MemEngine: A Unified and Modular Library for Developing Advanced Memory of LLM-based Agents

Zeyu Zhang
Renmin University of China
Beijing, China
zeyuzhang@ruc.edu.cn

Quanyu Dai
Huawei Noah's Ark Lab
Shenzhen, China
daiquanyu@huawei.com

Xu Chen[†]
Renmin University of China
Beijing, China
xu.chen@ruc.edu.cn

Rui Li
Renmin University of China
Beijing, China
lirui121200@ruc.edu.cn

Zhongyang Li
Huawei Technologies Ltd.
Beijing, China
lizhongyang6@huawei.com

Zhenhua Dong
Huawei Noah's Ark Lab
Shenzhen, China
dongzhenhua@huawei.com

Abstract

Recently, large language model based (LLM-based) agents have been widely applied across various fields. As a critical part, their memory capabilities have captured significant interest from both industrial and academic communities. Despite the proposal of many advanced memory models in recent research, however, there remains a lack of unified implementations under a general framework. To address this issue, we develop a unified and modular library for developing advanced memory models of LLM-based agents, called MemEngine. Based on our framework, we implement abundant memory models from recent research works. Additionally, our library facilitates convenient and extensible memory development, and offers user-friendly and pluggable memory usage. For benefiting our community, we have made our project publicly available at https://github.com/nuster1128/MemEngine.

CCS Concepts

• Information systems → Users and interactive retrieval; **• Software and its engineering → Software libraries and repositories**.

Keywords

Large Language Model, Autonomous Agent, Memory Mechanism, Information Retrieval, Library Resource

ACM Reference Format:
Zeyu Zhang, Quanyu Dai, Xu Chen[†], Rui Li, Zhongyang Li, and Zhenhua Dong. 2025. MemEngine: A Unified and Modular Library for Developing Advanced Memory of LLM-based Agents. In *Companion Proceedings of the ACM Web Conference 2025 (WWW Companion '25), April 28-May 2, 2025, Sydney, NSW, Australia.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3701716.3715299

† Corresponding author.

1 Introduction

With the rapid development of large language models (LLMs), LLM-based agents have been widely applied across various fields, due to their capabilities to perform complex tasks and fulfill different roles [11]. Among various internal modules, memory is one of the most critical components for agents, as it determines how they store historical data, reflect on existing knowledge, and recall useful information to support decision-making [13]. Specifically, in complex tasks, memory enables the recording of critical information in the past agent-environment interactions, and provides task-related experiences from previous trajectories. In role-playing and social simulations, it highlights the characteristic and personality of each role, allowing for distinctiveness among different roles.

Although some recent works have proposed various memory models for LLM-based agents, they are implemented under different pipelines and lack a unified framework. This inconsistency presents challenges for developers to attempt different models in their experiments. Moreover, many basic functions (such as retrieval and summarization) are duplicated across different models, and researchers often need to implement them repeatedly when developing new models. Besides, many academic models are tightly integrated with agents in a non-pluggable manner, making them difficult to apply across different agents.

In order to address the above problems, we develop a unified and modular library named **MemEngine**, which facilitates the development of advanced memory models for LLM-based agents. The primary features of our library are summarized as follows:

Unified and Modular Memory Framework. We propose a unified memory framework composed of three hierarchical levels to organize and implement existing research models under a general structure. The lowest level comprises memory functions, implementing basic functions (*e.g.,* retrieval) as foundational supports for different memory operations. The intermediate level encompasses memory operations, constituting basic operations (*e.g.,* memory recall) to construct different memory models. The highest level involves memory models, implementing various existing research models (*e.g.,* MemoryBank [14]) that can be conveniently applied in different agents. All these three levels are modularized inside our framework, where higher-level modules can reuse lower-level modules, thereby improving efficiency and consistency in implementation. Besides, we provide a configuration module for easy modification of hyper-parameters and prompts at different levels. We also implement a utility module to conveniently save and demonstrate memory contents.

Abundant Memory Implementation. Based on our unified and modular framework, we implement a wide range of memory models from recent research works, many of which are widely applied in diverse applications. All of these models can be easily switched and tested under our framework, with different configurations of hyper-parameters and prompts that can be adjusted for better application across various agents and tasks.

