Bridging Reflective and Semantic Memory for Lifelong Learning in LLM-based Agents

Rodrigo G. Flexa[1[0009−0006−7591−7560]], Aissa H. Mohamed[1[0000−0001−7354−7240]], Frances A. Santos[1[0000−0002−0110−6507]], Leandro A. Villas[1[0000−0002−3372−3366]], and Julio Cesar dos Reis[1[0000−0002−9545−2098]]

University of Campinas, SP, Brazil
{r299214, a265189}@dac.unicamp.br, {frsantos, villas, jreis}@ic.unicamp.br

Abstract. Large Language Model (LLM) agents increasingly rely on external memory to support long-term reasoning, adaptation, and generalization. While prior work has explored reflective memory for self-evaluation and semantic memory for storing abstract knowledge, the interaction between them remains underexplored. This article introduces a unified memory architecture in which reflection and semantic memory co-evolve to support lifelong learning. Reflection is used to evaluate and refine semantic memory through selective strengthening, stabilization, refinement, and forgetting, while semantic memory enriches the reflection process by grounding self-critique in accumulated knowledge. This bidirectional coupling enables more effective use of past experience and improves downstream decision-making. Experiments on the ARC Challenge dataset with the Phi-2 model show consistent improvements over baselines using isolated or loosely coupled memory mechanisms. Overall, our proposed approach advances the design of adaptive LLM-based agents capable of robust, long-horizon reasoning through tightly integrated, heterogeneous memory systems.

Keywords: LLM Agent · Memory-Augmented LLMs · Bidirectional Memory Interaction · Memory Pruning and Refinement

1 Introduction

Large Language Model (LLM) agents have transformed human–computer interaction, enabling conversational systems for information retrieval, explanation, and decision support. Many of these systems rely on semantic memory, typically implemented through Retrieval-Augmented Generation (RAG) [5, 15], to provide curated external knowledge that enhances in-context reasoning.

Tasks requiring long-term coherence, adaptation, and sustained reasoning remain challenging because LLM-based agents are inherently stateless. To address this limitation, recent studies have introduced memory-augmented agents that incorporate explicit memory modules (*e.g.*, episodic or reflective stores) to retain and reuse past experiences [11, 13, 9]. While these approaches improve adaptation, they often treat memory components independently.

Most reflection mechanisms rely only on short-term trajectories or local feedback [7, 8, 12], which can lead to shallow, inconsistent, or non-transferable insights. At the same time, semantic memory may contain redundant, outdated, or conflicting knowledge, which can degrade effectiveness over time if not properly curated. These limitations highlight a gap in the connection between semantic and reflective memory in LLM-based agents.

This study specifically focuses on the interaction between semantic memory (structured, generalized knowledge) and reflective memory (self-critique and reasoning feedback). Reflection mechanisms help agents evaluate past decisions and refine behavior, while semantic memory captures reusable abstractions from experience. To the best of our knowledge, existing approaches rarely explore how these two memory types should explicitly inform and reinforce one another in the context of LLM-based agents.

Our solution proposes a unified architecture in which semantic and reflective memory interact bidirectionally. In our approach, semantic memory grounds and enriches reflection, while reflection evaluates and refines semantic knowledge through pruning and rewriting. This synergy enables the agents to exhibit more coherent, adaptive, and self-improving behavior over time.

We experimentally evaluate the effectiveness of the proposed solution on the ARC (AI2 Reasoning Challenge) dataset using Microsoft's Phi-2 as the core small language model (SLM) within the agent. Our study includes ablation analyses of semantic and reflective memory components, assessing their individual and combined impact on reasoning performance across the official training, validation, and test splits. Results show that each memory component yields measurable gains over the baseline, and their bidirectional integration yields the best effectiveness, improving accuracy by 7.16 percentage points over a no-memory baseline and by 3.84 percentage points over static semantic retrieval alone.

The remainder of this article is organized as follows. Section 2 reviews related work on memory-augmented LLM agents and reflection-based learning. Section 3 defines our unified memory architecture. Section 4 presents the methodology for the experimental evaluation. Section 5 reports our results, and Section 6 discusses our key findings. Section 7 presents the concluding remarks.

2 Related Work

Early studies on reflection-based agents showed that explicit self-critique improves reliability. Shinn *et al.* [14] introduced Reflexion, which converts environmental feedback into verbal reinforcement signals. Building on this paradigm, Zhao *et al.* [20] proposed ExpeL, which distills Reflexion experiences into reusable insights without parameter updates. Renze and Guven [12] characterized how self-critique affects problem-solving, and Wu *et al.* [18] proposed Editable-LLM to correct flawed reasoning. However, these approaches rely solely on local trajectories, without persistent knowledge to ground the reflection process.

A growing body of studies argues that introspection alone is insufficient and that structured memory is essential for long-term agent behavior. Tan *et al.*

[16] proposed Reflective Memory Management (RMM) to build personalized memories by combining prospective summaries with retrospective reflections, which improves retrieval and long-context dialogue performance. Liang *et al.* introduced SAGE [7] to integrate iterative reflection with memory compression guided by the Ebbinghaus forgetting curve, demonstrating that the long-term organization of information enhances reflective reasoning.

Park *et al.* [10] showed that natural-language memory streams can be periodically synthesized into reflections that steer agent behavior, but they do not explicitly incorporate structured semantic knowledge. MemInsight [13] proposed an autonomous memory-augmentation framework where agents self-curate, evaluate, and manage memory entries. By learning when to write, update, or prune memories, MemInsight demonstrated that adaptive memory policies can substantially improve multi-step tasks.

