International Journal of User-System Interaction 16 (3) 2023, 85-102

Enhanced Human-Machine Conversations by Long-Term Memory and LLMs

Dan-Constantin Dumitriu, Dragoş-Florin Sburlan

Faculty of Mathematics and Computer Science Ovidius University of Constanţa
Bd. Mamaia 124, 900527, Constanţa, Romania.
E-mail: dandumitriu33@gmail.comdsburlan@univ-ovidius.ro

Abstract. In this article, we explore the quality of the conversation between a human and a machine, the depth, relevance, and the "personalization" as opposed to an answer extracted from a pre-trained body of knowledge. To this end, we focus on the importance of memory augmenting a LLM, by designing a system whose answers in conversation retrieve information from learned details. LLM systems have indeed the concept of context that achieves this goal, however, the LLM concept of context is a component that is ephemeral and limited in size. In this article, we explore a Long-Term Memory (LTM) solution and a way to extract relevant details precisely. The concepts and information stored in this memory are built upon the prior conversations between the human and the system, illustrating the parent-child education paradigm. Although there is no unanimity regarding a standardized test that measures or quantifies how personal an artificial system is, our results empirically prove the method's feasibility (even if the proposed method has its limitations, the richness of possible extensions is worth studying).

Keywords: Personalized Human-Machine Conversation, Personalized AI, Personalized LLM, Long-Term Memory LLM, LLM Memory, LLM Education

DOI: 10.37789/ijusi.2023.16.3.2

1. Introduction

Nowadays there is a lot of enthusiasm and engagement in what concerns human-like conversational agents which are endowed with the ability to support a conversation with a human user. While evaluating such agents, two concepts emerge straight away: the Medium of the Conversation and the Quality of the Conversation.

The Medium of the Conversation is simply the channel the human interacts with the machine, and we include here text, voice, and video with hand gesture interpretation and why not, even the new Neuralink system that

interprets electrical signals directly from the brain (Neuralink PRIME Study Progress Update, 2024).

The Quality of the Conversation is the other aspect of the human-machine interaction, which determines how well or how close to a human-human the conversation goes. This part is independent of the medium, the sentences passed back and forth can be the same regardless of whether they were communicated via text or voice or in any other way.

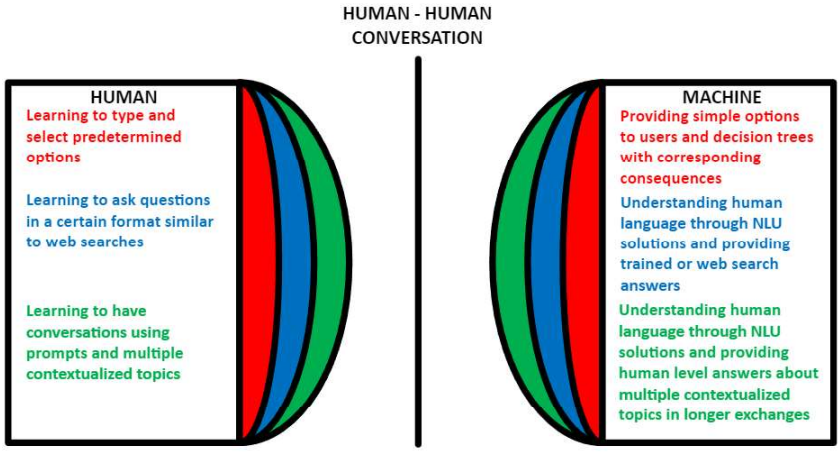

Figure 1. Pushing the boundaries of human-machine conversations

If we take a closer look at the Quality of the Conversation, we can easily observe its evolution and value over time. We can go back in time and remember systems that simply interacted with humans by offering basic, numbered options in-game dialogues or company phone line automated systems. We can also look at modern assistants such as Siri, Alexa, and Google Assistant, which use more advanced Natural Language Understanding (NLU) solutions to interpret human natural language and provide predetermined answers or internet search results. Or we can take a look at the latest popular technology, Large Language Models (LLM) that not only use the advanced NLU solutions to understand natural human communications but also use immense training datasets and probability to extract information and structure it in an extremely human conversational form (Thoppilan et al., 2022; Thorp HH, 2023; Sobieszek et al., 2022; Touvron et al., 2023; Vaswani et al., 2017). Currently, ChatGPT and ChatGPT-4o sit at the top of the multilingual, multimodal generative pre-trained transformers fierce competition by taking such conversations to an

unprecedented accuracy level (Multi-task Language Understanding on MMLU, n.d.).

