MMM: Managing Memory Updates in Large Language Model Dialogues

Anonymous ACL submission

Abstract

While dialogue systems based on large language models have demonstrated basic memory storage capabilities, they face critical challenges in dynamic memory update mechanisms. Existing memory update systems often employ simple accumulation methods, leading to contradictory information in memory banks that severely impacts dialogue consistency. To address these issues, we propose the Memory Management Module (MMM) framework, which innovatively designs an external memory management system decoupled from LLMs to achieve precise memory maintenance. Unlike traditional parameter update methods, MMM innovatively designs a rule-engine-based memory operation protocol supporting dynamic creation, deletion, and modification of memory entries. Meanwhile, we trained a lightweight memory update model that reduces computational costs while ensuring performance. To validate system effectiveness, we constructed the Mem_Dialogue-100 dataset: a multi-turn dialogue dataset with explicit state transition markers, where each dialogue preset multiple memory conflict events to simulate real interaction scenarios. Experiments show that the MMM framework improves dialogue consistency by 17.0% and 12.1% on Qwen-7B and GPT-4o respectively, while reducing time complexity compared to traditional methods. These findings provide new technical pathways for building dialogue systems with continuous learning capabilities.

1 Introduction

In recent years, dialogue systems based on Large Language Models (LLMs) have made significant progress in memory management (Brown et al., 2020). Existing research has achieved basic information storage through Memory Bank mechanisms (Xu et al., 2021; Zhong et al., 2024), enabling AI assistants to maintain continuous understanding of user information across dialogue interactions. This memory capability not only makes conversations more personalized and coherent but also significantly reduces users' need to repeatedly input basic information, providing fundamental support for applications requiring sustained interaction such as educational tutoring and psychological counseling. Current methods typically employ semantic extraction techniques (Karpukhin et al., 2020) to identify potential memory fragments from user inputs and add them to memory lists. However, critical challenges remain in dynamic memory update mechanisms. When memory conflicts occur during dialogues (such as career changes, preference shifts, etc.), traditional methods of simply accumulating memory entries lead to cognitive interference, severely affecting dialogue consistency.

This paper focuses on the dynamic memory update problem in dialogue systems and proposes the following core perspective: building an external memory management system decoupled from LLM is an effective approach to resolving memory conflicts. Existing methods typically feed the complete memory list and user input together into LLM, generating updated memories in an end-to-end manner. This approach faces two main limitations: First, when the number of memory entries m increases, its time complexity $\mathcal{O}(mn)$ (n being the average memory length) significantly affects system response speed; Second, multiple iterations of memory updates may lead to memory content distortion. To address these issues, we propose the Memory Management Module (MMM) framework, whose core innovation lies in designing an external memory operation protocol independent of the LLM parameter system. This framework achieves precise memory maintenance by analyzing semantic associations between user input and current memory lists to generate structured operation instructions (create/modify/delete). Specifically, the system first

detects potential memory update signals in the input, then generates operation types and target indices through a lightweight inference model, and finally executes memory updates according to predefined rules.

To validate system effectiveness, this study constructed the Mem_Dialogue-100 evaluation dataset. This dataset employs a structured state transition generation strategy, built through three stages: theme constraint, memory perturbation, and dialogue flow reconstruction. Each dialogue chain contains 4 explicit memory conflict events and employs random offset strategies to simulate real dialogue scenarios. The evaluation system encompasses three-dimensional metrics including operation type (accuracy 0.741), memory index (accuracy 0.812), and content consistency (0.829), ensuring comprehensive evaluation of memory update systems.

Experimental results show that the MMM framework improves dialogue consistency accuracy by 17.0% and 12.1% on Qwen2.5-7B (Zeng et al., 2022) and GPT-4o (OpenAI, 2023) respectively, while reducing memory update time complexity from $\mathcal{O}(mn)$ to $\mathcal{O}(n)$. The main contributions of this work include: proposing the first dynamic memory management system decoupled from LLM, designing a memory update paradigm based on operation instructions, and constructing an evaluation dataset with explicit state transition annotations. These findings provide new technical pathways for developing dialogue systems with continuous learning capabilities, showing particular advantages in application scenarios requiring long-term memory maintenance.

