Think-in-Memory: Recalling and Post-thinking Enable LLMs with Long-Term Memory

Lei Liu*
liulei1497@gmail.com
CUHK-Shenzhen, Ant Group

Xiaoyan Yang
joyce.yxy@antgroup.com
Ant Group

Yue Shen†
zhanying@antgroup.com
Ant Group

Binbin Hu, Zhiqiang Zhang
{bin.hbb,lingyao.zzq}@antfin.com
Ant Group

Jinjie Gu
jinjie.gjj@antfin.com
Ant Group

Guannan Zhang
zgn138592@antfin.com
Ant Group

ABSTRACT

Memory-augmented Large Language Models (LLMs) have demonstrated remarkable performance in long-term human-machine interactions, which basically relies on iterative recalling and reasoning of history to generate high-quality responses. However, such repeated recall-reason steps easily produce biased thoughts, *i.e.*, inconsistent reasoning results when recalling the same history for different questions. On the contrary, humans can keep thoughts in the memory and recall them without repeated reasoning. Motivated by this human capability, we propose a novel memory mechanism called TiM (Think-in-Memory) that enables LLMs to maintain an evolved memory for storing historical thoughts along the conversation stream. The TiM framework consists of two crucial stages: (1) before generating a response, a LLM agent recalls relevant thoughts from memory, and (2) after generating a response, the LLM agent post-thinks and incorporates both historical and new thoughts to update the memory. Thus, TiM can eliminate the issue of repeated reasoning by saving the post-thinking thoughts as the history. Besides, we formulate the basic principles to organize the thoughts in memory based on the well-established operations, (*i.e.*, insert, forget, and merge operations), allowing for dynamic updates and evolution of the thoughts. Furthermore, we introduce Locality-Sensitive Hashing into TiM to achieve efficient retrieval for the long-term conversations. We conduct qualitative and quantitative experiments on real-world and simulated dialogues covering a wide range of topics, demonstrating that equipping existing LLMs with TiM significantly enhances their performance in generating responses for long-term interactions.

KEYWORDS

Large Language Model, Response Generation, Long-term Memory

1 INTRODUCTION

Impressive advancements in Large Language Models (LLMs) have revolutionized the interaction between human beings and artificial intelligence (AI) systems. These advancements have particularly showcased superior performance in human-agent conversations, as demonstrated by ChatGPT [1] and GPT-4 [2]. From finance [3] and healthcare [4] to business and customer service [5], these advanced LLMs exhibit a remarkable ability to understand questions and generate corresponding responses. Notably, the large model scale, reaching up to hundreds of billions of parameters, enables the emergence of such human-like abilities within LLMs [6].

Despite the remarkable abilities of LLMs pre-trained on large corpora, LLM-based AI agents still face a significant limitation in long-term scenarios, *i.e.*, inability to process exceptionally lengthy inputs [7]. This is particularly important in some specific tasks, *e.g.*, medical AI assistants [4] rely on the symptoms of past conversations to provide accurate clinical diagnosis. Thus, LLMs without the capability of dealing with long-term inputs may hinder the diagnosis accuracy due to forgetting important disease symptoms (see in Section 4.4). Therefore, it is necessary to develop AI systems with long-term capabilities for more accurate and reliable interactions.

There have been various studies conducted to improve the capabilities of LLMs to handle long-term inputs. Overall, these studies can be roughly divided into two types: (1) **Internal memory based methods** [8] aims to reduce the computational costs of self-attention for expanding the sequence length. To accommodate longer input texts, special positional encoding should be exploited to learn relative positions. For example, [9] explored a block-local Transformer with global encoder tokens, combined with additional long input pre-training. (2) **External memory based methods** (also called long-term memory mechanism [10]) generally utilize a physical space as a memory cache to store historical information, where relevant history can be read from the memory cache to augment LLMs without forgetting. In particular, both token and raw text can be maintained as history in the memory. For instance, [11] demonstrated a significant performance improvement by augmenting LLMs with an external memory cache containing trillions of tokens assisted by BERT embeddings [12]. It should be noticed that token-based memory mechanism requires to adjust the LLM's architecture for adaption, which is hard to be combined with different LLMs. By accessing an external memory cache, the augmented LLMs have achieved new state-of-the-art records in various language modeling benchmarks, which generally performs better than internal memory based methods. Therefore, this work focuses on designing an LLM-agnostic external memory mechanism to enhanced the memorization capacity of LLMs.

In general, the utility of memory-augmented LLMs primarily hinges on their ability for iterative recalling and repeated reasoning over the history in an external memory cache. In detail, for conversations after the n-th turn, LLMs are required to re-understand and re-reason the history from 0-th to $(n-1)$-th conversations. For example, as shown in Figure 1, to answer the questions of 2-th and 3-th turns, LLMs recall 1-th turn and reason over it for twice.

*Work was done when Lei Liu was a research intern at Ant Group.
†Corresponding Author.