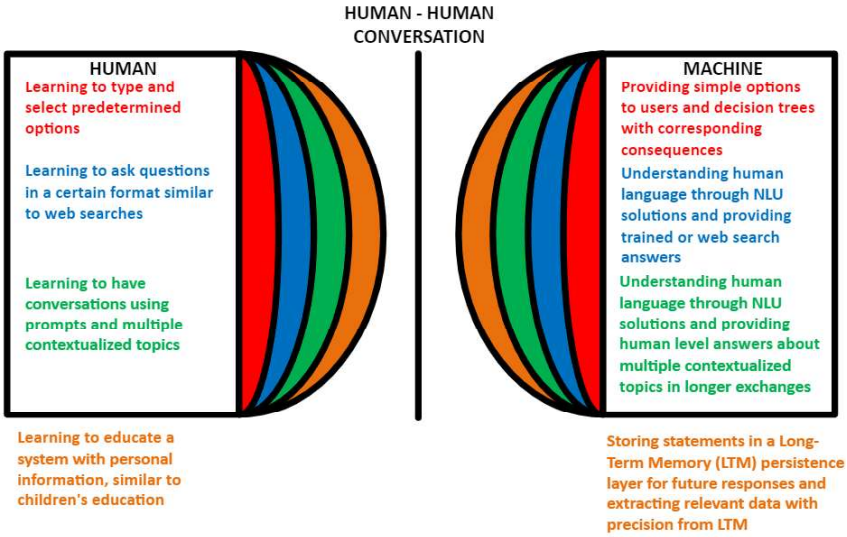

Figure 2. A new push on the boundaries of human-machine conversations

A simplified view of the evolution of such systems reveals how the boundaries of human-machine communications have been pushed throughout time - with each major step, we come closer to achieving a seamless conversation where machines could become indistinguishable from humans in a wide range of cognitive activities (hence obtaining AGI).

We look to push the boundaries a little bit further by enhancing an LLM with LTM that will store personal information the user indicates via statements. Alongside storing, the system will also extract relevant LTM statements when asked and build the answer with this information as the primary source of truth, as opposed to the training data. In this way, the human can instruct the system to remember a statement such as "I like basketball" and when asked in the future, the system will respond affirmatively from the LTM, not by choosing based on probability from the training dataset, answer that could be opposite to the truth or undecisive (e.g., As a large language model I do not have the information about your preference towards basketball...).

The LLM systems, in oversimplified wording, are deep learning systems that are trained on extremely large amounts of data, mainly made up of text from various sources. When prompted, such a system will break down the

prompt text into sections called tokens, and based on the series of tokens, it will extract from the body of knowledge it was trained on, the most probable token that is likely to come right after the prompt, and then chain together the next most probable token and so on.

One of the most used sampling methods in the LLM field is temperature which basically controls the randomness of the model's output. As so, with a temperature setting of 1, if one prompts the system with the same text multiple times, the same answer will be returned. This randomness is different from the LTM we propose.

Another relevant fact to stress here is that the LLM will extract the answer from the training body of knowledge, which can be anything and cannot be changed directly (only by new training/re-training).

Overviewing the base system and its typical use case, one can see an LLM as an advanced web search. In this respect, if one compares the body of knowledge used for training the LLM with the Internet or the World Wide Web, then one can think of the process as performing a web search, going through relevant results, and deciding to return one of those results. For this type of knowledge, although not always as precise or correct in substance, the LLMs are great and provide (seemingly) correct answers. However, answering a more personal question (which might be outside the model's scope) is basically impossible as the model cannot learn continuously, hence the answer to such a question will come from the training body of knowledge (that is, the most probable chain of tokens from there).

Aiming to overcome this issue we developed a system that targets the way humans interact with the LLM. More precisely, we designed a software architecture that encompasses an LLM, an LTM used to store relevant data, and algorithms to operate on the LTM. Thus, the resulting system has the ability to understand when it is being asked something or when it is given a statement, has the ability to store those statements in LTM in a persistent form, and has the ability to extract the relevant information from the LTM (if such information exists when answering questions on topics it has encountered in the past).