Convenient and Extensible Memory Development. Based on our modular memory operations and memory functions, researchers can conveniently develop their own advanced memory models. They can also extend existing operations and functions to develop their own ones. To better support researchers' development, we provide detailed instructions and examples in our document to guide the customization.

Pluggable and User-friendly Memory Usage. Our library offers multiple deployment options to empower LLM-based agent with powerful memory capabilities. Besides, we provide various memory usage modes, including default, configurable, and automatic modes. Moreover, our memory modules are pluggable and can be easily utilized across different agent frameworks. Our library is also compatible with some prominent frameworks of LLM-based agents, such as AutoGPT. These features collectively contribute to making our library more user-friendly.

In summary, MemEngine is the first library that implements a wide variety of memory models from research works under a unified and modular framework, facilitating both convenient development and ease of use. To further benefit the community, we have made our project publicly available at Github repository[1]. Additionally, we have also organized a comprehensive documentation[2] for both application and development purposes.

2 Comparison with Relevant Libraries

Several existing libraries can also empower memory capabilities for LLM-based agents, including (1) memory modules integrated into agent libraries, and (2) independent memory libraries. Specifically, AutoGen[3], MetaGPT [3], CAMEL [4], AgentScope [2], LangChain[4], AgentLite [5], CrewAI [5], AutoGPT [6], and AgentVerse [1] are prominent open-source libraries for building LLM-based AI agent systems. Besides, Memary[7] is an open-source library to empower AI agents with memory for continuous improvement. Cognee[8] provides a scalable and modular pipeline to interconnect and retrieve previous information for AI applications. Mem0[9] offers an artificial memory layer for LLM-based agents and assistants to make them personalized. Agentmemory[10] implements easy-to-use memory for LLM-based agents with document search and more.

[1]https://github.com/nuster1128/MemEngine
[2]https://memengine.readthedocs.io/en/latest/
[3]https://github.com/microsoft/autogen
[4]https://github.com/langchain-ai/langchain
[5]https://github.com/crewAIInc/crewAI
[6]https://github.com/Significant-Gravitas/AutoGPT
[7]https://github.com/kingjulio8238/Memary
[8]https://github.com/topoteretes/cognee
[9]https://github.com/mem0ai/mem0
[10]https://github.com/ai16z/agentmemory

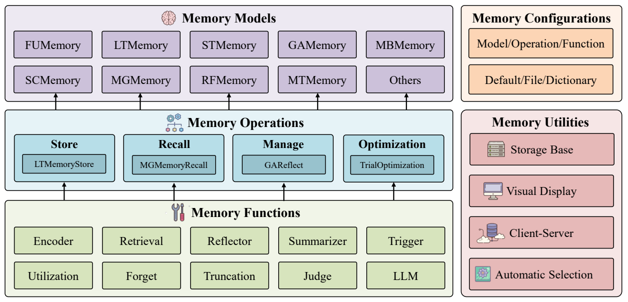

Figure 1: An overview framework of MemEngine Library.

We present a comprehensive comparison between MemEngine and relevant libraries in Table 1. While most of these libraries offer plug-and-play memory components to store and recall information for LLM-based agents, few of them implement advanced memory operations like reflection and optimization. Moreover, compared with other libraries, as a major contribution, MemEngine implements comprehensive research models under a unified framework, along with providing modular operations and functions that assist researchers in customizing advanced memory models.

3 MemEngine Library

3.1 Overview

The framework of our library is present in Figure 1. The lowest level implements basic functions as standard support. The intermediate level comprises memory operations for fundamental processes. The highest level features various memory models from previous research works. Additionally, we provide a configuration module and a utility module to facilitate convenient development and usage.

3.2 Memory Models

We implement a variety of memory models from recent research works under a general structure, allowing seamless switching among them. Specifically, these models are implemented with the interfaces including *reset*, *store*, *recall*, *manage*, and *optimize*.