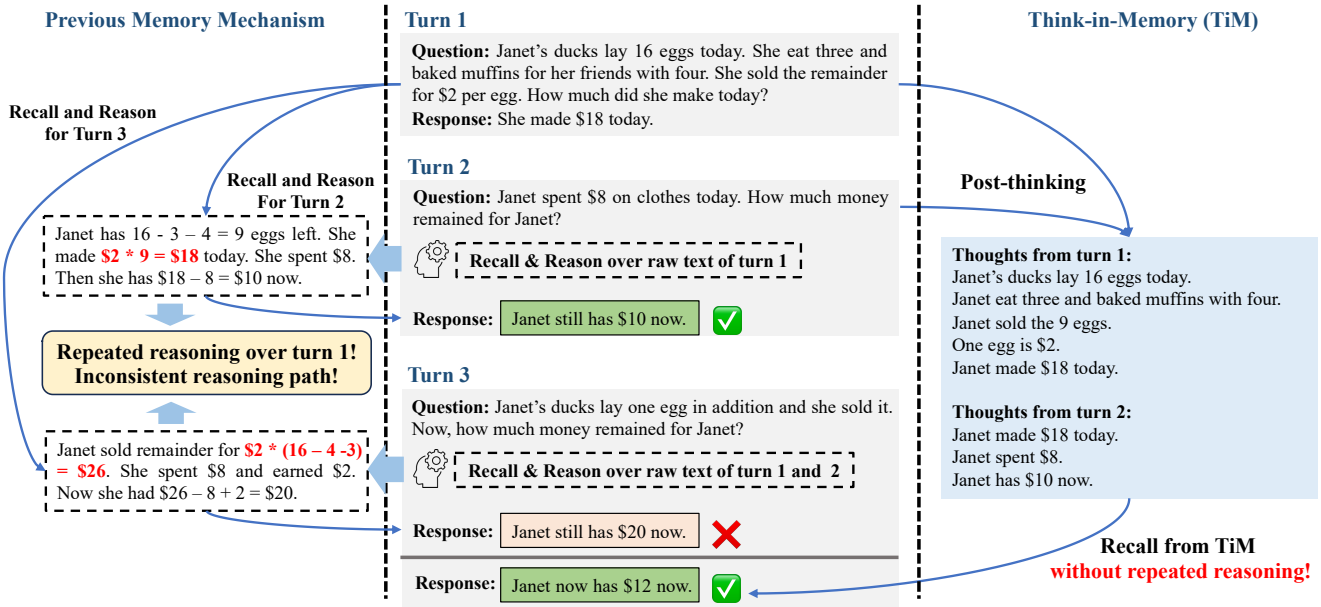

Figure 1: Comparisons between previous memory mechanisms with our proposed TiM. (Left): Existing memory mechanisms mainly save raw text of previous turns, which require repeated reasoning over the same history. This easily leads to the inconsistent reasoning path (*i.e.*, red part of the left) with wrong response. (Right): The proposed TiM stores the thoughts of LLMs for previous turns, which can avoid such inconsistency without repeated reasoning (*i.e.*, red part of the right).

Unfortunately, this paradigm is prone to encountering several issues and potentially causes a performance bottleneck in real-world applications. The main issues are shown in follows:

- **Inconsistent reasoning paths.** Prior studies [13, 14] has shown that LLMs easily generate diverse reasoning paths for the same query. As shown in Figure 1 (Left), LLMs give a wrong response due to inconsistent reasoning over the context.
- **Unsatisfying retrieval cost.** To retrieve relevant history, previous memory mechanisms need to calculate pairwise similarity between the question and each historical conversation, which is time-consuming for long-term dialogue.

To address these concerns, we would like to advance one step further in memory-augmented LLMs with the analogy to the typical process of metacognition [15], where the brain saves thoughts as memories rather than the details of original events. Thus, in this work, we propose a **T**hink-**i**n-**M**emory (TiM) framework to model the human-like memory mechanism, which enables LLMs to remember and selectively recall historical thoughts in long-term interaction scenarios. Specifically, as shown in Figure 2, the TiM framework is divided into two stages: (1) In the recalling stage, LLMs generate the response for the new query with recalling relevant thoughts in the memory; (2) In the post-thinking stage, the LLM engages in reasoning and thinking over the response and saves new thoughts into an external memory. Besides, to mirror the cognitive process of humans, we formulate some basic principles to organize the thoughts in memory based on the well-established operations (*e.g.*, insert, forget, and merge operations), allowing for dynamic updates and evolution of the thoughts. Specifically, TiM is built on a hash-based retrieval mechanism (*i.e.*, Locality-Sensitive Hashing [16]) to support efficient hand-in (*i.e.*, insert thoughts) and hand-out (*i.e.*, recall thoughts) operations. Additionally, TiM is designed to be LLM-agnostic, which means it can be combined with various types of language models. This includes closed-source LLMs such as ChatGPT [1], as well as open-source LLMs like ChatGLM[17].

The key contributions of this work are summarized as follows:

- We propose a novel human-like long-term memory mechanism called TiM, enabling LLMs to remember and selectively recall thoughts. TiM can let LLM think in memory without repeated reasoning over the long-term history.
- We formulate some basic principles to organize the thoughts in memory based on the well-established operations, which mirrors human cognitive process to empower dynamic updates and evolution for the thoughts in memory. Besides, a hash-based retrieval mechanism is introduced for efficient utilization of TiM.
- We conducted extensive experiments on multi-turn dialogue datasets. The results indicate that our method can substantially enhance LLM's performance across various dimensions: (1) It enables diverse topics ranging from open to specific domains; (2) It supports bilingual languages in both Chinese and English; (3) It improves response correctness and coherence.

