MemAgent: A cache-inspired framework for aligning LLM Web Agent with User Goal

Faria Huq [*,1] Nazmus Sakib [*,2] Protoy Barai [*,2] Sifat Ishmam Parisa [2] Abhik Bhattacharjee [2] Anindya Iqbal [2]

[1]Carnegie Mellon University [2]Bangladesh University of Engineering and Technology [*]These authors contributed equally to this work.

Motivation

Large Language Models (LLMs) used as web agents have great potential but aren't quite ready for everyday use. A major reason behind this is the misalignment between the user's expected outcome and the agent's corresponding action, induced by the lack of proper communication between them. To address this issue, we introduce **MemAgent**, a novel pipeline for LLM web agents, inspired by caching mechanisms to store task-specific information in a memory component. Our methodology involves two key steps: *Alignment* and *Execution*. We showcase the effectiveness of MemAgent in enabling LLM agents to query for additional information relevant to the current task and execute it with minimal user interaction, marking a significant improvement over current methods.

Dataset

Since there is no existing dataset of our required format, we synthetically process the existing dataset, Mind2Web [2] using `GPT-4-Turbo` [4]. We decompose each task description into two parts: 1) *abstract task description*: The simplified description of a given task and 2) *conversation pairs*: A subsequent pair of conversations between the agent and the user to ask for detailed information necessary to conduct the task. We use the conversation pairs to construct *memory bank*; which contains the variable types and values for each task. We took inspiration from intent and slot tagging to formulate the data structure [1]. Table 1 shows an example task description from the original Mind2Web dataset and its corresponding augmented data in our pipeline.

Mind2Web Task Description	From Birmingham (BHX) to Paris search for packages with casinos, restaurant, fitness and a free internet from April 7th to 11th.
GPT-4 Generated Description for MemAgent	<abs> Search for travel packages </abs> <Questions> <Q>What is the departure location for the travel package search? </Q> <A>Birmingham (BHX) </A> <mem>Departure Location: Birmingham (BHX) </mem> <Q>What is the destination for the travel package search? </Q> <A>Paris </A> <mem>Destination: Paris </mem> *(cont.)* </Questions>
Memory Bank Description	{Departure location: New York, Destination: Athens, *(cont.)* }

Table 1. An example data sample generated by our pipeline.

Method

We propose a two-step process for MemAgent: *Alignment* and *Execution*.

Alignment: During the alignment phase, the agent engages in a multi-turn conversation to inquire about missing information from the users. During the alignment phase, the agent is serving two key activities: 1) learning to infer missing information and pose relevant questions and 2) parsing user responses to understand the type and value of the information shared by the user. The parsed information will be used to construct a memory bank to be used during the next phase.

Execution: During the execution phase, the agent will utilize the constructed memory bank and the abstract task description to carry out the task. The agent takes the context HTML DOM as input and outputs the corresponding action at each step until the task is finished.

Main Takeaway

- We present **MemAgent**, a novel pipeline for LLM Web Agent inspired by caching mechanism.
- MemAgent follows a two-step process, *Alignment* and *Execution* which allows it to engage into a multi-turn conversation with users and work on underspecified task description otherwise impossible to execute.
- We show how MemAgent can be incorporated with RAG framework significantly reducing the length of conversation up to 28%.

Experiments

Alignment: We finetuned Vicuna$_{7B}$ model for the alignment phase.

Execution: We follow the similar pipeline, MindAct as shown in Mind2Web [2]. For candidate generation, we use the off-the-shelf DeBerta model they finetuned. For action prediction, we fine-tune both Flan-T5$_B$ and MindAct's Flan-T5$_B$ over our augmented dataset. MindAct's Flan-T5$_B$ performs the best among all outperforming the other models.