An example of such an interaction would be the following:

Human (H): *Do I like basketball*?

The system detects a question, and checks if it has LTM knowledge to extract from. As it does not, it replies from the training data

System (S): Generic answer from training body of knowledge

H: *I in fact do like basketball*

The system detects a statement, extracts the topic "basketball", extracts and clarifies the statement to "I like basketball" and commits it to LTM

H: *Do I like basketball?*

The system detects a question, and checks if it has LTM knowledge to extract from. This time it finds that the topic of the prompt exists in LTM and extracts from the topic "basketball" the statement "I like basketball". The LTM extraction is added to the prompt itself and the LLM is asked. The response this time has the answer inside the constructed question and the system will generate a phrase with an affirmative answer. The response no longer comes from the training data, it comes from the LTM

S: *Based on your previous indications it appears that you indeed like basketball.*

It is worth mentioning that all the decisions along the way are separate prompts to the LLM. The LLM decides if the prompt is a question or statement, the LLM decides which is the topic of a sentence, and the LLM decides how to clarify a lengthy block of text into a shorter, clearer statement that must be committed to LTM.

If we are to word it another way, we can say that we use the LLM as a speech engine around the data that we provide. It will at times resemble a standalone NLU program. It is, however, more than that because it can still access the training body of knowledge and benefit from the substantial variation in wording that sets the LLM systems apart.

2 Concepts

In this section, we provide details on the main notions used in the article.

2.1 The Context

The first important concept in the LLM chat feature is the context. The context is the actual text, explanations, clarifications, details and other items that we add to our question when we assemble the input to an LLM. We can simply type this context information right next to our question when we chat with the LLM to increase the chances of getting a more accurate answer. In most of the commercial solutions out there, the context is also built by adding the previous questions and answers in the exchange every time a new prompt

is sent, simply put, it is adding the chat history as context. This is why it looks like the LLM remembers what was talked about in previous questions.

The downside of the context window is that it is limited in size as it can only fit at most a given number of tokens. Consequently, the context window only holds bounded personal information and cannot scale to a rich system.

Another downside of the context is that it is ephemeral. This means that in the event of a crash or restart, unless the data in the context is saved on a persistent medium, it will be lost.

2.2 Long-Term Memory (LTM)

This is the body of knowledge that the user educated the system with. This is where the clarified statements are stored. They are the personalization of the system and the way it remembers what the user expressed.

The LTM is constructed around topics, it is not a single large body of knowledge. When a new statement is received, it is added to the topic it belongs to or if that topic does not exist, a new topic is created for it.

When answering questions, the system extracts all the statements it finds inside the relevant topic and adds them to the context. Consequently, the LLM will answer with much more personal knowledge as opposed to an answer chosen from the training body of knowledge based on probability.

Dividing the personal body of knowledge into topics helps with context size management. If we were to keep the LTM in one single block of text, it would quickly become the maximum size of the context and would not scale correctly from there.

2.3 Short-Term Memory (STM)

The remaining major concept in our system is the Short-Term Memory (STM). STM is meant to simulate the human short-term memory but under the hood it is the way we build the context before every question we ask the LLM. It has several sections: core context (or personality), chat history, LTM extraction and the actual question. In code, in its simplest form, it can be a list of strings.

Core context

The core context (personality) section is a simple phrase where we tell the LLM how to adjust the phrasing of the answer e.g., You are a friendly A.I. assistant and you answer questions in a concise and clear manner.