From a complementary angle, Pink *et al.* [11] argued that episodic memory is essential for long-term LLM agents. They emphasized that agents need temporally organized, revisitable experiences to support planning and robust adaptation. Similarly, Liu *et al.* introduced Echo [9], a temporal episodic memory module to enable dynamic recall based on event timing and relevance, producing more coherent long-horizon reasoning than standard context-based methods. We understand these episodic-centric approaches as complementary to our proposal, targeting the interaction between consolidated semantic knowledge and reflective traces that are themselves anchored in specific feedback events.

Recent research on reflection has focused on improving critique. Lippmann *et al.* [8] introduced Sweet&Sour, which balances positive and negative feedback using separate buffers. Li *et al.* [6] and Zhang *et al.* [19] enhanced critique quality through multi-stage synthesis and CoT-based aggregation. Hadj Mohamed *et al.* [3] presented an externalized reflection memory framework where an LLM generates structured feedback to improve SLM-based agent reasoning. However, these methods address only current questions and lack a persistent semantic memory to support critique.

To the best of our knowledge, no prior research has explored the active role of semantic memory in shaping the reflection process, nor has it examined how the outcomes of reflection can enhance and refine semantic memory. Our proposed architecture addresses this gap through a bidirectional interaction: semantic memory provides a foundation for reflection, facilitating deeper and more transferable critiques, while reflection consistently curates and updates semantic memory by enriching associated metadata.

3 A Unified Reflective-Semantic Memory Architecture

This section presents our proposal for integrating reflective and semantic memory within LLM-based agents to support lifelong learning and improve reasoning quality, adaptability, and behavioral consistency.

3.1 Architecture and Approach Overview

Figure 1 presents our unified memory architecture, tightly integrating the semantic and reflection memory components into the LLM agent's architecture. Our solution combines memory systems, enabling the LLM-based agent to access both its semantic knowledge and relevant reflections to generate more effective responses to future user prompts.

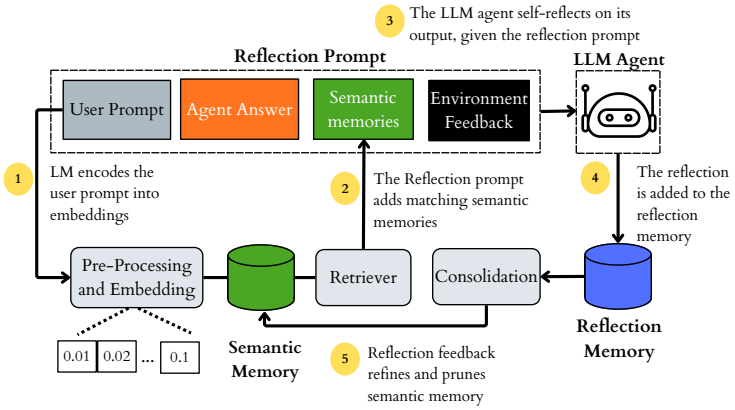

Fig. 1. Our unified memory architecture establishes a bidirectional interaction between reflection and semantic memories. These two memory components are constitutive parts of the LLM-based agent; the *LLM Agent* block denotes its core reasoning module.

The reflection memory is a mechanism that enables an LLM-based agent to evaluate its previous outputs and refine its future responses [14]. During that process, the agent identifies weaknesses in its reasoning, questions its assumptions, and improves its decision-making. More precisely, the mechanism prompts the agent to self-critique its past outputs based on environmental feedback [14]. The generated self-critique is stored for reuse in similar future tasks. Our reflection memory items take the form of textual annotations, explanations of reasoning flaws, or recommended strategies for approaching similar future tasks, also known as verbal reinforcement learning [14]. When retrieved for a new user prompt, such reflections help improve the agent's reasoning consistency and reduce the risk of error loops (the agent repeating the same errors).

The semantic memory stores distilled, structured knowledge extracted from prior experiences, such as conceptual summaries, inferred rules, recurring error patterns, and high-level strategies developed by the LLM-based agent for task completion.

We leverage the agent's reflective memory mechanism to support the ongoing consolidation and curation of semantic knowledge. The agent's reflective memory acts as a continuous engine for consolidating and curating its own semantic knowledge.

In our architecture, the retention or forgetting of information in semantic memory is modeled by a multidimensional system inspired by human cognition. Just as a student reviews content after a class, not only strengthening their memory but also judging its usefulness for an upcoming exam, our approach periodically evaluates each semantic memory item (*chunk*) by combining its relevance, temporal persistence, and empirical value during the reflection process. Subsection 3.2 presents the details of our solution, leveraging reflection to consolidate semantic knowledge.

Although effective in many situations, the reflection mechanism is usually isolated ([7, 8, 12]). When evaluated in isolation, LLM-based agents tend to focus on only the reasoning path that led to the incorrect output. As a result, the reflections can be too specific and tied to the immediate situation rather than applicable to other similar cases. Consequently, the absence of knowledge from semantic memory can lead to occasional repetitions of error patterns in reflection and to the failure to identify recurring reasoning mistakes.

These limitations motivate the need for an enriched reflective mechanism that leverages the knowledge accumulated in the semantic memory. To this end, our architecture incorporates relevant semantic memory during reflection, enabling the agent to produce deeper and more effective self-critiques. Given a broader knowledge context, the agent can more effectively evaluate its faulty reasoning, identify its flaws, and extract insights that remain valuable in both immediate and future-related situations. Subsection 3.3 presents how our solution explores semantic memory to strengthen reflection.

3.2 Leveraging reflection to consolidate semantic knowledge

In our architecture, each stored semantic memory item is periodically assessed through a reflective reasoning process that assigns a relevance score to determine whether the content should be retained, prioritized, or selectively forgotten.

