object.
```

**Structuring: entity extraction (G-Ent only)**

```
Extract all named entities (people, places, organizations, specific objects/events) from
    this dialogue excerpt.

Dialogue:
⟨raw chunk⟩

Return a JSON array of entity name strings (deduplicated, singular canonical form). Return
    ONLY the JSON array.
```

## D Implementation Details of MemRx

**Training.** Table 5 lists the hyperparameters. All training runs on a single NVIDIA B200 GPU.

Table 5: Hyperparameters of MemRx.

| Component | Hyperparameter | Value |
|---|---|---|
| Probe | Probe depth $K_p$ | 20 |
| Routing state | Projection dim. of $\Pi_q, \Pi_p$ | 32 |
| Scorer | Hidden dim. of $W_x, W_d$ | 32 |
| | Layers of $g$ | 3 |
| Supervision | F1-spread threshold $\delta$ | 0.2 |
| | Judge weight $\lambda$ | 0.5 |
| Optimization | Optimizer | AdamW |
| | Learning rate | $3 \times 10^{-3}$ |
| | Weight decay | $10^{-4}$ |
| | Epochs | 50 |

**View descriptions.** The scorer (Equation 9) represents each view $v \in \mathcal{V}$ by the Qwen3-Embedding-0.6B embedding $e_{d_v}$ of a short description $d_v$, listed verbatim in Table 6.

Table 6: View descriptions $d_v$ used by the scorer.

| View | Description $d_v$ |
|---|---|
| Raw | Raw dialogue chunks kept verbatim, retrieved by dense similarity to the question. |
| Consolidate | Each chunk is compressed by an LLM into one dense, self-contained restatement with full names and absolute timestamps instead of pronouns and relative time. Compact; drops the original wording. |
| Augment | Raw chunks kept verbatim, with LLM-generated entities, events, absolute dates and topic keywords appended for dense and keyword retrieval. Good when the question uses different words than the conversation or asks about a date. |
| Structure | Raw chunks linked to chunks that share a rare entity and to their most similar chunks; the best semantic matches pass relevance to their linked chunks. Good for questions whose evidence is spread over related parts of the conversation. |

**Judges.** Both judges use DeepSeek-V3.1 in non-thinking mode, return a binary verdict, and are not told which view produced their input; the judge score is the probability of the affirmative token. The evidence judge receives $(q, y, \mathcal{C}_q^{(v)})$ and the answer judge receives $(q, y, \hat{y}_q^{(v)})$.

**System prompt (both judges)**

```
\begin{lstlisting}[style=prompt]
```

```
You are a meticulous evaluator for a long-term conversational memory benchmark. You apply
     the grading rules you are given exactly, never use outside knowledge, and never
     explain yourself. Your reply always begins with a single verdict character: Y or N.
\end{lstlisting}
```

**Evidence judge**

```
\begin{lstlisting}[style=prompt]
You are auditing a memory retrieval system. A user asked a question about a long
     conversation between two people, and the system retrieved the context below from its
     memory. Decide whether the retrieved context, on its own, contains the evidence
     needed to arrive at the reference answer.

<question>
{question}
</question>

<reference_answer>
{gold}
</reference_answer>

<retrieved_context>
{ctx}
</retrieved_context>

Judging rules:
1. Sufficiency, not relevance. Answer Y only if a careful reader who sees nothing but this
     context could derive the reference answer. Mentioning the right people, topic, or
     event without the decisive detail is N.
2. Every part is required. If the answer combines several facts (several items, two
     events, a person and a date), every essential part must be supported. The parts may
     come from different context blocks. If any essential part is missing, answer N.
3. Dates and times. A date is supported if it is stated explicitly, or if it can be
     computed from a relative expression ("last Saturday", "two weeks ago") together with
     a timestamp shown in the context. A relative expression with no timestamp to anchor
     it is not enough.
4. Inference. Support may be indirect when the answer follows from what is stated through
     a short commonsense step (e.g., "she adopted a puppy" supports "she has a dog").
     Answers that require outside knowledge, speculation, or unstated facts are not
     supported.
5. Unanswerable questions. If the reference answer says the information is not mentioned
     or not available, answer Y when the context contains nothing that would lead a reader
     to a specific answer, and N when it contains a statement that appears to answer the
     question.
6. Conflicts. If the context gives conflicting statements about the needed fact and does
     not make clear which one holds, answer N.
7. Ignore noise and form. Irrelevant, redundant, or reworded material does not make
     sufficient evidence insufficient. The number, order, and style of context blocks do
     not matter.

Reply with exactly one character: Y if the evidence is sufficient, N otherwise.
\end{lstlisting}
```

**Answer judge**

```
\begin{lstlisting}[style=prompt]
You are grading an answer produced by a question-answering system against a reference
     answer. The question is about a long conversation between two people. Decide whether
     the produced answer conveys the same information as the reference answer.