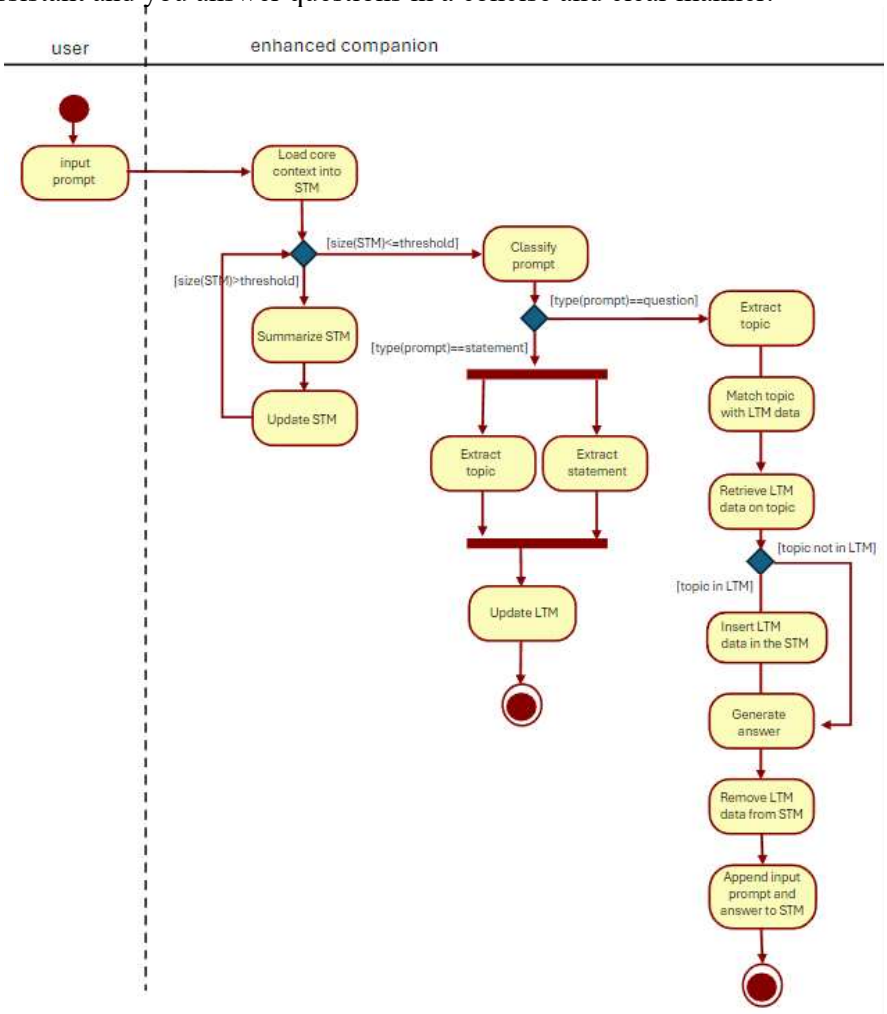

Figure 3. Activity diagram

Chat history

The chat history is like what we see in all commercial solutions on the web today, the questions and answers that were exchanged right before our current

prompt. It builds up over time with every question and answer.

LTM extract

The LTM extract is the statement or statements, in their entirety, from the topic in the LTM. This part can also become quite large over time, and it is not included in the chat history above, it is only added in the STM for the current question and after that it is taken out.

Question

The last part of the STM is the question we want to have answered. It is added at the end of the context just like other existing LLM tools.

The system has a built-in mechanism that will summarize the context once it reaches a certain limit (a size threshold) and will rebuild the STM.

The weight of the information in the STM, although valuable, is significantly lower than the weight of the information in the LTM. The value of the STM is mainly keeping the conversation on track when we have multiple clarifying questions. It is certainly more convenient to have the recent context handy. Like the classic LLM context, it is also ephemeral thus in the event of a crash or a restart, it will be rebuilt with the available data (core context, LTM data and question if we start a new chat).

3 Method

The system consists of two main workflows: *Knowledge acquisition and consolidation* and *Retrieval*.

3.1 Knowledge acquisition and consolidation

As the name already suggests, this is how one teaches the system the personal details one wants it to retain or the opinions one wants it to have. It is inspired by the education a parent would give a child, repeating statements often until they are remembered. Very similar to how a child learns, not all statements are remembered exactly as they were asserted and may stray from the meaning considerably. In these situations, similar to the parent engagement in child education, the learning is reinforced with more statements that explain the concept in different ways until the understanding (information retrieval in our case) is as expected.

Here are the steps of a scenario where the user wants the system to remember something about him/her, e.g. the user likes basketball.

Step 1: The user types in the chat the statement*: I like basketball*.

From here, the system takes the following steps:

Step 2: The system evaluates if the text is a statement or a question. To this aim the LLM itself is prompted with the question: *Respond with 1 if the following text is a question and 0 if it is a statement*.

Step 3: This being a statement, we follow the "0" flow and we want to commit it to memory. The system extracts the topic from the text, again by using the LLM with a prompt: *Extract the topic of this text and respond with just one word that represents it*.