As illustrated in Figure 1, after generating a self-critique and storing it in the reflection memory repository ($\mathcal{M}_R$) (Steps 3 and 4), the agent emits utility signals associated with the retrieved semantic memory items. The *Consolidation* module uses this feedback to refine and prune the semantic memory repository ($\mathcal{M}_S$) (Step 5), updating multidimensional metrics that capture semantic relevance, temporal stability, and practical utility. Formally, let $t \in \mathbb{N}$ denote the current interaction cycle, in which the agent processes a query q_t. The metric updates at cycle t trigger two complementary processes: *Dynamic Pruning* (DP), responsible for selective forgetting, and *Self-Refinement* (SR), which corrects or expands stored knowledge.

Stability. The first metric used by the *Consolidation* mechanism is *Stability* ($S_i(t)$) (cf. Equation 1), which captures the temporal dynamics of memory. To avoid repository saturation and the frequent reuse of privileged knowledge, inactive items decay exponentially over time, reducing their retention probability unless reinforced through reuse. Our Stability formulation adapts the two-component model of memory [17], in which retrievability decays exponentially, $R = \exp(-t/S)$, and stability S increases with repeated retrieval. This

model was empirically grounded in the classical forgetting curve of Ebbinghaus [2] and is computed in our architecture as:

$$S_i(t) = \exp\left(-\frac{t - \tau_i}{s_i}\right) \tag{1}$$

In Equation 1, τ_i denotes the *timestamp* of the last retrieval of memory item m_i, and $s_i = s_0 + \alpha n_i$ defines its intrinsic stability (s_0 represents the *base stability*, α the *learning rate*, and n_i the cumulative number of activations). Stability renders frequently accessed content structurally stable, allowing rarely accessed information to degrade naturally over time.

Affectivity. The frequent use of information does not guarantee that it is factually correct or beneficial for problem-solving. It may even reinforce repetitive behaviors that are not necessarily useful in the current context [4]. In this sense, we consider the quality and veracity of knowledge using the *Affectivity* metric ($A_i(t)$) (cf. Equation 2), which serves as a reflective *feedback loop* to quantify the historical utility of the memory item, adapting the verbal reinforcement learning principle introduced by [14]. After generating a response, the agent's reflection module evaluates *post-hoc* whether the retrieved context led to assertive reasoning or induced a hallucination. Based on this self-assessment, the agent updates the value of $A_i(t)$ for each memory item by applying a utility signal:

$$A_i(t+1) = A_i(t) + \lambda_i(t) \tag{2}$$

where $\lambda_i(t) \in [-1, 1]$ denotes the contribution score for item m_i at interaction t. The reflection module computes $\lambda_i(t)$ by inspecting the external feedback f together with the reasoning trace and judging whether m_i supported the correct answer ($\lambda_i(t) > 0$), misled the agent ($\lambda_i(t) < 0$), or had no clear effect ($\lambda_i(t) \approx 0$). This experiential learning mechanism enables the agent to reinforce reliable knowledge and downweight misleading or outdated information.

Relevancy. To ensure that the retrieved memory items are contextually aligned with the current user input, we define *Relevancy* ($R_i(t)$) as the cosine similarity between the embedding of the current query, $e(q_t)$, and the embedding of the memory item, $e(m_i)$.

Our architecture combines the *Stability*, *Affectivity*, and *Relevancy* metrics to retrieve the most appropriate context. Relevance acts as a multiplicative filter that ensures contextual alignment, since memories unrelated to the current query should not be retrieved, regardless of their *Stability* or *Affectivity*. The additive term then balances structural persistence (S_i) and empirical usefulness (A_i) through the factor β. We define the final retrieval *Score* as:

$$\text{Score}_i(t) = R_i(t) \cdot \left[\beta S_i(t) + (1-\beta) A_i(t)\right] \tag{3}$$

Algorithm 1 defines the semantic memory retrieval procedure (Figure 1, Step 2) by exploring the defined metrics. Given a *query* q_t, the agent computes its *embedding* and calculates the metrics $R_i(t)$, $S_i(t)$, and $A_i(t)$ for each memory item currently stored in the repository $\mathcal{M}_S$. The composite *Score* (line 7) determines

the *top-K* most relevant memory items $\mathcal{M}_K$, which the agent integrates with the *query* to form the context C_t. The agent then forwards this refined context to subsequent reasoning steps.

Algorithm 1 Retrieval and Context Construction

Require: *Query* q_t, Semantic memory repository $\mathcal{M}_S$, Final context size K
Ensure: Context C_t

```
Determine current cycle t
Calculate query embedding e(q_t)
for each m_i ∈ M_S do
    R_i(t) ← CosineSimilarity(e(q_t), e(m_i))
    S_i(t) ← Stability(m_i, t)
    A_i(t) ← Affectivity(m_i)
    Score_i(t) ← R_i(t)[βS_i(t) + (1 − β)A_i(t)]
end for
M_K ← TopK(M_S, Score_i(t), K)
C_t ← {M_K, q_t}
return C_t
```

Once the agent uses the context C_t to reason and generate its final output for the user's query q_t, the *Consolidation* module executes the process described in Algorithm 2. Initially, for each active memory item $m_i \in \mathcal{M}_K$, the module updates the *timestamp* τ_i, the activation count n_i, the stability parameter s_i, and the metric $A_i(t)$ based on the signal $\lambda_i(t)$ (lines 2–5).