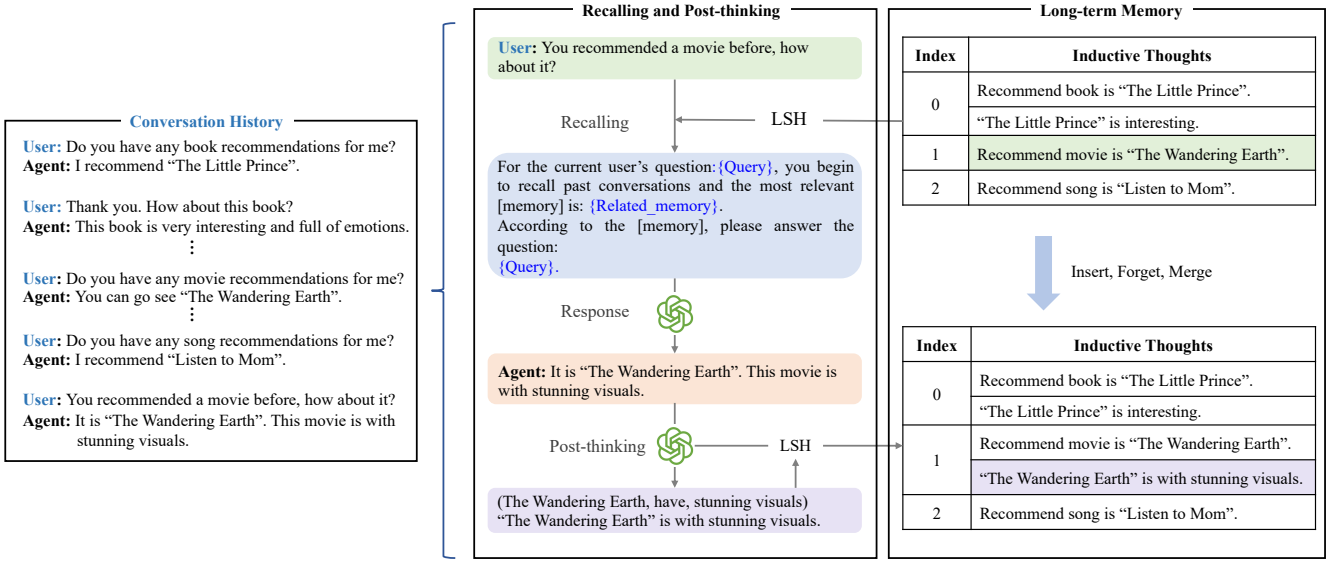

Figure 2: The overview of TiM framework. LLMs firstly recall history and give response for the question. Then new thoughts can be generated via the post-thinking step. These thoughts are saved as the memory to avoid repeated reasoning on the history.

2 RELATED WORK

2.1 Large Language Models

Recently, Large Language Models (LLMs) have attracted significant attention for their superior performance on a wide range of Natural Language Processing tasks, such as machine translation [18], sentiment analysis [19], and question answering systems [20]. These advancements are indeed supported by the developments of deep learning techniques and the availability of vast amounts of text data. From the perspective of open source, existing LLMs can roughly divided into two types: (1) cutting-edge closed-source LLMs, *e.g.*, PaLM [21], GPT-4 [2], and ChatGPT [1]; (2) open-source LLMs, *e.g.*, LLaMa [22], ChatGLM [17], and Alpaca [23]. Researchers have studied various methods for the applications of these popular LLMs. For example, many strategies are proposed to fine-tune pre-trained LLM models on specific tasks [24], which can further improve their capabilities in specific domains. Besides, some efforts have been made to enhance the quality of the generated content of LLMs, *e.g.*, generating more diverse and creative text while maintaining coherence and fluency [25]. Overall, recent developments of LLMs cover a broad range of topics, including model architecture [17], training methods [26], fine-tuning strategies [27], as well as ethical considerations [21]. All these methods aim to enhance the understanding capabilities of LLMs for real-world applications. However, these powerful LLM models still have some shortcomings. One notable limitation of LLMs is their lack of a strong long-term memory, which hinders their ability to process lengthy context and retrieve relevant historical information.

2.2 Long-term Memory

Numerous efforts have been conducted to enhance the memory capabilities of LLMs. One approach is to utilize memory-augmented networks (MANNs) [28], such as Neural Turing Machines (NTMs) [29], which is designed to utilize more context information for dialogue. In general, MANNs are proposed with an external memory cache via the storage and manipulation of information, which can well handle tasks of long-term period by interacting with memory. In addition, many studies focused on long-term conversations [30–33]. For example, Xu *et al.* [30] introduced a new English dataset consisting of multi-session human-human crowdworker chats for long-term conversations. Zhong *et al.* [32] proposed a MemoryBank mechanism inspired by Ebbinghaus' forgetting curve theory. However, these methods still face some great challenges to achieve a reliable and adaptable long-term memory mechanism for Language and Learning Models (LLMs). Concretely, these methods only considered storing the raw dialogue text, requiring repeated reasoning of the LLM agent over the same history. Besides, these models need to calculate pairwise similarity for recalling relevant information, which is time-consuming for the long-term interactions.

3 METHODOLOGY

In this section, we first introduce the overall workflow of our proposed framework. Then we provide a detailed description for each stage of TiM, involving storage for memory cache, organization principle for memory updating, and retrieval for memory recalling.

3.1 Framework Overview

Given a sequence of conversation turns, each turn is denoted by a tuple (Q, R), representing the user's query (Q) and the agent's response (R) at that specific turn. The main objective is to generate a more accurate response R_y like a human for a new coming query Q_x, while remembering the contextual information of historical conversation turns. The proposed TiM allows the agent to process

long-term conversation and retain useful historical information after multiple conversations with the user.