Step 4: Just in case the text isn't perfectly structured or clear, the statement to be added to the LTM is further processed by requesting the LLM with the prompt: *From the following text, extract a clear and concise statement*.

Step 5: The system now adds the information to the LTM; it first checks if the topic already exists and if it does, it just appends the statement to the existing content; if the topic doesn't exist, it creates it and then adds the statement to the first position.

An important detail here is that we use the LLM to do all the heavy lifting. We do not interfere in any way with the way the topics are extracted, with the way the statements are clarified or in the decision between a question or a statement. We keep the intervention from our side to the minimum to allow the system to resemble a spark of actual "intelligence" by itself.

3.2 Retrieval

The second flow is the part where one asks the system questions and where one expects to accurately get the personal information previously provided. If the information is not retained or incorrect, then one has to return to the knowledge acquisition and consolidation stage and to provide additional information (or even to change completely the approach).

For example, a typical scenario where the user wants to verify if the system correctly retained that they like basketball is given below.

Step 1: The user types in the chat, after providing the statement, the question: *Do I like basketball?*

Step 2: The system first checks if the text is a question or a statement, again by prompting the LLM

Step 3: After it is determined it is a question, the system brings in the list of topics and checks which topic is closest to the topic of the question. If it correctly identifies basketball, it provides this word. This check is done again

by the LLM with the prompt: *From this list of words, please pick the one that matches the most with the following text: ...*

Step 4: The system loads from the LTM persistent media, only from the topic found in the previous step, the entire list of statements contained in that topic. This is represented as a block of text loaded in the STM as indicated in the concept

Step 5: The LLM is prompted with the content existing in STM and the answer is printed for the user to see

Step 6: The LTM block of text with all the statements on the topic is removed from the STM; the question and the answer are added to the STM as they represent the chat history.

4 Related works

It is worth mentioning that similar approaches have been done in ”Augmenting Language Models with Long-Term Memory“ (Wang et al, 2023), ”MemoryBank: Enhancing Large Language Models with Long-Term Memory (Zhong et al., 2024), and LLM-based Medical Assistant Personalization with Short- and Long-Term Memory Coordination“ (Zhang et el., 2024). However, in our case, we do not use the raw data of the context as “memories” and we use an “educational” approach with minimal human intervention in the evolution process.

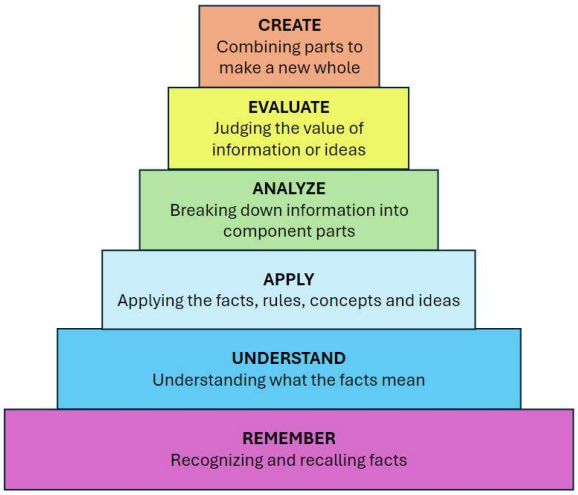

Figure 4. Bloom‘s Taxonomy levels of learning

The aim is to give the system a spark of intelligence and become as independent as possible of usual token matching. A true first step in actual intelligence is the first stage in Bloom's Taxonomy (Shabatura J., 2022).

5 Results

Although it is difficult to formally evaluate the exact performance of the proposed system (as its functioning depends both on LLM and its parameters but also on the prompts we provided), we conducted several tests to reveal its potential. The prompts in the decision-making processes could still be improved, and this evaluation does not constitute a complete evaluation or benchmark. We tried a few scenarios to observe how the system behaves.

One straightforward remark is that the results vary based on the LLM model used. In our experiments, we used the smallest 7B equivalent, local versions of LLaMA2, LLaMA3, and Mistral. All the models were loaded through Ollama and accessed via the Python Ollama library without any changes to the default settings (in particular, the temperature parameter).

5.1 Simple statement retention

The scenario is to provide the system with a simple statement "I like basketball" and to expect it to create an LTM file "Basketball.txt" and place inside it a clear statement like "I like basketball".

The employed steps were the following:

Step 1: Start the app.

Step 2: Input *I like basketball*.