The implemented memory models are described as follows:
- **FUMemory** (Full Memory): Naively concatenates all the information into a single string, also known as long-context memory.
- **LTMemory** (Long-term Memory): Calculates semantic similarities with text embeddings to retrieve the most relevant information.
- **STMemory** (Short-term Memory): Maintains the most recent information and concatenates them into a single string.
- **GAMemory** (Generative Agents [7]): A pioneer memory model with weighted retrieval combination and self-reflection mechanism.
- **MBMemory** (MemoryBank [14]): A multi-layered memory model with dynamic summarization and forgetting mechanism.
- **SCMemory** (SCM [10]): A self-controlled memory model that can recall minimal but necessary information for inference.
- **MGMemory** (MemGPT [6]): A hierarchical memory model that treats the memory system as an operation system.
- **RFMemory** (Reflexion [9]): A prominent memory model that can learn to memorize from previous trajectories by optimization.
- **MTMemory** (MemTree [8]): A dynamic memory model with a tree-structured semantic representation to organize information.

All of these memory models are implemented by combining various memory operations, and we make some reasonable adaptations

Table 1: Comparison with relevant open-source libraries. We focus on both memory modules integrated in prominent agent libraries, and independent memory libraries. Besides the libraries mentioned below, AutoGPT and AgentVerse do not specify their memory support in their library. AutoGen supports MemGPT, Mem0, and Zep as extensions.

Memory Features	Memory Integrated in Agent Libraries						
	AutoGen	MetaGPT	CAMEL	AgentScope	Langchain	AgentLite	CrewAI
Plug-and-play Integration	✓	✗	✓	✓	✓	✓	✓
Basic Read and Write Support	✓	✓	✓	✓	✓	✓	✓
Reflection and Optimization Support	✓	✗	✗	✗	✗	✗	✗
Comprehensive Default Models	✗	✗	✗	✗	✗	✗	✗
Advanced Model Customization	✗	✗	✓	✗	✗	✗	✗
Memory Features	**Independent Memory Libraries**						
	Memary	**Cognee**	**Mem0**	**Agentmemory**	**MemoryScope**	**Zep**	**MemEngine (Ours)**
Plug-and-play Integration	✓	✓	✓	✓	✓	✓	✓
Basic Read and Write Support	✓	✓	✓	✓	✓	✓	✓
Reflection and Optimization Support	✗	✗	✗	✗	✓	✗	✓
Comprehensive Default Models	✗	✗	✗	✗	✗	✗	✓
Advanced Model Customization	✗	✗	✗	✗	✓	✗	✓

in their implementations. Further details can be found in our project documentation and source code.

3.3 Memory Operations

We implement various types of memory operations for constructing memory models, including store, recall, manage, and optimize.

Memory Store Operation intends to receive observations from the environment, processing them to obtain memory contents and adding them into memory storage. Another critical function of the memory store operation is to establish foundations for the memory recall operation, such as creating indexes and summaries.

Memory Recall Operation intends to obtain useful information to assist agents in their decision-making. Typically, the input is a query or observation representing the current state of agents. Some human-like agents may also endow the memory recall operation with certain retention characteristics like human memory.

Memory Manage Operation intends to reorganize existing information for better utilization, such as memory reflection. Besides, simulation-orientated agents may be equipped with a forgetting mechanism during the memory manage operation.

Memory Optimize Operation intends to optimize the memory capability of LLM-based agents by using extra trials and trajectories. It enables agents to extract meta-insight from historical experiences, which can be considered as a learn-to-memorize procedure.

Different memory models may share common memory operations or implement their unique operations according to their requirements. For example, *MTMemory* and *LTMemory* share the common memory recall operation *LTMemoryRecall*, while MTMemory has its own memory store operation *MTMemoryStore* to implement the tree-structured information update.

3.4 Memory Functions

We implement various types of memory functions to support the construction of memory operations, which are listed as follows.

Encoder can transfer textual messages into embeddings to represent in latent space by pre-trained models, such as E5 [12].

Retrieval is utilized to find the most useful information for the current query or observation, commonly by different aspects like semantic relevance, importance, recency, and so on.

Reflector aims to draw new insights in a higher level from existing information, commonly for reflection and optimization.

Summarizer can summarize texts into a brief summary, which can decrease the lengths of texts and emphasize critical points.