3.1.1 ***Main Components.*** As illustrated in Figure 2, our TiM comprises the following components, working together to provide more accurate and coherent responses for long-term conversation:

- Agent $\mathcal{A}$: $\mathcal{A}$ is a pre-trained LLM model to facilitate dynamic conversations, such as ChatGPT [1] and ChatGLM [17].
- Memory Cache $\mathcal{M}$: $\mathcal{M}$ a continually growing hash table of key-value pairs, where key is the hash index and value is a single thought. More details of $\mathcal{M}$ can refer to Section 3.2. To be clear, $\mathcal{M}$ supports varying operations as shown in Table 1.
- Hash-based Mapping $\mathbf{F}(\cdot)$: Locality-sensitive Hashing is introduced to quickly save and find the relevant thoughts in $\mathcal{M}$.

3.1.2 ***Workflow.*** Overall framework is divided into two stages:

- **Stage-1: Recall and Generation.** Given a new question from the user, LLM agent $\mathcal{A}$ retrieves relevant thoughts for generating accurate responses. Since we save the self-generated reasoning thoughts as external memory, this stage can directly recall and answer the question without repeated reasoning over the raw historical conversation text.
- **Stage-2: Post-think and Update.** After answering the question, we let the LLM agent post-think upon Q-R pair and insert the newly self-generated reasoning thoughts into memory cache $\mathcal{M}$.

3.2 Storage for Memory Cache

3.2.1 Thoughts-based System. TiM's storage system $\mathcal{M}$ aims to save the knowledge of AI-user interactions via self-generated inductive thoughts (Definition 3.1) upon the conversations. Each piece of thought T is stored in the format of the tuple (H_{idx}, T), where H_{idx} is the hash index obtained by hash function $\mathbf{F}(T)$. This hash-based storage not only aids in quick memory retrieval but also facilitates the memory updating, providing a detailed index of historical thoughts.

Definition 3.1. **Inductive Thought.** The inductive thought is defined as the text which contains the relation between two entities, *i.e.*, satisfying a relation triple (E_h, r_i, E_t). E_h is head entity connected with tail entity E_t via the relation r_i, where $i \in [0, N]$ and N is the relation number. Conceptually, $R_h = \{r_1, \cdots, r_N\}$ consists of all the one-hop relations for the entity E_h.

The main challenge of utilizing inductive thoughts for LLM is obtaining high-quality sentences matching relation triples. Here we provide two kinds of solutions to obtain inductive thoughts: (1) pre-trained model for open information extraction, such as OpenIE [34]; (2) In-context learning with few-shot prompts based on LLM. In this work, we utilize the second solution, *i.e.*, utilizing LLM Agent $\mathcal{A}$ to generate inductive thoughts, as shown in Figure 3.

3.2.2 Hash-based Storage. We aim to save inductive thoughts into the memory following a certain rule, *i.e.*, similar thoughts should be stored in the same group in the memory for efficiency. To this end, we adopt a hash table as the architecture of TiM's storage system, where similar thoughts are assigned with the same hash index.

Given a query, we propose to quickly search its nearest thoughts in a high-dimensional embedding space, which can be solved by the locality-sensitive hashing (LSH) method. The hashing scheme

Prompt for Generating Thoughts

Given the following question and response pairs, please extract the relation (subject, relation, object) with corresponding text:

Example 1.
Input:
Question: Do you have any company recommendations for me?
Response: I recommend Google.
Output:
(Company, Recommended, Google).
Recommended company is Google.

Example 2.
Input:
Question: Which City is the capital of China?
Response: Beijing.
Output:
(China, Capital, Beijing).
The capital of China is Beijing.

Input:
Question: Do you have any book recommendations for me?
Response: I recommend "The Little Prince".
Output:

Figure 3: An example of prompts for generating thoughts.

of LSH is to assign each d-dimension embedding vector $x \in \mathbf{R}^d$ to a hash index $\mathbf{F}(x)$, where nearby vectors get the same hash index with higher probability. We achieve this by exploiting a random projection as follows:

$$\mathbf{F}(x) = \arg\max\left([xR; -xR]\right), \tag{1}$$

where R is a random matrix of size $(d, b/2)$ and b is the number of groups in the memory. $[u; v]$ denotes the concatenation of two vectors. This LSH method is a well known LSH scheme [16] and is easy to implement. Figure 2 shows a schematic exhibition of TiM's storage system based on LSH.

3.3 Retrieval for Memory Recalling

Built on the memory storage, the memory retrieval operates a two-stage retrieval task for the most relevant thoughts, *i.e.*, LSH-based retrieval followed by similarity-based retrieval. The paradigm involves the following detailed points.

- **Stage-1: LSH-based Retrieval.** For a new query Q, we first obtain its embedding vector x based on LLM agent. Then LSH function (*i.e.*, Eq. 1) can produce the hash index of the query. This hash index also indicates the its nearest group for similar thoughts in the memory cache according to the property of LSH.
- **Stage-2: Similarity-based Retrieval.** Within the nearest group, we calculate the pairwise similarity between the query and each piece of thought in the group. Then top-k thoughts are recalled as the relevant history for accurately answering the query. It

Prompt for Forgetting Thoughts

Given the following thoughts, please remove the counterfactual thoughts or contradictory thoughts:

Example 1.
Input:
The capital of China is Beijing.
The capital of China is Shanghai.
The capital of the United States is Washington.
The capital of the United States is New York.
Output:
The capital of China is Beijing.
The capital of the United States is Washington.

Example 2.
Input:
Michael likes to play football.
Michael does not like to play football.
James likes to swim.
Mary likes to read books.
Output:
James likes to swim.
Mary likes to read books.

Input:
[A group of thoughts]
Output:

Figure 4: An example of prompts for forgetting thoughts.

Prompt for Merging Thoughts

Given the following thoughts, please merge the similar thoughts with the same entity:

Example 1.
Input:
John works as an actor.
John works as a director.
John works as a writer.
Mike works as a teacher.
Output:
John works as an actor, a director, and a writer.
Mike works as a teacher.