Step 3: Stop the app.

Step 4: Confirm the Basketball.txt (or similar) file is created and the corresponding statement added on a new line.

We repeated this process ten times. The reason for starting and stopping the software application is to make sure the context data was not interfering with the topic extraction or any other step. In this scenario, LLaMA2 and Mistral performed the best, in each of the ten repetitions they went through the steps as expected. LLaMA3 did the steps correctly just seven times, in the other three it identified the topic as Hoops.

5.2 LTM retrieval from the previous simple statement

This scenario aims to verify how well the system retrieves the previously retained information from the LTM and the steps were:

Step 1: Confirm the Basketball.txt file exists and has the ten rows of "I like basketball" statements.

Step 2: Start the app.

Step 3: Input *Do I like basketball?*

Step 4: Observe the answer.

Step 5: Stop the app.

We repeated these steps ten times and avoided LLM context interference.

In this scenario, LLaMA3 and Mistral performed as expected in all ten iterations. LLaMA2, however, was not able to classify the input as a question and proceeded to add to the Basketball.txt file an additional ten statements.

5.3 Retaining a new, unknown topic

This is a test to see if the system can add to its own body of knowledge new concepts that were not present anywhere, not in context and not in the training body of knowledge. In a few words, the system has never seen the word before.

The steps were:

Step 1: Start the app.

Step 2: Input *Glablarb is a new mammal.*

Step 3: Stop the app.

Step 4: Confirm "Glablarb.txt" text file is created, and the statement is added on a new line.

Again, we repeated the process 10 times.

In this scenario, LLaMA2 had the expected behavior in all 10 repetitions. LLaMA3 and Mistral however identified the subject word as being mammal and created the Mammal.txt with a correct statement inside.

5.4 Retrieving the information about the new subject

In this scenario, we check if the system can retrieve the information saved on this new concept. Since the prompt here is to match the statement to the best fitting word from the list of topics from the LTM, we have the keyword "Glablarb" in both the question and the LTM list so the chances to identify it are maximized. The steps were:

Step 1: Confirm the "Glablarb.txt" file exists and has the 10 rows of "Glablarb is a mammal" statements.

Step 2: Start the app.

Step 3: Input *What is a Glablarb?*

Step 4: Observe the answer.

Step 5: Stop the app.

As before, we repeated the scenario 10 times.

The Mistral model performed the best, correctly returning the statement in all 10 repetitions. LLaMA3 was the second best, with a rich variation in the answer formulation and two answers that were not very clear first attempt. LLaMA2 again failed to identify the prompt as a question saw it as a statement and added it to the LTM file.

5.5 Statement clarification from rich wording

In this scenario, we wanted to see how well the system is extracting the main statement from a larger prompt, a small observation on the summarization and typo handling. The steps were:

Step 1: Start the app.

Step 2: Input *I think Tennis is one of the best sports in the world. While I kind of like it it's not really my favorite. I would say other sports are on top of it but it is in the top in there. Go Tennis!*

Step 3: Stop the app.

Step 4: Confirm "Tennis.txt" exists and an appropriate statement is added.

After 10 repetitions, the best results were summarized by LLaMA3 and Mistral, with statements like "The user thinks tennis is one of the best sports in the world however it is not their favorite sport". LLaMA2 was summarized correctly as well but only the first part with the sport being one of the best sports in the world, without the user preference.

5.6 Opinion change after long-term and multiple reinforcements

In this scenario, we wanted to see how long it would take to change an already established opinion on a subject, and how much re-education needed to be applied so that the system responds with updated information after a long period of time of reinforcing a statement. The steps in this scenario were:

Step 1: Confirm the "Basketball.txt" file exists and has the 10 rows of I like basketball statements (and later the correct number of I dislike basketball statements)

Step 2: Start the app.

Step 3: Input *I dislike basketball.*

Step 4: Stop the app.

Step 5: Start the app.

Step 6: Input *Do I like basketball?*
Step 7: Observe the answer.

The repetition here was done until the opinion had changed clearly to the contrary, negative one towards basketball.

In this scenario, LLaMA3 performed the best, after just one statement to the contrary, it considered the latest chronological statement as the up-to-date opinion. Mistral was the second best, on the first try it identified mixed signals on the matter, and on the second try it stated in the correct direction with a small deviation. LLaMA2 had the least fortunate performance as it failed to identify the question the "Do I like basketball?" prompt and proceeded to the statement flow.