Trigger is designed to call functions or tools in extensible manners. One significant instance is utilizing LLMs to determine which function should be called with certain arguments.

Utilization aims to deal with several different parts of memory contents, formulating this information into a unified output.

Forget is typically applied in simulation-oriented agents, such as role-playing and social simulations. It empowers agents with features of human cognitive psychology, aligning with human roles.

Truncation helps to formulate memory contexts under the limitations of token numbers by certain LLMs.

Judge intends to assess given observations or intermediate messages on certain aspects. For example, *GAMemory* judges the importance score of each observation when stored into memory, as an auxiliary criterion for the retrieval process.

LLM provides a convenient interface to utilize the powerful capability of different large language models.

All these memory functions are designed to conveniently construct different memory operations for various models. For example, *GAMemoryStore* utilizes *LLMJudge* to provide the importance score on each observation.

3.5 Memory Configurations

In order to improve convenience for developers and facilitate parameter tuning by researchers, we have developed a unified memory configuration module. First, we design a hierarchical memory configuration module corresponding to our three-level memory framework, enabling adjustments to both hyper-parameters and prompts within the memory models. Second, we provide a comprehensive set of default hyper-parameters and prompts, where developers and researchers can adjust only the specific parts without altering others. Finally, our configuration supports both statistic manners (*e.g.,* files) and dynamic manners (*e.g.,* dictionaries).

3.6 Memory Utilities

We implement extra utilities as auxiliary components, which are loosely coupled with the above modules. We implement a storage module as a database to retain the contents of information. We also provide a display module to visualize the specific contents within the memory. Besides, we offer a client module to utilize MemEngine through remote deployment on a server implemented by FastAPI. We also implement an automatic selector to assist developers to choose memory models with hyper-parameters for their own tasks.

4 Usage of MemEngine

In this section, we describe the usage of MemEngine to empower LLM-based agents with advanced memory capabilities. We divide our usage into two aspects: (1) utilize pre-implemented memory models, and (2) customize new memory models.

4.1 Utilize Pre-implemented Memory Models

There are two primary ways to deploy our library as follows.

Local Deployment. Developers can easily install our library in their Python environment via pip, conda, or from source code. Then, they can create memory modules for their agents, and utilize unified interfaces to perform memory operations within programs.

Remote Deployment. Alternatively, developers can install our library on computing servers and launch the service through a port. Then, they can initiate a client to perform memory operations by sending HTTP requests remotely from their lightweight devices.

After deployment, there are three modes available for utilizing pre-implemented memory models. In the default mode, the library provides a comprehensive set of hyper-parameters and prompts for default usage. In the configurable mode, developers can adjust certain hyper-parameters and prompts to better adapt to their applications. In the automatic mode, the library automatically selects the appropriate memory models, hyper-parameters, and prompts from the provided ranges, based on a specific task's criteria.

Additionally, our library offers compatibility with several well-known tools and frameworks, such as vllm and AutoGPT.

4.2 Customize New Memory Models

Our library provides support for developers to customize advanced memory models, offering comprehensive documentation and examples. There are three major aspects to customizing new models.

Customize Memory Functions. Researchers may need to implement new functions in their models to extend existing ones for additional features. For example, they may extend *LLMJudge* to design a *BiasJudge* for poisoning detection.

Customize Memory Operations. In developing a new model, customizing memory operations is crucial as they constitute the major pipelines of the detailed processes. For instance, a new memory recall operation can be implemented with a series of memory functions with advanced design and combination.

Customize Memory Models. By integrating newly customized memory operations with existing ones, researchers can design their models with various combinations to best suit their applications.

5 Conclusion

In this paper, we introduce a unified and modular library for developing advanced memory of LLM-based agents. In the future, we plan to provide support for multi-modal memory (such as visual and audio memory) to further enrich and enhance the memory capabilities of LLM-based agents for wider applications.