<question>
{question}
</question>

<reference_answer>
{gold}
</reference_answer>

<produced_answer>
{ans}
</produced_answer>

Grading rules:
```

```
1. Meaning, not wording. Ignore phrasing, word order, capitalization, punctuation, length,
     and whether the answer is a sentence or a short phrase. Synonyms and paraphrases
     match.
2. Dates and times. Different formats of the same date match ("7 May 2023", "2023-05-07",
     "May 7th, 2023"). A relative description matches only if it clearly denotes the same
     time as the reference. An answer coarser than the reference does not match ("2023"
     does not match "7 May 2023"); a finer answer consistent with a coarser reference does
     match.
3. Essential content. The produced answer must contain every essential element of the
     reference: for lists, every listed item (in any order); for who/what/when/where
     questions, the key entity, value, or time. Extra details are acceptable as long as
     they do not contradict the reference.
4. No hedging. If the produced answer lists several alternatives or stays noncommittal
     instead of committing to the reference answer, answer N.
5. Refusals. "Not mentioned", "unknown", or similar matches only when the reference also
     says the information is not available. When the reference says the information is not
     available, any specific answer is N.
6. Agreement only. Judge only whether the produced answer agrees with the reference. Do
     not use outside knowledge to decide whether either answer is true.

Reply with exactly one character: Y if the answers match, N otherwise.
\end{lstlisting}
```

## E Dataset Details

**LoCoMo.** LoCoMo (Maharana et al., 2024) contains 10 long multi-session dialogues between two speakers. Following prior work (Chhikara et al., 2025; Liu et al., 2026), we evaluate on the multi-hop, temporal, open-domain, and single-hop categories and exclude the 446 adversarial questions (category 5), whose answers are absent by construction.

**PerLTQA.** PerLTQA (Du et al., 2024) (English release, `en_v2`) provides personal memories for 32 characters, each comprising profile attributes, social relationships, narrative events, and timestamped dialogues grounded in those events; questions are labeled by the memory type of their evidence, and we report results by type. To reuse the LoCoMo pipeline unchanged, each character is converted into one chronological history: profile attributes, relationships, and events become narrator-style entries attributed to the protagonist, while dialogues keep their speakers and timestamps. Because building all views and judge labels is costly, we use 10 characters with memory sizes comparable to a LoCoMo dialogue: after removing Chen Zhi, whose memory is a single 10-utterance dialogue, we keep the 10 characters with the fewest questions.

**Splits.** We split by conversation (LoCoMo) or character (PerLTQA) at a 2:1:7 train/val/test ratio, so the router never sees a test memory during training. For LoCoMo, `conv-26` and `conv-30` are used for training, `conv-41` for validation, and the other seven for testing. Tables 7 and 8 summarize both datasets and the PerLTQA split.

Table 7: Dataset statistics by split. Words are whitespace-delimited tokens of the memory content.

| | | Train | Val | Test | Total |
|---|---|---|---|---|---|
| **LoCoMo** | #Conversations | 2 | 1 | 7 | 10 |
| | #Sessions | 38 | 32 | 202 | 272 |
| | #Utterances | 788 | 663 | 4,431 | 5,882 |
| | #Words | 18,447 | 16,165 | 99,160 | 133,772 |
| | #Questions | 233 | 152 | 1,155 | 1,540 |
| | Multi-hop | 43 | 31 | 208 | 282 |
| | Temporal | 63 | 27 | 231 | 321 |
| | Open-domain | 13 | 8 | 75 | 96 |
| | Single-hop | 114 | 86 | 641 | 841 |
| **PerLTQA** | #Characters | 2 | 1 | 7 | 10 |
| | #Profile attributes | 30 | 14 | 100 | 144 |
| | #Relationships | 15 | 9 | 64 | 88 |
| | #Events | 39 | 25 | 167 | 231 |
| | #Dialogue sessions | 38 | 24 | 168 | 230 |
| | #Utterances | 578 | 373 | 2,596 | 3,547 |
| | #Words | 18,265 | 11,414 | 78,672 | 108,351 |
| | #Questions | 421 | 224 | 1,552 | 2,197 |
| | Profile | 16 | 12 | 86 | 114 |
| | Social Rel. | 43 | 17 | 129 | 189 |
| | Events | 229 | 98 | 672 | 999 |
| | Dialogues | 133 | 97 | 665 | 895 |

Table 8: PerLTQA characters used in our experiments. Ev.: events; Sess.: dialogue sessions; Utt.: utterances. Question columns: Pro. (profile), Soc. (social relationship), Evt. (event), Dlg. (dialogue).

| Split | Character | Ev. | Sess. | Utt. | Words | Pro. | Soc. | Evt. | Dlg. | All |
|---|---|---|---|---|---|---|---|---|---|---|
| Train | Kong Tingting | 16 | 18 | 255 | 8,536 | 3 | 23 | 133 | 57 | 216 |
| | Xu Jia | 23 | 20 | 323 | 9,729 | 13 | 20 | 96 | 76 | 205 |
| Val | Li Hua | 25 | 24 | 373 | 11,414 | 12 | 17 | 98 | 97 | 224 |
| Test | Zhang Xiaohong | 23 | 21 | 344 | 11,539 | 13 | 17 | 77 | 95 | 202 |
| | Sun Xiaoming | 23 | 23 | 359 | 11,232 | 13 | 21 | 91 | 85 | 210 |
| | Zhou Ting | 23 | 23 | 373 | 10,350 | 14 | 15 | 104 | 81 | 214 |
| | Wang Ming | 23 | 27 | 361 | 10,321 | 13 | 13 | 108 | 85 | 219 |
| | Liu Liang | 23 | 21 | 354 | 11,024 | 11 | 18 | 103 | 99 | 231 |
| | Yang Wei | 25 | 24 | 367 | 11,046 | 11 | 22 | 99 | 105 | 237 |
| | Zhao Li | 27 | 29 | 438 | 13,160 | 11 | 23 | 90 | 115 | 239 |