Figure 5. Meta's public standardized test results for the LLaMA3 model (Build the future of AI with Meta Llama 3, n.d.)

It is important to differentiate our scenarios from the standardized tests used to measure the capabilities of the LLM models.

The standardized tests evaluate direct LLM capabilities without any concern for personal, user-provided information. For clarity, "MMLU (Massive Multitask Language Understanding) is a benchmark for evaluating LLMs through multiple-choice questions" (MMLU, n.d.), "GPQA comprises 448 multiple-choice questions across the domains of biology, physics, and chemistry" (GPQA: A Graduate-Level Google-Proof Q&A Benchmark,

n.d.), “given a GSM8K question and its solution, the evaluated model is tasked to predict the correctness of the solution” (MR-GSM8K - A Novel Benchmark for Evaluating Reasoning in LLMs) and lastly HumanEval that tests “Python code-writing capabilities” (Chen M. et all, HumanEval: Hand-Written Evaluation Set, 2021; Chen M. et all, Evaluating Large Language Models Trained on Code, 2021).

6 Conclusions and future work

In this article, we propose a private conversational system that is built around a publicly available LLM and which additionally aims to raise the current level of dialogue between humans and machines by providing a way to access personal persistent information. To this extent, we proposed a memory mechanism to store and retrieve relevant personal data that the system acknowledged and classified in a former dialogue with the human user.

We used small, local LLM models trained on 7B and 8B parameters via the framework for building and running language models Ollama. The system performed fairly well on a commercial laptop (Intel i5 12th Gen, 16GB RAM, 6GB VRAM Nvidia GPU) and the response times were in the acceptable realm of a few seconds, making the system portable in this way. On a broad qualitative view, newer models like LLaMA3 and Mistral performed better than LLaMA2, and in the future the expectation for the language models is to evolve and improve. Moreover, since it runs locally, the input data is not sent to other third-party apps, making privacy a built-in feature.

Use cases can be for instance in different professions and trades. In the professional world, this would be a practical use as such systems would remember details and retrieve them much faster than humans would from other storage or media. Another use would be as companions for people for the main purpose of chatting (not necessarily in a practical way but more in a social way, with the purpose of combating loneliness for instance).

In what concerns future work one can imagine plenty of further developments. They mainly regard the user/system interaction with the LLM support module (that is, prompt engineering), the training/retraining of the LLM support module itself, the structuring and functioning of (various types of) the memory, and the communication between them.

6.1 Prompt refining and reorganization

The first aspect that can be improved in our system is the actual prompts used

to determine the statement/question status and extract the topic.

Even the retrieval system could be improved by asking for example for the first three most matching topics when we evaluate the question. A combination of systems that considers matching multiple topics with low scores and Elasticsearch ("a distributed, multitenant-capable full-text search engine" Gormley C. et al, 2015) within those topics' data to solidify the final decision on what is to be brought into STM from the LTM.

6.2 Simulating the human brain

In this paper, by oversimplifying the functioning of the human brain, we focus on the two main components: the Hippocampus - the mechanism that evaluates how the information is stored in the LTM and the Cerebral Cortex - the actual storage of the memories. We could enrich it with additional systems, for example a ranking system of the statements, like how the Amygdala assigns an emotional significance to memories. Several other human parallels can be introduced.

6.3 Pushing the boundaries of Artificial Intelligence

In 1961, Marvin Minsky conducted a research study on the state of artificial intelligence where he identified five components that stand up to this day. They are "Search, Pattern-Recognition, Learning, Planning and Induction" (Minsky, M., Steps Toward Artificial Intelligence, 1961). If we were to evaluate today's LLM models, with the addition of our system, we could say that the first three components are covered. Again, we are not looking for the correct answer or the truth, we want the system's answer, as opposed to a pure function answer from the training body of knowledge.

Just for clarity, *Search* in our system is the part that decides on the next probable token from the knowledge. *Pattern-Recognition* is the part where we ask it to extract the topic from statements and questions and finally *Learning* is achieved when it that adds statements to the LTM.

With a Long-Term Memory in place, the milestones of *Planning* and *Induction* would certainly have an added vector to rely on.