Acknowledgments

This work is supported in part by National Natural Science Foundation of China (No. 62422215 and No. 62472427), Beijing Outstanding Young Scientist Program NO.BJJWZYJH012019100020098, Intelligent Social Governance Platform, Major Innovation & Planning Interdisciplinary Platform for the "DoubleFirst Class" Initiative, Renmin University of China, Public Computing Cloud, Renmin University of China, fund for building world-class universities (disciplines) of Renmin University of China, Intelligent Social Governance Platform. This work is also sponsored in part by Huawei Innovation Research Program. We gratefully acknowledge the support from Mindspore[11], CANN (Compute Architecture for Neural Networks) and Ascend AI Processor used for this research.

References

[1] Weize Chen, Yusheng Su, Jingwei Zuo, Cheng Yang, Chenfei Yuan, Chi-Min Chan, Heyang Yu, Yaxi Lu, Yi-Hsin Hung, Chen Qian, et al. 2023. Agentverse: Facilitating multi-agent collaboration and exploring emergent behaviors. In *ICLR*.

[2] Dawei Gao, Zitao Li, Xuchen Pan, Weirui Kuang, Zhijian Ma, Bingchen Qian, Fei Wei, Wenhao Zhang, Yuexiang Xie, Daoyuan Chen, et al. 2024. Agentscope: A flexible yet robust multi-agent platform. *arXiv preprint arXiv:2402.14034* (2024).

[3] Sirui Hong, Xiawu Zheng, Jonathan Chen, Yuheng Cheng, Jinlin Wang, Ceyao Zhang, Zili Wang, Steven Ka Shing Yau, Zijuan Lin, Liyang Zhou, et al. 2023. Metagpt: Meta programming for multi-agent collaborative framework. *arXiv preprint arXiv:2308.00352* (2023).

[4] Guohao Li, Hasan Hammoud, Hani Itani, Dmitrii Khizbullin, and Bernard Ghanem. 2023. Camel: Communicative agents for" mind" exploration of large language model society. *NeurIPS* 36 (2023), 51991–52008.

[5] Zhiwei Liu, Weiran Yao, Jianguo Zhang, Liangwei Yang, Zuxin Liu, Juntao Tan, Prafulla K Choubey, Tian Lan, Jason Wu, Huan Wang, et al. 2024. AgentLite: A Lightweight Library for Building and Advancing Task-Oriented LLM Agent System. *arXiv preprint arXiv:2402.15538* (2024).

[6] Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G Patil, Ion Stoica, and Joseph E Gonzalez. 2023. Memgpt: Towards llms as operating systems. *arXiv preprint arXiv:2310.08560* (2023).

[7] Joon Sung Park, Joseph O'Brien, Carrie Jun Cai, Meredith Ringel Morris, Percy Liang, and Michael S Bernstein. 2023. Generative agents: Interactive simulacra of human behavior. In *UIST*. 1–22.

[8] Alireza Rezazadeh, Zichao Li, Wei Wei, and Yujia Bao. 2024. From Isolated Conversations to Hierarchical Schemas: Dynamic Tree Memory Representation for LLMs. *arXiv preprint arXiv:2410.14052* (2024).

[9] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. 2024. Reflexion: Language agents with verbal reinforcement learning. *NeurIPS* 36 (2024).

[10] Bing Wang, Xinnian Liang, Jian Yang, Hui Huang, Shuangzhi Wu, Peihao Wu, Lu Lu, Zejun Ma, and Zhoujun Li. 2023. Enhancing large language model with self-controlled memory framework. *arXiv preprint arXiv:2304.13343* (2023).

[11] Lei Wang, Chen Ma, Xueyang Feng, Zeyu Zhang, Hao Yang, Jingsen Zhang, Zhiyuan Chen, Jiakai Tang, Xu Chen, Yankai Lin, et al. 2024. A survey on large language model based autonomous agents. *FCS* 18, 6 (2024), 186345.

[12] Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2022. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533* (2022).

[13] Zeyu Zhang, Xiaohe Bo, Chen Ma, Rui Li, Xu Chen, Quanyu Dai, Jieming Zhu, Zhenhua Dong, and Ji-Rong Wen. 2024. A survey on the memory mechanism of large language model based agents. *arXiv preprint arXiv:2404.13501* (2024).

[14] Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. 2024. Memorybank: Enhancing large language models with long-term memory. In *AAAI*, Vol. 38. 19724–19731.

[11] https://www.mindspore.cn/