Example 2.
Input:
Michael likes to play football.
Michael likes to play basketball.
James likes to swim.
Mary likes to read books.
Output:
Michael likes to play football and basketball.
James likes to swim.
Mary likes to read books.

Input:
[A group of thoughts]
Output:

Figure 5: An example of prompts for merging thoughts.

should be noticed that pairwise similarity is calculated within a group rather than the whole memory cache, which can achieve more efficient retrieval than previous memory mechanisms.

3.4 Organization for Memory Updating

With the above-discussed memory storage and retrieval, the long-term memory capability of LLMs can be well enhanced. Motivated by the human memory, there needs some organization principles based on the well-established operations for dynamic updates and evolution of the thoughts, *e.g.*, insert new thoughts, forget less important thoughts, and merge repeated thoughts, which can make the memory mechanism more natural and applicable.

Beginning with the architecture of the storage for memory cache, TiM adopts the hash table to store the self-generated thoughts, where each hash index corresponds a group containing similar thoughts. Within same group, TiM supports the following operations to organize the thoughts in the memory:

- **Insert**, *i.e.*, store new thoughts into the memory. The prompt for generating thoughts is shown in Figure 3.
- **Forget**, *i.e.*, remove unnecessary thoughts from the memory, such as contradictory thoughts. The prompt of this operation is shown in Figure 4.
- **Merge**, *i.e.*, merge similar thoughts in the memory, such as thoughts with the same head entity. The prompt of this operation is shown in Figure 5.

Table 1: Organization comparisons between previous memory mechanisms and ours. KG denotes the knowledge graph and Q-R denotes the question and response pairs.

Method	Content	LLM-agnostic	Insert	Forget	Merge
SCM [33]	Q-R	✓	✓	✗	✗
RelationLM [7]	KG	✗	✓	✗	✗
LongMem [10]	Token	✗	✓	✗	✗
MemoryBank [32]	Q-R	✓	✓	✓	✗
Ours (TiM)	Thoughts	✓	✓	✓	✓

3.5 Parameter-efficient Tuning

We adopt a computation-efficient fine-tuning approach called Low-Rank Adaptation (LoRA) [27] for the scenarios with limited computational resources. LoRA [27] optimizes pairs of rank-decomposition matrices while keeping the original weights frozen, which can effectively reduce the number of trainable parameters. Specifically, considering a linear layer defined as $y = Wx$, LoRA fine-tunes it according to $y = Wx + BAx$, where $W \in \mathbf{R}^{d\times k}$, $B \in \mathbf{R}^{d\times r}$, $A \in \mathbf{R}^{r\times k}$, and $r \ll \min(d; k)$. Essentially, this fine-tuning stage can adapt LLMs to multi-turn conversations, providing appropriately and effectively response to users. For all experiments, we set LoRA rank r as 16 and train the LLM models for 10 epochs.

3.6 Insightful Discussion

Here we make a summary for previous memory mechanisms and our method in Table 1, including memory content, LLM-agnostic, and organization operations. There are several important observations from Table 1: (1) Previous memory mechanisms only save raw conversation text (Q-R pairs) as the memory, which requires repeated reasoning over the history. Our method maintains thoughts in the memory cache and can directly recall them without repeated reasoning. (2) Previous memory mechanisms only support simple read and write (insert) operations, while our method provides more manipulate way for the memory. (3) Some previous memory mechanisms store the tokens in the memory, which requires adjusting LLM architecture (LLM-aware) for applications. Our method is deigned as a LLM-agnostic module, which can be easily combined with other LLMs.

4 EXPERIMENT

4.1 Experimental Settings

4.1.1 Dataset. Three datasets are used to demonstrate the effectiveness of the proposed method.

- **KdConv**: KdConv is a Chinese multi-domain knowledge-driven conversation benchmark [35] grounding the topics to knowledge graphs, which involves 4.5K conversations and 86K utterances from three domains (film, music, and travel). The average turn number is 19.
- **Generated Virtual Dataset (GVD)**: GVD is a long-term conversation dataset [32] involving 15 virtual users (ChatGPT) over 10 days. Conversations are synthesized using pre-defined topics, including both English and Chinese languages. For the test set, [32] manually constructed 194 query questions (97 in English and 97 in Chinese) to evaluate whether the LLM could accurately recall the memory and produce the appropriate answers.
- **Real-world Medical Dataset (RMD)**: To evaluate the effectiveness of the proposed memory mechanism in the real-world scenarios, we manually collect and construct a dataset containing 1,800 conversations for medical healthcare consumer. For the test set, 80 conversations are used to evaluate whether the LLM could provide the accurate diagnosis.

4.1.2 LLM. We integrate two powerful LLMs to demonstrate the effectiveness of the proposed TiM mechanism. These LLMs originally lack long-term memory and specific adaptability to the long-term conversations. The detailed introduction of these LLMs are follows.

- **ChatGLM** [17]: ChatGLM is an open-source bilingual language model based on the General Language Model (GLM) framework [17]. This model contains 6.2 billion parameters with specific optimization, involves supervised fine-tuning, feedback bootstrap, and reinforcement learning with human feedback.
- **Baichuan2** [36]: Baichuan2 is an open-source large-scale multilingual language model containing 13 billion parameters, which is trained from scratch on 2.6 trillion tokens. This model excels at dialogue and context understanding.

4.1.3 Baseline. One baseline is to answer questions without using any memory mechanism. Another baseline is SiliconFriend [32], a classical memory mechanism, which can store the raw text into the memory and support reading operation.