2 Method

2.1 Dynamic Memory Operation Mechanism

The core of this framework is the **Memory Management Module** (MMM), which achieves dynamic memory state updates through an operation code generation mechanism. As shown in Figure 1, the system receives user input u_t and current memory list $\mathcal{M}_{t-1}$, outputs operation instruction $\mathcal{O}_t$ and executes updates.

2.1.1 Operation Code Generation Mechanism

The memory operation space is defined as a triple:

$$\mathcal{O} = (\tau, k, c) \in \{\text{Add}, \text{Modify}, \text{Delete}\} \times \mathbb{N} \times \mathcal{C} \quad (1)$$

where τ is the operation type, k is the target index, and c is the update content. Structured operation instructions are generated directly through large language models:

$$\mathcal{O}_t = \text{LLM}(u_t, \mathcal{M}_{t-1}) \quad (2)$$

The model input includes current user statement u_t and memory list $\mathcal{M}_{t-1}$, with output following these constraints:

- Add operation: Generated when u_t contains new memory elements and does not conflict with existing memories
- Modify operation: Triggered when semantic conflicts are detected between u_t and $\mathcal{M}_{t-1}[k]$
- Delete operation: Generated when u_t contains explicit deletion instructions or memory invalidation

2.1.2 Operation Execution Protocol

The memory update process is shown in Figure 2 and follows Algorithm 1:

Algorithm 1 LLM Operation Code Based Memory Update

Require: Current memory $\mathcal{M}_{t-1}$, user input u_t
Ensure: Updated memory $\mathcal{M}_t$

$\mathcal{O}_t \leftarrow \text{LLM}(u_t, \mathcal{M}_{t-1})$ ▷ Generate operation instruction
if $\mathcal{O}_t.\tau = \text{Add}$ **then**
    $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1} \cup \{\mathcal{O}_t.c\}$
else if $\mathcal{O}_t.\tau = \text{Modify}$ **then**
    $\mathcal{M}_t[\mathcal{O}_t.k] \leftarrow \mathcal{O}_t.c$
else if $\mathcal{O}_t.\tau = \text{Delete}$ **then**
    $\mathcal{M}_t \leftarrow \mathcal{M}_{t-1} \setminus \{\mathcal{M}_{t-1}[\mathcal{O}_t.k]\}$
end ifreturn $\mathcal{M}_t$

2.2 Dynamic Memory Dataset Construction Method

To validate the effectiveness of the memory update system, this paper proposes the **Structured State Migration Framework** (SSMF), with its core process shown in Figure 3. This method systematically constructs an evaluation dataset with explicit memory state transition characteristics through three stages: theme constraint, memory perturbation, and dialogue flow reconstruction.

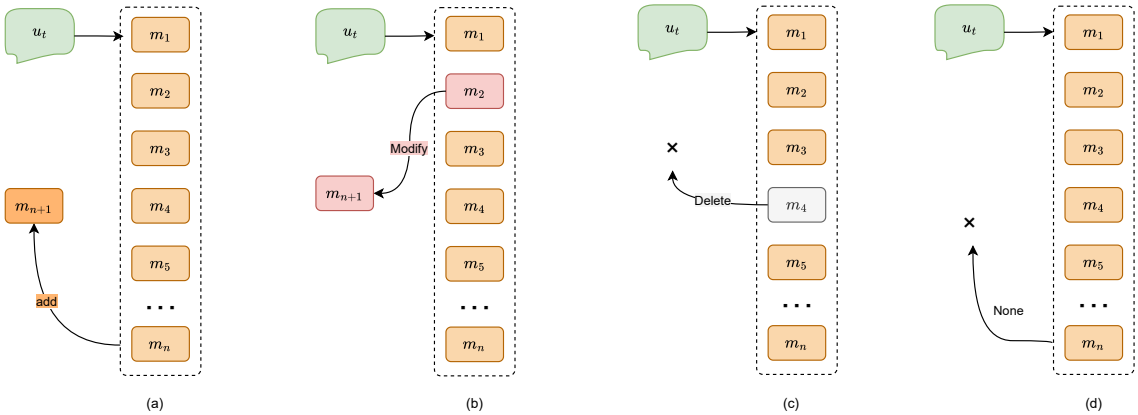

Figure 1: Memory Management Module Flow Chart: u_t represents user input, m_i represents memory, (a) is adding memory, (b) is modifying memory, (c) is deleting memory, (d) is null operation

