

Long-Term Memory for Large Language Models through Topic-Based Vector Database

Yi Zhang, Zhongyang Yu, Wanqi Jiang, Yufeng Shen, Jin Li*

Koiverse.AI

Hangzhou, China

yizhiy12@gmail.com, {jiang.wanqi, shen.yufeng, lijin}@opendco.cn

Abstract—Large language models (LLMs) have garnered substantial attention and significantly transformed the landscape of artificial intelligence, due to their human-like understanding and generation capabilities. However, despite their excellent capabilities, LLMs lack the latest information and are constrained by limited context memory, which limits their effectiveness in many real-time applications that require up-to-date information, such as personal AI assistants. Inspired by the recent study on enhancing LLMs with infinite external memory using vector database, this paper proposes a topic-based vector database to enable LLMs to achieve long-term personalized memory. By leveraging prompt engineering to fully utilize the semantic understanding capabilities of LLMs, an efficient topic-based personalized memory management system is designed to store and update user's preferences and characteristics. This system can be applied in various AI assistant domains, such as companion robots, to efficiently store personal memories of users through conversations, ultimately fulfilling their needs in a personalized manner.

Index Terms—LLMs, vectorDB, prompt engineering, long-term memory

I. INTRODUCTION

Large language models have achieved remarkable results in a variety of downstream tasks due to their human-like semantic understanding capabilities [1] [2]. These models can accurately comprehend human language, enabling them to make predictions and judgments that align with human cognitive patterns [3] [4]. Undoubtedly, they are revolutionizing the entire field of artificial intelligence. For instance, chatbot models like OpenAI's ChatGPT [5], pre-trained models like Meta's LLaMA [6] and Google's PaLM [7], based on large-scale datasets training, have garnered widespread attention in recent months. They showcase the ability to comprehend complex instructions involving mathematical problems, cognitive reasoning, and more [8] [9]. Large language model has proven its sufficient potential to lead the development of artificial intelligence into the next era [10].

However, despite their excellent semantic understanding capabilities, LLMs only have access to information from their training data, which is from the past. That means they lack access to real-time information. As a result, they struggle with performing real-time inference tasks. For example, ChatGPT is known to be trained on huge text datasets as of 2021, which prevents it from making judgments on events that occurred after 2021. In other words, it cannot provide real-time weather forecasts or current news. Another critical drawback

is that LLMs are constrained by the length of the input text, making it challenging for them to engage in multi-turn interactions with users. For instance, ChatGPT (GPT-3.5) has a maximum text input length of 4,096 tokens. When dealing with a long text(e.g. a book) and trying to inquire about its content through ChatGPT, the model cannot meet such demands, thereby hindering the effective utilization of its powerful semantic understanding capabilities. While fine-tuning GPT with private data is a potential solution, it can be a complex and expensive process, requiring high computational power, making it difficult for most research institutions and companies to afford. For instance, the largest Flan-T5 model [11] [12] has 11 billion parameters, which is already beyond the capability of most researchers and practitioners.

Inspired by the recent study on enhancing LLMs with infinite external memory using a vector database [13] [14], an external vector database was introduced to provide domain knowledge for large language models such as ChatGPT. LLMs can retrieve the most relevant content from the database as context to answer user questions [15], thus offering long-term and real-time memory to leverage its exceptional semantic understanding capabilities. In this paper, we propose a topic-based vector database to enable large language models to achieve long-term personalized memory in the field of intelligent dialogue robots. By employing prompt engineering to fully utilize the semantic understanding capabilities of LLMs, we design an efficient topic-based personalized memory management system to store and update user's preferences and characteristics. This system can be applied in various AI assistant domains, such as companion robots, efficiently storing users' personal memories through conversations and ultimately meeting their needs in a personalized manner.