4.1.4 Evaluation Protocol. Following [32], three metrics are adopted to evaluate the performance of the proposed method.

- **Retrieval Accuracy** evaluates whether the relevant memory is successfully recalled (labels: {0: no; 1: yes}).
- **Response Correctness** evaluates if correctly answering the probing question (labels: {0: wrong; 0.5: partial; 1 : correct}).
- **Contextual Coherence** evaluates whether the response is naturally and coherently generated, *e.g.*, connecting the dialogue context and retrieved memory (labels: {0: not coherent; 0.5: partially coherent; 1: coherent}).

To be fair, during evaluation, the prediction results of all LLMs are firstly shuffled, ensuring the human evaluator does not know which LLM the results come from. Then the final evaluation results are obtained by the human evaluation.

4.2 Comparison Results

4.2.1 Results on GVD dataset. We evaluate our method on both English and Chinese test sets of GVD dataset. The following insights are observed from Table 2: (1) Compared with SiliconFriend [32], our method exhibits superior performance for all metric, especially for the contextual coherence, indicating the effectiveness of TiM mechanism. (2) TiM delivers better results on both languages. The performance improvement on Chinese is larger than English, which may be attributed to the abilities of the LLMs.

4.2.2 Results on KdConv dataset. Table 2 illustrates the comparison results on KdConv dataset. We evaluate 2 different LLMs with TiM over different topics (film, music, and travel). As shown in Table 2, it is observed that our method can obtain best results across all topics. Our method can achieve high retrieval accuracy to recall the relevant thoughts. When without the memory mechanism, these LLMs usually exhibit lower response correctness due to lack of long-term memory capability, while TiM can well eliminate such negative issue. Furthermore, TiM can also help to improve the contextual coherence of the response.

4.2.3 Results on RMD dataset. Table 2 reports the comparison results on RMD dataset, which contains the realistic conversations between the doctors and patients. As shown in Table 2, our method can improve the overall response performance for the real-world medical conversations. In detail, using TiM, both ChatGLM and Baichuan2 can improve their capability for long-term conversations, *i.e.*, significant improvements on the response correctness and the contextual coherence. The main reason is that TiM is more similar to the workflow of human memory, which can enhance the ability of LLMs to produce more human-like responses.

4.3 More Analysis

4.3.1 Retrieval Time. We report the comparison results of retrieval time. The baseline is to calculate pairwise similarity between the question and the whole memory, which is utilized as the default retrieval way for most previous mechanisms. For both baseline and our method, the memory length is as 140 and the memory context is fixed. Table 2 shows the time cost for making a single retrieval. It

Table 2: Comparison Results on Three Datasets. Top-5 thoughts are recalled from the memory cache.

Dataset	LLM	Language/Topic	Memory	Retrieval Accuracy	Response Correctness	Contextual Coherence
GVD	ChatGLM	English/Open	SiliconFriend	0.809	0.438	0.680
			TiM (Ours)	**0.820**	**0.450**	**0.735**
		Chinese/Open	SiliconFriend	0.840	0.418	0.428
			TiM (Ours)	**0.850**	**0.605**	**0.665**
Kdconv	ChatGLM	Chinese/Film	✗	-	0.657	0.923
			TiM (Ours)	**0.920**	**0.827**	**0.943**
		Chinese/Music	✗	-	0.666	0.910
			TiM (Ours)	**0.970**	**0.826**	**0.926**
		Chinese/Travel	✗	-	0.735	0.906
			TiM (Ours)	**0.940**	**0.766**	**0.912**
	Baichuan2	Chinese/Film	✗	-	0.360	0.413
			TiM (Ours)	**0.913**	**0.743**	**0.870**
		Chinese/Music	✗	-	0.253	0.283
			TiM (Ours)	**0.900**	**0.710**	**0.780**
		Chinese/Travel	✗	-	0.207	0.280
			TiM (Ours)	**0.833**	**0.757**	**0.807**
RMD	ChatGLM	Chinese/Medical	✗	-	0.806	0.893
			TiM (Ours)	**0.900**	**0.843**	**0.943**
	Baichuan2	Chinese/Medical	✗	-	0.506	0.538
			TiM (Ours)	**0.873**	**0.538**	**0.663**

Table 3: Comparisons of Retrieval Time. Baseline calculates pairwise similarity between the question and memory.

Method	Retrieval Time (ms)
Baseline	0.6287
Ours (TiM)	0.5305

is observed that our method can reduce about 0.1 ms retrieval time compared with baseline method.

4.3.2 Top-k Recall. We report the retrieval accuracy using different values of k on Kdconv dataset (Travel). As shown in Figure 6, top-1 retrieval accuracy is higher than 0.7. The overall retrieval accuracy is improved with increasing value of k, where top-10 can achieve 0.973 retrieval accuracy. Besides, as shown in Table 2, top-5 recall can significantly improve the performance of existing LLMs for long-term conversations.

4.4 Industry Application

In this section, based on the ChatGLM and TiM, we develop a medical agent (named TiM-LLM) in the context of patient-doctor conversations (as shown in Figure 7). Note that TiM-LLM is only an **auxiliary** tool for the clinical doctors to give treatment options and medical suggestions for patients' needs.