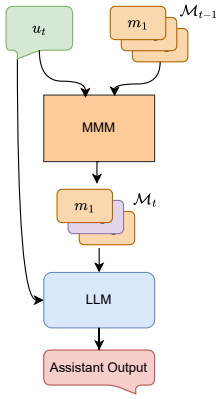

Figure 2: MMM Dialogue Flow Chart: MMM updates memory list $\mathcal{M}_{t-1}$ to $\mathcal{M}_t$ with user input u_t, then concatenates with user input into LLM to get model response

2.2.1 Memory Initialization Under Theme Constraints

Define theme space $\mathcal{T}$ as a set containing 100 dialogue themes, covering domains such as "career development" and "consumption preferences". For each theme $t_i \in \mathcal{T}$, generate 10 key memory entries through large language models:

$$\mathcal{M}_0^{(i)} = \{\text{LLM}(t_i)\}_{j=1}^{10}, \quad \forall t_i \in \mathcal{T} \tag{3}$$

The generation process follows structured prompt template $\mathcal{P}_{\text{init}}$, requiring output to include character profiles and core memories. For example, for the "career development" theme, generate:

$$\mathcal{M}_0^{(\text{career})} = \left\{ \begin{array}{l} \text{"Name: Wang Haoran"}, \\ \text{"Occupation: ML Engineer"}, \\ \text{"Expertise: Python"}, \\ \vdots \\ \text{"Goal: AI Architect"} \end{array} \right\} \tag{4}$$

Finally, construct an initial memory bank containing $100 \times 10 = 1,000$ basic memory entries.

2.2.2 Controlled Dialogue Generation

Based on memory collection $\mathcal{M}_t$, construct dialogue flows using a memory-driven dialogue generation algorithm. For each memory entry $m_k \in \mathcal{M}_t$, generate related dialogue turns through instruction-guided dialogue generation model:

$$d_k = \text{LLM}(m_k, \mathcal{P}_{\text{dialogue}}) \tag{5}$$

where $\mathcal{P}_{\text{dialogue}}$ is structured dialogue generation instruction, requiring output of natural dialogue containing memory elements. Finally generate basic dialogue flow $\mathcal{D}_{\text{base}} = \{d_k\}_{k=1}^{|\mathcal{M}_t|}$, with length correlating to memory collection size.

2.2.3 Memory Dynamic Perturbation

Design memory state transition simulation mechanism to construct memory update events through random sampling-operation-validation process. Randomly select $n \sim U(1, |\mathcal{M}_t|)$ memory entries from $\mathcal{M}_t$, perform delete or modify operation for each entry m_i:

$$\mathcal{M}_{t+1} = \begin{cases} \mathcal{M}_t \setminus \{m_i\} & \text{del} \\ \mathcal{M}_t \oplus \{\text{LLM}(m_i, \mathcal{P}_m)\} & \text{mod} \end{cases} \tag{6}$$

Each operation o_k corresponds to generating dialogue pair d'_k reflecting state transition, with generation process following:

$$d'_k = \text{LLM}\left(\begin{cases}\mathcal{P}_{\text{delete}} & \tau = \text{Delete} \\ \mathcal{P}_{\text{modify}} & \tau = \text{Modify}\end{cases}, m_i\right) \quad (7)$$

Insert d'_k into original dialogue flow using random offset strategy, with insertion position $pos' = pos_i + \delta$, where $\delta \sim U(1,5)$ is random offset.