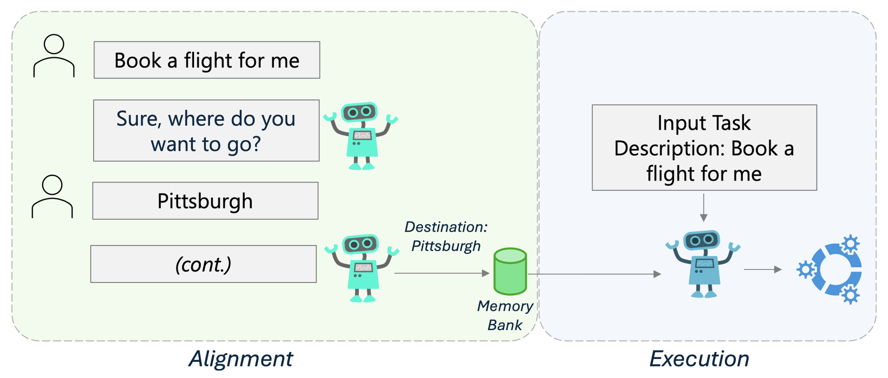

Figure 1. MemAgent Framework. In *Alignment*, it poses follow-up questions to users, storing their responses in the *memory bank*. In *Execution*, it leverages the memory bank to perform tasks.

Result

Alignment:We report the accuracy in terms of BertScore, BleuScore and turn of conversation. We also report the accuracy for fine-tuned Vicuna$_{7B}$ with pre-filled memory bank. The pre-filled memory bank helps the agent to reduce the turn of conversation by $27.04\% - 28.24\%$.

	Cross-Task			Cross-Website			Cross-Domain		
	BleuScore	BertScore	Avg. #	BleuScore	BertScore	Avg. #	BleuScore	BertScore	Avg. #
Vicuna$_{7B}$ (*/w prefilled mem bank*)	**47.17**	**0.95**	**2.52**	**50.07**	**0.96**	**2.44**	**50.41**	**0.96**	**2.67**
Vicuna$_{7B}$	37.64	0.91	3.49	39.31	0.92	3.40	39.87	0.94	3.66

Table 2. Results from Alignment Phase.

Execution: We follow the similar pipeline of MindAct as shown in Mind2Web [2]. For candidate generation, we use the off-the-shelf DeBerta model from Mind2Web. For action prediction, we fine-tune both Google Flan-T5-base (Flan-T5$_B$) and MindAct Flan-T5$_B$ over our augmented dataset.

	Cross-Task				Cross-Website				Cross-Domain			
	Ele. Acc.	Op. F1	Step SR	SR	Ele. Acc.	Op. F1	Step SR	SR	Ele. Acc.	Op. F1	Step SR	SR
Flan-T5$_B$ (fine-tuned)	34.02	71.82	31.49	1.98	27.88	65.53	24.33	**1.13**	29.63	65.73	27.01	**1.75**
MindAct Flan-T5$_B$	41.17	74.48	37.08	2.78	31.58	63.64	27.74	0.56	31.53	63.79	28.45	1.32
MindAct Flan-T5$_B$ (fine-tuned)	**42.36**	**76.17**	**39.17**	**3.57**	**33.1**	**68.71**	**29.22**	0.56	**33.04**	**67.69**	**30.02**	1.21

Table 3. Results from Execution Phase.

How MemAgent can be incorporated with Retrieval Augmented Generation (RAG) pipeline

The proposed memory bank in MemAgent can be easily incorporated with a vector database enabling retrieval augmented generation (RAG) [3]. It enables the agent to utilize task information from past executions and reduce the turn of conversation with users. Each time a new query is initiated, MemAgent queries the vector database and fetches the closest entry ($t_s \geq 0.70$). If found, the retrieved information is prepended to the user query in our framework.