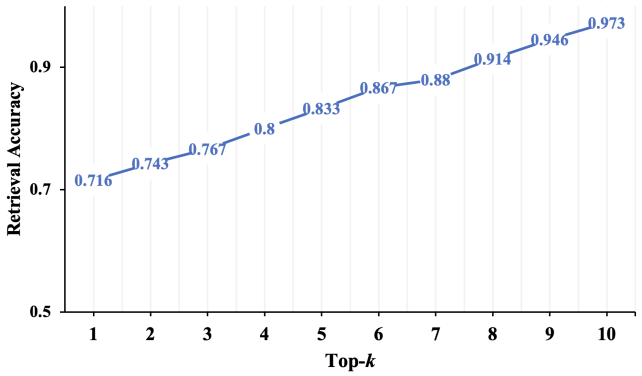

Figure 6: Tendency of retrieval accuracy with different k.

Figure 7 illustrates a real-world conversation between a patient and a doctor, where the clinical diagnosis results are given by the medical agent with and without TiM. As shown in Figure 7, without TiM, the medical agent may struggle to recall previous symptoms,

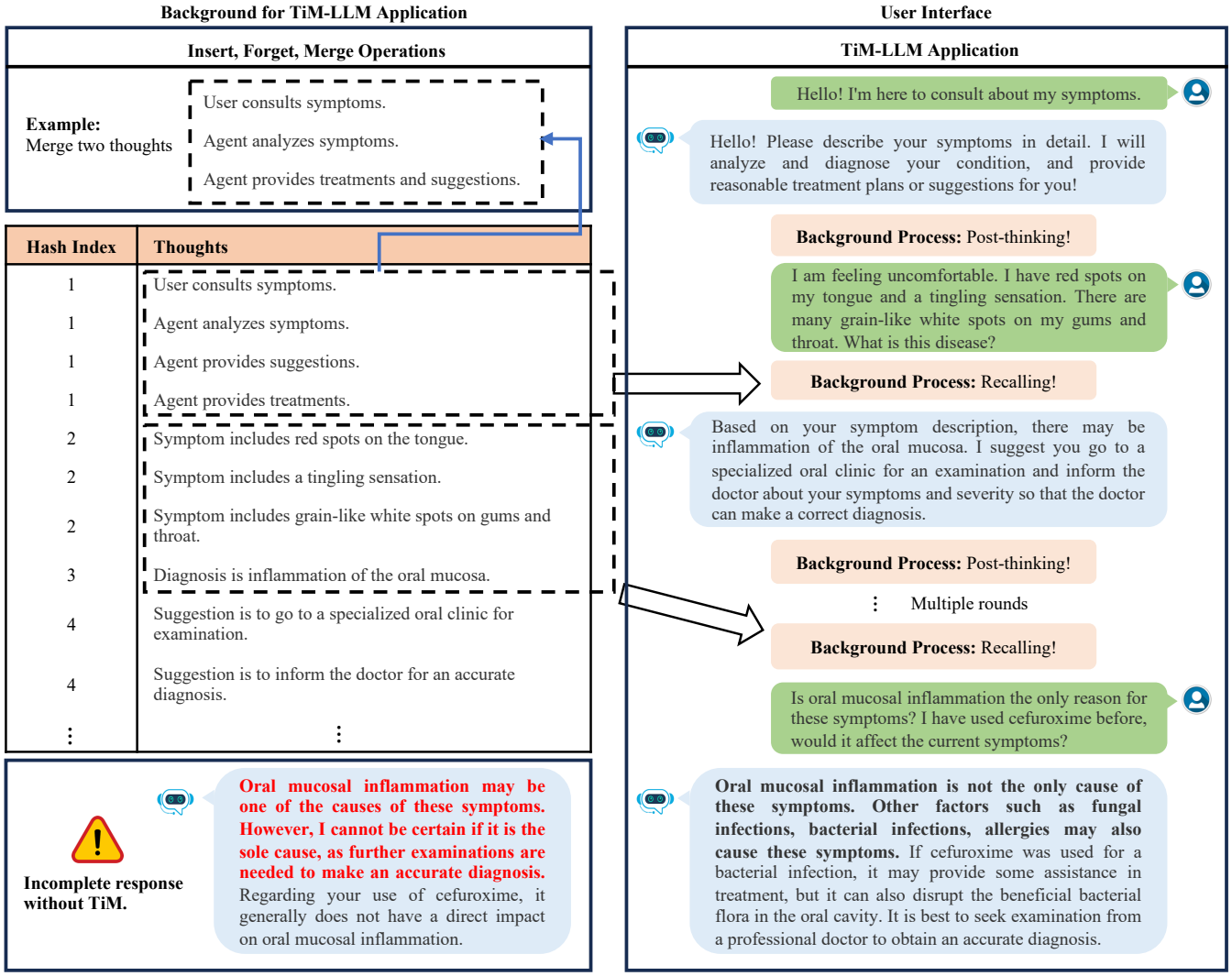

Figure 7: The application of TiM. The left is the background of TiM-LLM application and the right is user interface.

resulting in incomplete or incorrect assessments (red part), *i.e.*, the agent has forgotten previous symptoms so it is uncertain whether oral mucosal inflammation is the only cause. Assisted by TiM, the medical agent can recall relevant symptoms and make a comprehensive understanding of a patient's diseases. Thus it provide accurate diagnosis and treatment (**bold part**).

5 CONCLUSION

In this work, we propose a novel memory mechanism called TiM to address the issue of biased thoughts in Memory-augmented LLMs. By storing historical thoughts in an evolved memory, TiM enables LLMs to recall relevant thoughts and incorporate them into the conversations without repeated reasoning. TiM consists of two key stages: recalling thoughts before generation and post-thinking after generation. Besides, TiM works with the several basic principles to organize the thoughts in memory, which can achieve dynamic updates of the memory. Furthermore, we introduce Locality-Sensitive Hashing into TiM to achieve efficient retrieval for the long-term conversations. The qualitative and quantitative experiments conducted on real-world and simulated dialogues demonstrate the significant benefits of equipping LLMs with TiM. Overall, TiM is designed as an approach to improve the quality and consistency of responses for long-term human-AI interactions.