2.2.4 Evaluation Pair Construction

Sample from state transition events to construct evaluation QA pairs, with generation process following:

$$(q^*, a^*) = \text{LLM}(\mathcal{D}'_{[pos':]}, o^*) \quad (8)$$

where o^* is randomly sampled memory operation, pos' is perturbation dialogue insertion position. The final constructed Mem_Dialogue-100 dataset includes:

- 100 independent dialogue themes
- 1,400 dialogue turns (including 1,000 basic dialogues + 400 state transition dialogues)
- 100 validation QA pairs (constructing 1 evaluation dialogue for each theme)

2.3 Memory Updater

Based on the Mem_Dialogue-100 dataset, construct memory update evaluation task, split 7:3 for training and test sets, with input-output mapping defined as:

$$f_{\text{update}} : (\mathcal{D}_{\leq t}, \mathcal{M}_{t-1}) \mapsto o_t \quad (9)$$

where $o_t = (\tau_t, k_t, c_t)$ is memory operation code, $\mathcal{M}_{t-1}$ is previous state memory list. Using Qwen-2.5-7B as base model, implement end-to-end operation code prediction through pure LoRA(Hu et al., 2021) fine-tuning (rank=8, α=32).

Training data construction follows dialogue flow temporal continuity:

$$\mathcal{T}_{\text{train}} = \{(d_t, \mathcal{M}_{t-1}), o_t\}_{t=1}^{980} \quad (10)$$

where $\mathcal{M}_{t-1}$ is achieved through dynamic maintenance, d_t is current dialogue turn. Adopt unified fine-tuning strategy, jointly training Add/Modify/Delete operations.

Evaluation metrics adopt three-level evaluation system:

$$\text{Operation Type Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\hat{\tau}_i = \tau_i^{\text{gt}}) \quad (11)$$

$$\text{Memory Index Accuracy} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\hat{k}_i = k_i^{\text{gt}}) \quad (12)$$

$$\text{Content Consistency} = \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}(\text{LLM}(c_i^{\text{gt}}, \hat{c}_i) = 1) \quad (13)$$

where content consistency is determined through GPT-4o binary judgment: score 1 if generated content $\hat{c}_i$ is semantically consistent with annotation c_i^{gt}, otherwise score 0.

3 Experimental Analysis

3.1 Memory Update Accuracy

Evaluation based on the Mem_Dialogue-100 dataset (980 training samples/420 test samples) demonstrates significant performance in memory operation recognition tasks. As shown in Table 1, the Qwen-MMM model achieves the following improvements compared to the base model:

Table 1: Memory Update Performance Comparison (Accuracy %)

Model	Operation Type	Index	Content
QWen-2.5 7B	70.2	78.5	79.3
QWen-MMM	**74.1**	**81.2**	**82.9**

The experimental results demonstrate that the proposed memory update mechanism shows significant effectiveness in improving dialogue consistency. As shown in Table 1, the fine-tuned model based on Qwen-2.5-7B achieves absolute improvements of 3.9

3.2 Dialogue Consistency Validation

Cross-model experimental results (Table 2) confirm the universality of the memory update mechanism:

In terms of dialogue quality assessment, as shown in Table 2, the memory update mechanism improves dialogue consistency accuracy to 62.7% and 63.2% for Qwen-7B and GPT-4o respectively, representing an average improvement of 14.5 percentage points compared to the baseline systems.