Algorithm 2 Consolidation and Forgetting

Require: Active memories $\mathcal{M}_K$, *timestamp* t, utilities $\lambda_i(t)$
Ensure: Updated semantic memory repository $\mathcal{M}_S$

```
1  for each m_i ∈ M_K do
2      τ_i ← t                                   ▷ Update recency of use
3      n_i ← n_i + 1                             ▷ Increment activation count
4      s_i ← s_0 + αn_i                          ▷ Strengthen stability through repetition
5      A_i(t + 1) ← A_i(t) + λ_i(t)              ▷ Update Affectivity based on reflection
6  end for
7                                                ▷ Global evaluation for selective forgetting
8  for each m_i in M_S do
9      S_i(t) ← exp(−(t − τ_i)/s_i)              ▷ Recalculate decay
10     if S_i(t) < θ_S and A_i(t) < θ_A then
11         remove m_i                            ▷ Prune idle and useless information
12     end if
13 end for
14 return updated semantic memory repository M_S
```

Dynamic Pruning. The architecture performs a global sweep of $\mathcal{M}_S$ to recompute $S_i(t)$ based on elapsed time since last activation. The *Consolidation* module then removes memory items whose Stability and Affectivity fall below thresholds θ_S and θ_A (cf. line 10 in Algorithm 2), implementing *Dynamic Pruning*. This joint criterion avoids discarding structurally stable or still-useful knowledge, enabling selective forgetting.

Self-Refinement. Complementarily, the *Affectivity* metric drives *Self-Refinement*. When a memory item exhibits persistently negative utility, the reflection module revises its premise and stores a corrected version. Analysis of the reasoning chain may also yield generalizable principles absent from the repository. Both corrected items and newly derived knowledge are immediately incorporated and subjected to the same scoring, update, and decay mechanisms.

3.3 Leveraging semantic memory to strengthen reflection

Algorithm 3 formalizes how semantic memory is integrated into the reflection loop. The rationale is that reflection is not performed in isolation (i.e., it is not solely driven by the language model). Instead, it is grounded in previously consolidated knowledge from the semantic memory. This design transforms reflection from a reactive critique into a knowledge-aware self-evaluation process.

Algorithm 3 Reflection with Semantic Memory Context

Require: User prompt q_t, initial response r_0, feedback f, semantic memory repository $\mathcal{M}_S$, Final context size K

Ensure: Updated reflection memory repository $\mathcal{M}_R$

if f indicates positive, negative, or corrective signal **then**
  for each $m_i \in \mathcal{M}_S$ **do** ▷ Calculate similarity for each memory item
    compute similarity $\sigma_i \leftarrow \text{sim}(q_t, m_i)$
  end for
  $\mathcal{M}_K \leftarrow \text{Top-}K(m_i \mid \sigma_i)$ ▷ Select most relevant items for reflection
  $p_{ref} \leftarrow \text{Compose}(q_t, r_0, f, \mathcal{M}_K)$ ▷ Construct reflection prompt with context
  $r_{ref} \leftarrow \text{LLM}(p_{ref})$ ▷ Generate structured reflection
  $\mathcal{M}_R \leftarrow \mathcal{M}_R \cup \{r_{ref}\}$ ▷ Store reflection for future reuse
end if
return updated reflection memory repository $\mathcal{M}_R$

In Step 2 of the Figure 1, upon receiving positive, negative or corrective feedback from the environment regarding its initial response r_0, the LLM-based agent retrieves the K most relevant semantic memory items $\mathcal{M}_K$, from semantic memory repository $\mathcal{M}_S$, associated with the user prompt. This retrieval step relies on semantic similarity between the current user prompt and the stored memory items. Then, the retrieved semantic memory items are appended to the reflection prompt and submitted to the LLM-based agent for processing.

Formally, in Algorithm 3, the reflection prompt p_{ref} is constructed by composing four elements: (i) the original user prompt q_t; (ii) the agent's initial response

r_0; (iii) the external feedback signal f; and (iv) the retrieved Top-K semantic context $\mathcal{M}_K$. By explicitly presenting both the reasoning outcome and the relevant knowledge background, the LLM-based agent is encouraged to perform structured self-evaluation. Instead of merely stating that an answer is incorrect, the agent can identify why it is incorrect, which assumptions were flawed, which retrieved memory items were misapplied, and how the reasoning strategy should be adjusted.

Guided by this rich context, the LLM-based agent generates a reflection that critiques its response while situating the critique within relevant prior knowledge (Step 3 in Figure 1, and line 7 in Algorithm 3). This reflection includes error identification, causal analysis of the mistake, and a corrective guideline or revised reasoning strategy. In this sense, the reflection memory module functions as a meta-cognitive layer, transforming feedback into actionable knowledge.

Once the agent generates a reflection, it stores it in the reflection memory repository $\mathcal{M}_R$ for future reuse (Step 4 in Figure 1, and line 8 in Algorithm 3). These reflective memories are later retrieved in semantically similar situations, enabling the agent to generalize beyond the original failure case. Over time, this mechanism supports continual improvement without gradient updates, as the agent accumulates structured self-critiques that guide future reasoning.

By grounding reflection in semantic memory, our solution ensures that self-evaluation is both knowledge-aware and feedback-driven, thereby increasing the quality, transferability, and long-term utility of the agent's reflective memories.

4 Evaluation Methodology

This section presents the experimental methodology for evaluating our proposed approach.

4.1 Dataset and Language Models

We selected the ARC (AI2 Reasoning Challenge) dataset[1] as the benchmark for our experiments. It consists of multiple-choice questions that require factual recall, application of scientific knowledge, and logical inference. The "Challenge" contains questions that are particularly difficult for current LLMs due to their need for abstract reasoning, understanding causal relationships, and synthesizing multiple pieces of information. The dataset was introduced by Clark et al. [1] as a benchmark requiring reasoning beyond surface-level pattern matching. Small and mid-sized language models continue to struggle with the Challenge split, motivating their use in our evaluation. The code for all experiments is publicly available.[2]

[1] Challenge ARC Dataset, `https://huggingface.co/datasets/allenai/ai2_arc`. Last accessed: July 8, 2026.

[2] `https://github.com/RodrigoFlexa/Bridging-Reflective-and-Semantic`. Last accessed: July 8, 2026.

We chose Microsoft's Phi-2[3] as the Small Language Model (SLM) within the agent. It is a compact transformer-based model with 2.7 billion parameters, designed to excel in reasoning and mathematical tasks despite its relatively small size.