REFERENCES

[1] OpenAI. Chatgpt. 2022.

[2] OpenAI. Gpt-4 technical report. 2023.

[3] Hongyang Yang, Xiao-Yang Liu, and Christina Dan Wang. Fingpt: Open-source financial large language models. *arXiv preprint arXiv:2306.06031*, 2023.

[4] Hongbo Zhang, Junying Chen, Feng Jiang, Fei Yu, Zhihong Chen, Jianquan Li, Guiming Chen, Xiangbo Wu, Zhiyi Zhang, Qingying Xiao, et al. Huatuogpt, towards taming language model to be a doctor. *arXiv preprint arXiv:2305.15075*, 2023.

[5] Tyna Eloundou, Sam Manning, Pamela Mishkin, and Daniel Rock. Gpts are gpts: An early look at the labor market impact potential of large language models. *arXiv preprint arXiv:2303.10130*, 2023.

[6] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. *Advances in Neural Information Processing Systems*, 35:22199–22213, 2022.

[7] Qi Liu, Dani Yogatama, and Phil Blunsom. Relational memory-augmented language models. *Transactions of the Association for Computational Linguistics*, 10:555–572, 2022.

[8] Quentin Fournier, Gaétan Marceau Caron, and Daniel Aloise. A practical survey on faster and lighter transformers. *ACM Computing Surveys*, 55(14s):1–40, 2023.

[9] Jason Phang, Yao Zhao, and Peter J Liu. Investigating efficiently extending transformers for long input summarization. *arXiv preprint arXiv:2208.04347*, 2022.

[10] Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. *arXiv preprint arXiv:2306.07174*, 2023.

[11] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, et al. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning*, pages 2206–2240. PMLR, 2022.

[12] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, volume 1, page 2, 2019.

[13] Daniel Adiwardana, Minh-Thang Luong, David R So, Jamie Hall, Noah Fiedel, Romal Thoppilan, Zi Yang, Apoorv Kulshreshtha, Gaurav Nemade, Yifeng Lu, et al. Towards a human-like open-domain chatbot. *arXiv preprint arXiv:2001.09977*, 2020.

[14] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models. *arXiv preprint arXiv:2203.11171*, 2022.

[15] John Dunlosky and Janet Metcalfe. *Metacognition*. Sage Publications, 2008.

[16] Alexandr Andoni, Piotr Indyk, Thijs Laarhoven, Ilya Razenshteyn, and Ludwig Schmidt. Practical and optimal lsh for angular distance. *Advances in Neural Information Processing Systems*, 28, 2015.

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

[18] Biao Zhang, Barry Haddow, and Alexandra Birch. Prompting large language model for machine translation: A case study. *arXiv preprint arXiv:2301.07069*, 2023.

[19] Boyu Zhang, Hongyang Yang, Tianyu Zhou, Ali Babar, and Xiao-Yang Liu. Enhancing financial sentiment analysis via retrieval augmented large language models. *arXiv preprint arXiv:2310.04027*, 2023.

[20] Jiaxian Guo, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Boyang Li, Dacheng Tao, and Steven Hoi. From images to textual prompts: Zero-shot visual question answering with frozen large language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10867–10877, 2023.

[21] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

[22] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[23] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[24] Bonan Min, Hayley Ross, Elior Sulem, Amir Pouran Ben Veyseh, Thien Huu Nguyen, Oscar Sainz, Eneko Agirre, Ilana Heintz, and Dan Roth. Recent advances in natural language processing via large pre-trained language models: A survey. *ACM Computing Surveys*, 56(2):1–40, 2023.

[25] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*, 2019.

[26] Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. Pretraining language models with human preferences. In *International Conference on Machine Learning*, pages 17506–17533. PMLR, 2023.

[27] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[28] Lian Meng and Minlie Huang. Dialogue intent classification with long short-term memory networks. In *Natural Language Processing and Chinese Computing: 6th CCF International Conference, NLPCC 2017, Dalian, China, November 8–12, 2017, Proceedings 6*, pages 42–50. Springer, 2018.

[29] Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

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

[31] Xinchao Xu, Zhibin Gou, Wenquan Wu, Zheng-Yu Niu, Hua Wu, Haifeng Wang, and Shihang Wang. Long time no see! open-domain conversation with long-term persona memory. *arXiv preprint arXiv:2203.05797*, 2022.

[32] Wanjun Zhong, Lianghong Guo, Qiqi Gao, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory. *arXiv preprint arXiv:2305.10250*, 2023.

[33] Xinnian Liang, Bing Wang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. Unleashing infinite-length input capacity for large-scale language models with self-controlled memory system. *arXiv preprint arXiv:2304.13343*, 2023.

[34] Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, 2015.

[35] Hao Zhou, Chujie Zheng, Kaili Huang, Minlie Huang, and Xiaoyan Zhu. KdConv: A Chinese multi-domain dialogue dataset towards multi-turn knowledge-driven conversation. In *ACL*, 2020.

[36] Aiyuan Yang, Bin Xiao, Bingning Wang, Borong Zhang, Chao Yin, Chenxu Lv, Da Pan, Dian Wang, Dong Yan, Fan Yang, et al. Baichuan 2: Open large-scale language models. *arXiv preprint arXiv:2309.10305*, 2023.