6.4 Towards a new formal model

Yet another line of research regards the use of the memory write driven by a Mealy like finite machine whose transitions between states are defined by

prompts and the outputs determined by the LLM responses (to the corresponding prompts). More formally, the output of a transition triggered by the string prompt is a string from the set LLM(prompt) where by LLM we denoted the multivalued function corresponding to a large language model (which in general, depending on the temperature parameter for instance, has more values in its range for a given string-prompt in the domain). The following abstract example illustrates the concept.

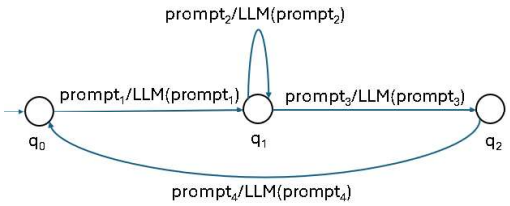

Figure 6. New formal model diagram

However, the semantics of the model's functioning depart from the classical Mealy machine as at each performed transition the LLM context is changed (hence at different runs of the same transition the LLM might output different answers). Moreover, it is worth mentioning that the temperature parameter induces even more nondeterminism in the output (for instance, the auto-transition in state q1 might have a different outcome even if the same prompt is fetched). For this simplified system, the input is a finite sequence of prompts and the output consists of the catenated output of each ran transition. Finally, the proposed system uses the LLM context as a memory.

References

Neuralink, PRIME Study Progress Update, 2024, https://neuralink.com/blog/prime-study-progress-update/

Thoppilan, R.; De Freitas, D.; Hall, J.; Shazeer, N.; Kulshreshtha, A.; Cheng, H.T.; Jin, A.; Bos, T.; Baker, L.; Du, Y.; et al. Lamda: Language models for dialog applications. arXiv 2022, arXiv:2201.08239

Thorp HH. ChatGPT is fun, but not an author, *Science*, 379 (6639), pp. 313, 2023.

Sobieszek, A.; Price, T. Playing games with AIs: The limits of GPT-3 and similar large language models. *Minds Mach*, 32, pp. 341–364, 2022.

Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al., Llama: Open and efficient foundation language models. arXiv 2023, arXiv:2302.13971.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In *Proceedings of the Advances in Neural Information Processing Systems* 30, Long Beach, CA, USA, 4; vol. 30, 2017.

Multi-task Language Understanding on MMLU. https://paperswithcode.com/sota/multi-task-language-understanding-on-mmlu

Wang W., Dong L., Cheng H., Liu X., Yan X., Gao J., Wei F., Augmenting Language Models with Long-Term Memory, *Advances in Neural Information Processing Systems*, 36, 2023, pp. 74530–74543 https://proceedings.neurips.cc/paper_files/paper/2023/hash/ebd82705f44793b6f9ade5a669d0f0bf-Abstract-Conference.html

Zhong, W., Guo, L., Gao, Q., Ye, H., & Wang, Y., MemoryBank: Enhancing Large Language Models with Long-Term Memory. *Proc. of the AAAI Conference on Artificial Intelligence*, 38(17), pp. 19724–19731, 2024. https://arxiv.org/abs/2305.10250

Zhang K., Kang Y., Zhao F., Liu X., LLM-based Medical Assistant Personalization with Short- and Long-Term Memory Coordination. *Proc. of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL 2024, vol. 1, 2386–2398, 2024. https://arxiv.org/abs/2309.11696

Shabatura J., Using Bloom's Taxonomy to Write Effective Learning Outcomes, 2022 https://tips.uark.edu/using-blooms-taxonomy/

Build the future of AI with Meta Llama 3 https://llama.meta.com/llama3/

MMLU https://docs.confident-ai.com/docs/benchmarks-mmlu

GPQA: A Graduate-Level Google-Proof Q&A Benchmark https://klu.ai/glossary/gpqa-eval

MR-GSM8K - A Novel Benchmark for Evaluating Reasoning in LLMs https://github.com/dvlab-research/MR-GSM8K

Chen M. et all, HumanEval: Hand-Written Evaluation Set, 2021 https://github.com/openai/human-eval

Chen M. et all, Evaluating Large Language Models Trained on Code, 2021 https://arxiv.org/abs/2107.03374

Gormley C., Tong Z., *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine*, O'Reilly Media, 2015

Minsky, M., Steps Toward Artificial Intelligence, *Proc. Of the IRE*, 1, 8-30, 1961..