4.2 Experimental Procedures and Ablation Study Design

Table 1 summarizes the experimental settings and procedures to evaluate our solution against the baseline approaches.

Table 1. Summary of Experimental Design

Component	Description
Dataset	ARC Challenge with official train/valid/test splits. Train: create memories. Validation: update/prune. Test: retrieval only.
LLM	Phi-2 (2.7B). Temperature set to 0.0 for deterministic question-answering, and to 1.0 to explore all possibilities during reflection.
Baselines	(1) No memory, (2) Semantic only, (3) Reflection only, (4) Semantic+Reflection (concat).
Ablation	Static semantic retrieval without refinement or memory creation.
Retrieval	MiniLM embeddings + ChromaDB. Cosine top-K with K=3, $\tau_{sim} = 0.24$. Metacognitive re-ranking ($\beta = 0.7$).
Metrics	Accuracy, accuracy gain, error reduction, memory effect (useful/harmful), memory size.

Following the official split of the ARC dataset, we use the train set to build semantic and reflection memory items. Given an ARC question with answer options, the agent answers it and generates semantic memory items. These memory items are a mixture of accurate and incorrect (or noisy) content. Reflection memory items are generated based on the agent's initial answer to the ARC question, the feedback received (whether the selected answer is correct or not), and the attached semantic memories. These semantic memory items are also scored during the reflection phase. The reflection process does not modify the content of semantic memory items.

For the validation set, the objective is to refine and prune the semantic memory repository. The agent answers the ARC question with relevant semantic memory items from the train set, then self-reflects. The scores of the retrieved semantic memory items are updated accordingly. Finally, we discard from the repository the semantic memory items whose scores fall below the pruning thresholds.

We use the test set to evaluate the agent. The memories are kept frozen.

Table 2 defines eight configurations built upon the base language model to evaluate the effectiveness of our proposed memory architecture. The Baseline (No Memory) configuration serves as a reference condition, in which the model relies solely on its parametric knowledge, enabling the measurement of accuracy

[3] `https://huggingface.co/microsoft/phi-2`. Last accessed: July 8, 2026

gains attributable to memory augmentation. We then assess three static memory settings. In the Semantic configuration, the model is augmented with a static semantic memory that provides factual context without reflective feedback.

Table 2. Configurations evaluated in the ablation study.

Configuration	**Semantic Context**	**Reflection Context**	**Dynamic Pruning**	**Self-Refinement**
Baseline (No Memory)	–	–	–	–
Semantic	✓	–	–	–
Reflection	–	✓	–	–
Concatenated	✓	✓	–	–
Semantic-DP	✓	–	✓	–
Semantic-DP^{++}	✓	–	✓	✓
Concatenated-DP	✓	✓	✓	–
Concatenated-DP^{++}	✓	✓	✓	✓

In the Reflection configuration, the model receives only reflection-based context without semantic memory items. Finally, the Concatenated configuration combines semantic and reflective contexts within a single *prompt* via simple concatenation and without any interaction between the two memory components.

Semantic-DP incorporates a *Dynamic Pruning Mechanism*, enabling the multidimensional scoring function (Equation 3) and the selective forgetting procedure (Algorithm 2). Memory items are continuously evaluated via Stability ($S_i(t)$) and Affectivity ($A_i(t)$) and pruned when both drop below thresholds (θ_S, θ_A). Thresholds were tuned by grid search on the validation set over $\theta_S \in [0, 1]$ and $\theta_A \in [-1, 1]$ (step 0.1), yielding $\theta_S = 0.1$ and $\theta_A = -0.6$ based on validation accuracy.

Semantic-DP^{++} extends Semantic-DP by incorporating a *Self-Refinement Mechanism*. In this configuration, the empirical utility signal (λ_i) derived from the *feedback loop* not only penalizes low-quality memories but also triggers the agent to revise and correct imprecise memory items, progressively refining its semantic repository over time.

Finally, we evaluate fully hybrid configurations. Concatenated-DP combines dynamic semantic retrieval and pruning with static reflection context. The Concatenated-DP^{++} extends this design by integrating scoring, pruning, semantic refinement, and static reflection into a unified architecture, as shown in our proposed architecture. In all dynamic configurations (DP and DP^{++}), the reflection process continues to generate the utility signal λ_i from external feedback, which drives Affectivity updates and Self-Refinement. The *Reflection Context* column in Table 2 refers specifically to whether stored reflections are injected into the agent's response prompt, not to whether the reflection mechanism itself is active.

4.3 Evaluation Metrics

We assess the effectiveness of the proposed approach by considering the following metrics:

Accuracy score: We report standard multiple-choice accuracy as the main effectiveness metric. It is defined as the percentage of questions for which the correct answers were selected among the total number of questions.

Memory quality (useful vs. harmful): We analyze the quality of semantic memory. For that purpose, we use OpenAI's GPT-5 mini as the *LLM-as-a-Judge* to classify each semantic memory item as correct or harmful.

Memory size: We track the number of semantic memory items over time. This allows us to evaluate our solution's scalability and performance gains relative to memory size, distinguishing improvements driven by effective memory curation from those driven by repository growth.

5 Experimental Results

We report results across eight configurations evaluated along three dimensions: (i) task accuracy (cf. Subsection 5.1), (ii) similarity between retrieved memory items and the input query (cf. Subsection 5.2), and (iii) semantic memory statistics (cf. Subsection 5.3), including repository size and harmful content ratios.

5.1 Task Accuracy

In Table 3, starting from the baseline accuracy of 65.96%, the introduction of static memory (Semantic, Reflection, Concatenated) already leads to clear gains, reaching up to 70.99%. This indicates that even simple retrieval of past semantic or reflective information helps the agent better interpret and respond to the user prompt. These improvements suggest that relevant prior context is an important factor and that the model can effectively leverage stored information when it is made available.