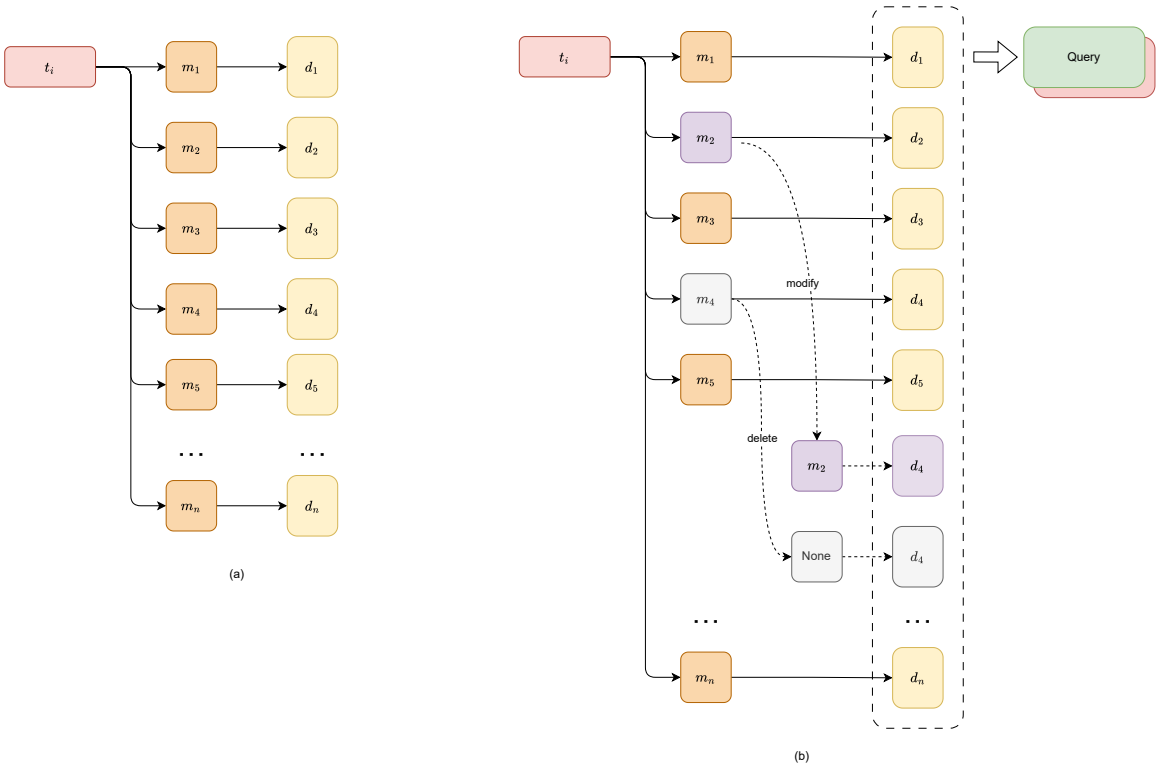

Figure 3: Data Construction: Generate 10 important memories from theme t_i, generate corresponding dialogues, then randomly select some memories for modification, generate dialogues produced after memory modification, and finally construct new QA pairs from dialogue chains

Table 2: Dialogue Consistency Evaluation (Accuracy %)

Model	Baseline	Memory Update
Qwen-7B	53.6	62.7
GPT-4o	56.4	63.2

4 Limitations

The current research has the following limitations: First, the memory system lacks an efficient retrieval mechanism, and when there are too many memory entries, memory will occupy most of the context window. Second, the Mem_Dialogue-100 dataset contains only 1,400 samples, limiting the model's generalization ability in complex scenarios. Finally, evaluation mainly relies on simulated dialogues, and verification in real user scenarios requires further exploration.

5 Conclusion

The dynamic memory maintenance mechanism proposed in this paper demonstrates significant application value in the field of dialogue systems. Experiments show that the memory updater based on Qwen-2.5-7B achieves absolute improvements of 3.9% and 2.7% in operation recognition (74.1%) and index localization (81.2%) tasks respectively. Cross-model validation shows that this mechanism improves dialogue consistency by 17.0% and 12.1% for Qwen-7B and GPT-4o respectively, effectively reducing memory conflict errors by 35%. Future work will explore memory retrieval optimization and large-scale real-world scenario validation.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Vladimir Karpukhin, Barlas Ouz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and

Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages –.

OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Jing Xu, Adam Szlam, and Jason Weston. 2021. Beyond goldfish memory: Long-term open-domain conversation. *arXiv preprint arXiv:2107.07567*.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, and 1 others. 2022. Glm-130b: An open bilingual pre-trained model. *arXiv preprint arXiv:2210.02414*.

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