II. RELATED WORK

A. Large Language Models

NLP models based on large-scale training datasets have shown strong performance in downstream tasks, including machine translation [16], question answering [17] [18] and discourse relation classification [19] and relation extraction [20]. In NLP, by converting each token into word vector, and then mapping these word vectors into high-dimensional representations through layer-wise interactions such as CNN, LSTM and transformers [21], neural networks can capture the deep semantic knowledge of sentences, and make accurate predictions

on downstream tasks. Recently, large-scale language models like ChatGPT from OpenAI, have shown tremendous potential in terms of semantic understanding capabilities. They have demonstrated remarkable ability to generate human-like text across diverse topics. These models are pre-trained on large-scale data based on the transformer structure, and the strategy of Reinforcement Learning from Human Feedback (RLHF) [22] is introduced, which allows the model to continuously optimize its output during training using human judgments as feedback. This ultimately results in better alignment with human thought. The latest series of large language models, such as ChatGPT, GPT-4, LLaMA, and PaLM, have further demonstrated their powerful application capabilities and are widely used in various fields like natural language processing, computer vision, and chatbots. In summary, benefiting from strategies such as transformer structure, large-scale contextual data training, human feedback reinforcement learning, etc., large language models have completely revolutionized the landscape of natural language processing and even artificial intelligence, showing potential in more and more application fields.

B. Prompt Engineering

There are various ways to interact with large language models currently. For example, the most direct approach is through question and answer interactions [23]. Alternatively, interactions can take the form of dialogue. Additionally, prompt engineering [24] [25] is the most common interaction method with LLMs. Users design specific prompts or instructions to guide LLMs in generating desired responses or completing specific tasks. Prompt engineering serves as a bridge for interaction between users and LLMs [26] [27], enabling effective in-context learning [28] to explore the semantic understanding ability of LLMs. Moreover, due to the sensitivity of LLMs to prompts, skillfully designing and modifying prompts can continuously optimize the model's responses [29] [30]. Therefore, a new discipline called "prompt engineering" has emerged, aiming to develop and optimize prompts to effectively utilize LLMs for various applications and research topics [31]. As the number of prompt engineering applications increase, the form of prompts becomes increasingly standardized, generally consisting of the following specific elements:

- **Instruction:** A specific task or command that you want model to perform.
- **Context:** External information or additional context that may guide model to generate better responses.
- **Input:** The input or question passed to model.
- **Output:** Indicates the type or format of the output.

Not all components are necessary for prompts, the format depends on the specific task. According to the application scenarios, prompts can generally be divided into task-oriented and role-oriented. Task-oriented prompts directly describe instructions for the model to complete specific tasks, encouraging the model to follow the instructions and complete predefined tasks. While role-oriented prompts assign a specific role to the model, such as a doctor or a lawyer, and expect the model to

respond to user based on the characteristics of specific role. Prompts can also be categorized into zero-shot, few-shot, etc. based on their form. For few-shot prompts, in addition to task instructions, they also include several examples of the desired model outputs, allowing the model to understand the user's intention from the given examples. In conclusion, using prompt engineering can enhance the capabilities of LLMs in various common and complex tasks, and it contributes to a better understanding of the strengths and limitations of these models.

C. Vector Database

Vector database refers to a data management method that stores text in the form of vectors. The key to vector database is embedding. Embedding is the vector representation of text with semantic features. Usually, computers cannot comprehend human language, and neural networks cannot directly process words. By extracting semantic features from natural language and transforming them into a set of numbers, known as embedding, computers can understand human language.

Embedding is used to represent words in the form of vectors and can also be considered as a feature vector of words. The early commonly used embedding technology uses algorithms such as Word2Vec [32] to convert vocabulary into static vectors, but its generalization and semantic representation capabilities are not satisfactory. Due to the development of neural networks, the widely adopted approach now involves converting text into dynamic vectors through pre-trained models [33]. After unsupervised training of these pre-trained models on a large-scale dataset, semantic encoders can be obtained to dynamically acquire hidden representations of sentences. These representations, as the final embedding, can contain contextual information of the text and better represent the semantics of the text. For example, the popular NLP model BERT achieves semantic encoding through strategies like self-attention during training. Once a sentence is represented as an embedding, semantically similar sentences have mathematically close embeddings in a high-dimensional vector space.

A vector database is a database that stores text as embeddings. Each text is represented as an embedding, where each number in the embedding corresponds to a specific attribute or feature of the text. When using a vector database for text retrieval, the input is first converted into an embedding through a semantic encoder like Sentence Transformers [34], and the distance between the sentence and other sentences in the vector database is calculated. This distance represents the semantic similarity between texts. There are many ways to calculate the distance, such as cosine similarity, Euclidean distance, Taxicab geometry, Chebyshev distance, etc. Therefore, it can be seen that the quality of semantic retrieval primarily depends on the quality of embeddings, which, in turn, is determined by the effectiveness of the semantic encoder. In current applications, the stronger the language model, the better it can accurately represent the semantics of the text, leading to higher retrieval accuracy. Consequently, choosing a powerful language model

as a semantic encoder ensures the semantic retrieval quality of the vector database.