Table 3. Accuracy across all experimental configurations. Results reported as percentages on validation and test sets. Bold indicates best effectiveness per dataset split.

Configuration	Valid. Acc. (%)	Test Acc. (%)
Baseline (No Memory)	69.57	65.96
Reflection	70.23	68.86
Semantic	70.90	69.28
Concatenated	72.91	70.99
Semantic-DP	73.58	70.05
Concatenated-DP	76.25	72.01
Semantic-DP^{++}	76.59	70.65
Concatenated-DP^{++}	**78.26**	**73.12**

When Dynamic Pruning is introduced (Semantic-DP and Concatenated-DP), effectiveness increases further to 72.01%. This step is particularly important because it shows that the benefit is not only due to having more memory, but to selecting the right memory items. By filtering out less useful or noisy memory items, the agent can focus on higher-quality knowledge, which leads to more accurate outputs. This shows that memory curation is a key component of the architecture, and that controlling the quality of retrieved memory items directly impacts the results.

The best result is obtained with the full hybrid configuration, Concatenated-DP^{++}, which reaches 73.12% accuracy. This result demonstrates that combining different types of memory contributes to an additional gain, indicating that integrating these memory types together with stronger filtering leads to more effective use of past experience. This corresponds to a 7.16 percentage point gain over the No Memory baseline and a 3.84 percentage point gain over the static Semantic configuration, which isolates the contribution of our bidirectional memory architecture beyond the use of semantic retrieval alone.

5.2 Effects of dynamic semantic retrieval

Table 4 reveals an important and somewhat counterintuitive pattern. In the dynamic configurations (Semantic-DP, Concatenated-DP, and their $^{++}$ variants), which rank memory items by the composite score of Equation 3 rather than by cosine similarity alone, the retrieved memory items exhibit lower average cosine similarity to the query than under static retrieval, yet these configurations consistently achieve higher accuracy (Table 3). For instance, although the Semantic configuration obtains the highest test similarity (0.5521), it is outperformed by Semantic-DP and Concatenated-DP^{++}, which retrieve memory items that are less similar in embedding space but more beneficial for the task. This suggests that relying solely on semantic proximity is not sufficient to identify the most useful memory items. Instead, the multidimensional scoring strategy favors memory items that are more informative, contextually appropriate, and aligned with prior successful reasoning, even if they are not the closest in vector space.

In addition, score-based retrieval substantially reduces the agent's exposure to harmful semantic memory items (i.e., memory items classified as incorrect). While the static Semantic configuration retrieves harmful memory items in 46.03% of cases at test time, this rate decreases to 35.17% with Semantic-DP and to 32.55% in the Concatenated-DP^{++} configuration (cf. Table 5). This reduction is significant because harmful or misleading memory items can negatively affect the agent's reasoning and lead to incorrect outputs. The pruning and refinement mechanisms therefore serve a dual role: they improve the overall quality of the semantic memory repository and actively reduce the likelihood of retrieving noisy information.

Taken together, the similarity patterns in Table 4 and the reduced harmful retrieval rates in Table 5 show that the proposed retrieval strategy improves not only the relevance of retrieved memory items but also their reliability. By favoring memory items that have demonstrated utility and filtering out those

associated with poor outcomes, the solution creates a more stable and effective semantic memory repository. This helps explain the consistent improvements in accuracy and supports the claim that score-based, feedback-aware memory management is a key factor in improving the effectiveness of LLM-based agents.

5.3 Effects of semantic memory size

Table 5 highlights the impact of dynamic memory management on the size and quality of the semantic memory repository. Static configurations maintain a fixed memory size (3,286 semantic facts), whereas Semantic-DP actively reduces the repository to 2,512 memory items, corresponding to a 23.5% reduction without any loss in accuracy. This shows that a significant portion of the stored information (memory items) is either redundant or of low utility, and can be safely removed without harming the agent's effectiveness. In practice, this leads to a more compact and efficient memory repository that is easier to search and less prone to retrieving irrelevant or misleading memory items.

Table 4. Average cosine similarity scores between retrieved memory items and ARC questions (the higher the value, the higher the similarity).

Configuration	Valid. Similarity	Test Similarity
Reflection	0.5072	0.5112
Semantic	0.5464	**0.5521**
Concatenated	0.5274	0.5333
Semantic-DP	0.4659	0.4772
Concatenated-DP	0.5281	0.5149
Semantic-DP^{++}	0.5101	0.5156
Concatenated-DP^{++}	0.5113	0.5175

The behavior of Semantic-DP^{++} further demonstrates the benefits of active memory curation. In this configuration, the memory repository initially grows to 3,993 as the Self-Refinement mechanism stores revised versions of imprecise or low-quality memory items alongside the original ones. After this expansion phase, Dynamic Pruning removes the versions that prove less useful, and the memory repository stabilizes at 3,504 memory items before evaluation on the test set. This pattern shows that the solution does not simply prune or compress memory, but also improves its content by replacing weaker memory items with more precise and informative versions.

6 Discussion

Our experimental results demonstrated that structured interaction between semantic and reflective memories improves both retrieval quality and downstream

Table 5. Semantic memory statistics. *Size* is the total number of semantic memory items stored in the repository. *Harmful (%)* is the proportion of stored memory items classified as incorrect. *Retrieved (%)* denotes the proportion of test-time retrievals in which harmful memory items were selected.