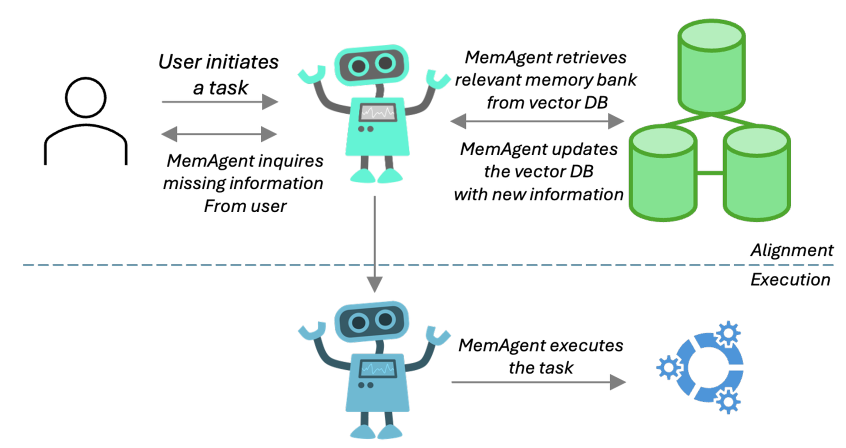

Figure 2. Incorporation of MemAgent with Vector database for Retrieval augmented Generation.

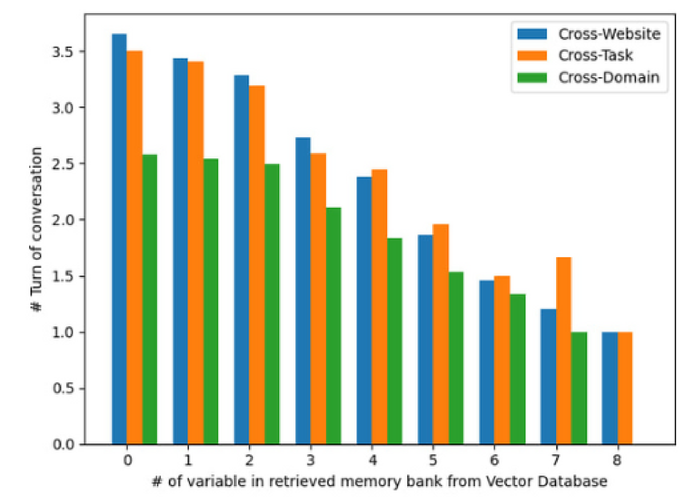

Figure 3. Statistics showing average count of conversation turn with respect of information present in the retrieved memory bank. With more information present in the memory bank, the conversation turn is significantly reduced without sacrificing the task completion accuracy.

Response from Vicuna$_{7B}$	*Response from Vicuna$_{7B}$ (/w prefilled memory bank)*
User: <mem_bank></mem_bank>find me a flight	User: <mem_bank>Destination City: Athens, Departure City: New York </mem_bank >find me a flight
Assistant: What is the departure city?	Assistant: How Many adults are travelling?
User: New York	User: 3
Assistant: <mem>Departure city: New York </mem>What is the destination city?	Assistant: <mem>Number of adults: 3 </mem>What is the departure date?
User: Athens	User: 22 April
Assistant: <mem>Destination city: Athens </mem> How many adults will be travelling?	Assistant: <mem>Departure date: 22 April </mem>What is the return date?
User: 3	User: 25 April
Assistant: <mem>Number of adults: 3</mem>	Assistant: <mem>Return date: 25 April </mem><Finish>
(cont.)	

Ongoing Work and Conclusions

In summary, we present a novel framework, MemAgent which is inspired by caching mechanism to learn user preference and perform repetitive tasks. Currently, we are benchmarking our method over a wide range of LLMs (including GPT-4, Llama-3, Mistral, CodeLlama etc.). In future, we aim to deploy our system as a plugin for public use.

References

[1] Wasi Uddin Ahmad, Jianfeng Chi, Tu Le, Thomas Norton, Yuan Tian, and Kai-Wei Chang. Intent classification and slot filling for privacy policies. *arXiv preprint arXiv:2101.00123*, 2021.

[2] Xiang Deng, Yu Gu, Boyuan Zheng, Shijie Chen, Samuel Stevens, Boshi Wang, Huan Sun, and Yu Su. Mind2web: Towards a generalist agent for the web. *arXiv preprint arXiv:2306.06070*, 2023.

[3] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

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

fhuq@cs.cmu.edu