	Validation Phase		**Test Phase**		
Configuration	**Size**	**Harmful (%)**	**Size**	**Harmful (%)**	**Retrieved (%)**
Semantic	3,286	47.47	3,286	47.47	46.03
Concatenated	3,286	47.47	3,286	47.47	45.88
Semantic-DP	3,286	47.47	2,512	43.51	35.17
Concatenated-DP	3,286	47.47	2,512	43.51	37.08
Semantic-DP^{++}	3,993	46.51	3,504	39.44	32.54
Concatenated-DP^{++}	3,993	46.51	3,559	**39.48**	**32.55**

reasoning. The results presented in Table 3 and Table 5 provided a consistent picture of how dynamic semantic memory management enhances agent behavior. Moving from static semantic augmentation to dynamic variants yields significant accuracy gains, indicating that the benefit arises not only from adding memory, but also from improving how that memory is selected, maintained, and updated over time.

The semantic memory statistics help explain this effect. Static configurations maintain a fixed memory size with a high proportion of harmful or low-utility memory items, suggesting that simple accumulation introduces noise and degrades retrieval quality. In contrast, dynamic pruning reduces repository size. It lowers the proportion of harmful memory items, demonstrating that selective forgetting enables the agent to focus on more relevant and trustworthy knowledge without sacrificing effectiveness. Self-refinement further strengthens this process by revising and consolidating existing knowledge rather than merely removing it. Although refinement temporarily increases memory size, the resulting repository contains fewer harmful memory items and more precise information, yielding a more stable and reliable knowledge base.

The pruning and refinement mechanisms partially tackle issues related to memory drift and knowledge integrity by minimizing harmful content and fostering more consistent semantic representations over time. Likewise, the incorporation of semantic cues into the reflection process offers initial evidence of enhanced semantic-reflective alignment, as reflections become more anchored in curated knowledge rather than depending solely on local reasoning traces.

While these findings support the feasibility of bidirectional memory interaction, they also highlight several unresolved challenges. A primary limitation concerns the reliability and calibration of reflective judgments. The proposed architecture assumes that the agent's self-critique provides a useful signal for scoring and refining memory; yet, it does not explicitly model uncertainty or detect unreliable critiques. To partially investigate this limitation, we conducted an external evaluation using GPT-5 as a fact verifier. The results show that the complete architecture reduces the prevalence of incorrect semantic memory items by 24%

relative to the static baseline, confirming its effectiveness in filtering erroneous knowledge. However, the same process removes 14% of correct memory items, indicating that overly conservative pruning thresholds or misclassification by the utility signal $\lambda_i(t)$ can discard valid knowledge. This exposes a fundamental trade-off in lifelong learning systems: aggressive consolidation risks losing valuable but infrequently used information, while lenient pruning allows noise to accumulate.

Closely tied to this trade-off is the issue of accountability. Attributing an erroneous removal or rewrite to its cause is inherently difficult: the utility signal $\lambda_i(t)$ compresses the agent's self-critique into a scalar whose rationale is not preserved, responsibility for a wrong answer is shared between the retrieved memory items and the model's parametric knowledge, and curation decisions compound across interaction cycles. Making these judgments interpretable and traceable remains an open requirement for deploying lifelong learning agents in high-stakes settings.

Additionally, although pruning and refinement help control memory growth, we have only evaluated the current implementation at a moderate scale. To extend this bidirectional architecture to long-horizon operation with very large memory, we need to develop more efficient indexing, retrieval, and update strategies. Another open issue involves task transfer and abstraction. Although refined reflections capture useful patterns, the system does not explicitly control the level of abstraction for storing or reusing knowledge across tasks, which might limit generalization. Moreover, our evaluation focused on aggregate accuracy metrics, leaving room for a qualitative analysis of how each memory component influences individual reasoning traces and the evolution of the agent's behavior. In future work, we plan to pursue this interpretability-focused analysis and make direct comparisons with architectures such as RMM, SAGE, and MemInsight.

Overall, our proposed bidirectional memory architecture increases (i) knowledge quality, (ii) decreases noisy, harmful content, (iii) mitigates the risk of memory drift, which in turn (iv) improves agent task accuracy. Addressing the broader set of challenges outlined here will allow us to design truly robust, scalable, and trustworthy lifelong learning agents.

7 Conclusion

Existing studies on LLM-based agent memory treat semantic and reflection modules separately, leaving it unclear how knowledge should meaningfully support reflection. This article investigated how to bridge semantic and reflection memories via a novel bidirectional memory architecture that integrates them, enabling continual learning in LLM-based agents. The knowledge extracted from the semantic module grounds the reflections, while the reflection memories curate, update, and refine semantic memory items. Experimental results on the ARC Challenge dataset showed that, although independent memory components yield relative improvements over the baseline, the simple concatenation of the different memory types remains suboptimal. The structured interaction between memories yielded the strongest gains, demonstrating that adaptive refinement, stabilization,

and pruning are essential for achieving more robust long-horizon reasoning in LLM-based agents. Future work will focus on improving memory efficiency by developing compression and consolidation strategies that preserve the stability and usefulness of refined memories while limiting unbounded growth. Explicitly integrating episodic memory into the bidirectional loop, beyond the episode-anchored reflections already used, is a natural extension. In addition, extending the proposed architecture to larger language models and evaluating it across a broader range of reasoning benchmarks will allow us to assess scalability, robustness, and generalization. Together, these directions will help characterize the conditions under which bidirectional memory interaction yields the greatest benefits and the limits of dynamic memory refinement for lifelong learning in LLM-based agents.

Acknowledgments. This study was sponsored by Petróleo Brasileiro S.A. (PETROBRAS) within the project "*Application of Large Language Models (LLMs) for online monitoring of industrial processes*" conducted in partnership with the University of Campinas.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., Tafjord, O.: Think you have solved question answering? try arc, the AI2 reasoning challenge. CoRR **abs/1803.05457** (2018), `http://arxiv.org/abs/1803.05457`
2. Ebbinghaus, H.: Memory: A Contribution to Experimental Psychology. Teachers College, Columbia University, New York (1913), original work published 1885 as Über das Gedächtnis
3. Hadj Mohamed, A., Villas, L.A., dos Reis, J.C.: Leveraging llm reflection to improve small language model agents' capabilities. In: Marcelloni, F., Madani, K., van Stein, N., Filipe, J. (eds.) Computational Intelligence. pp. 432–453. Springer Nature Switzerland, Cham (2026)
4. Hou, Y., Tamoto, H., Miyashita, H.: "my agent understands me better": Integrating dynamic human-like memory recall and consolidation in llm-based agents. In: Extended Abstracts of the CHI Conference on Human Factors in Computing Systems. CHI EA '24, Association for Computing Machinery, New York, NY, USA (2024). https://doi.org/10.1145/3613905.3650839
5. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive nlp tasks. In: Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20, Curran Associates Inc., Red Hook, NY, USA (2020)
6. Li, Y., Yang, C., Ettinger, A.: When hindsight is not 20/20: Testing limits on reflective thinking in large language models. In: Duh, K., Gomez, H., Bethard, S. (eds.) Findings of the Association for Computational Linguistics: NAACL 2024. pp. 3741–3753. Association for Computational Linguistics, Mexico City, Mexico (Jun 2024). https://doi.org/10.18653/v1/2024.findings-naacl.237

7. Liang, X., Tao, M., Xia, Y., Wang, J., Li, K., Wang, Y., He, Y., Yang, J., Shi, T., Wang, Y., Zhang, M., Wang, X.: Sage: Self-evolving agents with reflective and memory-augmented abilities. Neurocomputing **647**, 130470 (2025). https://doi.org/https://doi.org/10.1016/j.neucom.2025.130470
8. Lippmann, P., Spaan, M.T.J., Yang, J.: Positive experience reflection for agents in interactive text environments. In: Kamalloo, E., Gontier, N., Lu, X.H., Dziri, N., Murty, S., Lacoste, A. (eds.) Proceedings of the 1st Workshop for Research on Agent Language Models (REALM 2025). pp. 131–142. Association for Computational Linguistics, Vienna, Austria (Jul 2025). https://doi.org/10.18653/v1/2025.realm-1.10
9. Liu, W., Zhang, R., Zhou, A., Gao, F., Liu, J.: Echo: A large language model with temporal episodic memory. arXiv preprint arXiv:2502.16090 (2025)
10. Park, J.S., O'Brien, J., Cai, C.J., Morris, M.R., Liang, P., Bernstein, M.S.: Generative agents: Interactive simulacra of human behavior. In: Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology. UIST '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3586183.3606763
11. Pink, M., Wu, Q., Vo, V.A., Turek, J., Mu, J., Huth, A., Toneva, M.: Position: Episodic memory is the missing piece for long-term llm agents. arXiv preprint arXiv:2502.06975 (2025)
12. Renze, M., Guven, E.: Self-Reflection in Large Language Model Agents: Effects on Problem-Solving Performance. In: 2024 2nd International Conference on Foundation and Large Language Models (FLLM). p. 80. IEEE (Nov 2024). https://doi.org/10.1109/FLLM63129.2024.10852426
13. Salama, R., Cai, J., Yuan, M., Currey, A., Sunkara, M., Zhang, Y., Benajiba, Y.: MemInsight: Autonomous memory augmentation for LLM agents. In: Christodoulopoulos, C., Chakraborty, T., Rose, C., Peng, V. (eds.) Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing. pp. 33136–33152. Association for Computational Linguistics, Suzhou, China (Nov 2025). https://doi.org/10.18653/v1/2025.emnlp-main.1683
14. Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., Yao, S.: Reflexion: language agents with verbal reinforcement learning. In: Proceedings of the 37th International Conference on Neural Information Processing Systems. NIPS '23, Curran Associates Inc., Red Hook, NY, USA (2023)
15. Sumers, T., Yao, S., Narasimhan, K., Griffiths, T.: Cognitive architectures for language agents. Transactions on Machine Learning Research **2024** (2024)
16. Tan, Z., Yan, J., Hsu, I.H., Han, R., Wang, Z., Le, L., Song, Y., Chen, Y., Palangi, H., Lee, G., Iyer, A.R., Chen, T., Liu, H., Lee, C.Y., Pfister, T.: In prospect and retrospect: Reflective memory management for long-term personalized dialogue agents. In: Che, W., Nabende, J., Shutova, E., Pilehvar, M.T. (eds.) Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 8416–8439. Association for Computational Linguistics, Vienna, Austria (Jul 2025). https://doi.org/10.18653/v1/2025.acl-long.413
17. Woźniak, P.A., Gorzelańczyk, E.J., Murakowski, J.A.: Two components of long-term memory. Acta Neurobiologiae Experimentalis **55**(4), 301–305 (1995). https://doi.org/10.55782/ane-1995-1090
18. Wu, Y., Xu, G., Dongchen, Z.: Self-reflection like humans, editable-LLM (e-LLM) is all you need. In: Submitted to Tsinghua University Course: Advanced Machine Learning (2024), `https://openreview.net/forum?id=rk6PJlu562`, under review

19. Zhang, B., Zhang, X., Zhang, J., Yu, J., Luo, S., Tang, J.: CoT-based synthesizer: Enhancing LLM performance through answer synthesis. In: Che, W., Nabende, J., Shutova, E., Pilehvar, M.T. (eds.) Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 6286–6303. Association for Computational Linguistics, Vienna, Austria (Jul 2025). https://doi.org/10.18653/v1/2025.acl-long.315
20. Zhao, A., Huang, D., Xu, Q., Lin, M., Liu, Y.J., Huang, G.: Expel: Llm agents are experiential learners. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, pp. 19632–19642 (2024). https://doi.org/10.1609/aaai.v38i17.29936