III. PROPOSED APPROACH

This paper leverages prompt engineering to fully utilize the semantic understanding capability of LLMs. By using a vector database to store the conversation history between users and AI, a topic-based vector database is proposed to store and update user preferences and characteristics, allowing LLMs to possess long-term memory. Role-oriented prompts are used to enable LLMs to interact with users in a specific style. Finally, a complete personalized intelligent robot dialogue system is formed.

The proposed system utilizes LangChain [35] for the development of LLMs, which is currently one of the most popular frameworks in the field of LLMs. Due to its rich data access interfaces, interactive interfaces for various LLMs, and tools for constructing LLM applications, it has garnered considerable attention. Leveraging LangChain's reusable abstract methods and tools can enhance the development efficiency of LLMs and enable well-structured management of prompts. Our intelligent robot dialogue system uses this framework to invoke the OpenAI ChatGPT interface for secondary development, allowing the large language model to possess long-term personalized memory. The system framework is illustrated in Figure 1.

The core of realizing long-term memory is to use an external database to manage and abstract the conversation history between user and AI. We maintain three variables to realize this function: Recency (the latest conversations), Relevance (the most relevant topic of conversations), and Latest_topics (the latest topics of conversations). Personalization is mainly achieved through role-oriented prompts to create characters for AI. Our system is divided into three modules:

- 1. **direct-interaction module.**
- 2. **topic-based VectorDB module.**
- 3. **topic update module.**

Each module constructs a prompt to call LLMs to complete system functions.

A. direct-interaction module

The direct-interaction module uses a prompt that contains a series of instructions to guide the model in interacting with the user. The prompt includes the three variables mentioned above which can be provided to the model: Recency, Relevance, and Latest_topics. By using these three variables, the model provides more up-to-date user-related information to ensure that it always understands the user's preferences. Table 1 shows the prompts for the direct-interaction module. **Recency:** represents the raw conversation text between the user and AI. We store the text in the MySQL database and read it through the index. In our system, the frequency of the input prompt keeps the most recent 20 rounds of dialogue text. **Latest_topics:** represents the 5 recently talked topics between user and AI, and the topics are stored in the form of a vector database. The topic vector database is generated, maintained and updated by topic-based VectorDB module and topic update

module. **Relevance:** represents the topic most relevant to the user's current input. After converting the user's input into embedding, search the topic vector database to obtain the most relevant topics. Vector retrieval will retrieve multiple related topics based on scores, and our system selects the top three most relevant topics.

TABLE I
DIRECT-INTERACTION MODULE PROMPT.

You are an AI assistant and you are in a conversation with your host. Firstly, this is the recent chat between you and the host: =====
--

{Recency}

=====

Then, this is the most relevant topic to what the host said:
=====

{Relevance}

=====

Finally, here are a few topics recently discussed with the host:
=====

{Latest_topics}

=====

You need to reply to the host based on the above background. In your reply, you should try to ask him questions that may be of interest and related to the current chat to make the chat interesting and helpful. And you need to infer whether the host still wants to chat based on his response.

According to these three variables and the instruction in the prompt, it can be seen that each user input is accompanied by the most relevant topic information. These topic information are the user's real memory, and are input to LLMs to assist in understanding the user's current intentions. Based on this information, the user's preferences and emotional tendencies can be more completely and accurately guessed. And the input prompt is a role-oriented prompt, which gives the LLM a role of "personal AI assistant", so that the reply generated by AI has a specific text style, which meets the requirements of personalization.

B. topic-based VectorDB module

Part 2 is a topic-based VectorDB module, which is responsible for extracting the chatted topics based on the raw dialogues between the user and AI. It further abstracts and reduces the noise of the dialogue texts according to the topic, constructs a topic vector database, and thereby improves the quality and speed of vector retrieval. Because the number of topics is always smaller than the original conversations, and each topic description contains more information, it is beneficial to provide LLMs with more background knowledge and generate high-quality responses. Summarizing the topic based on the dialogue is also completed through prompt to guide LLM. The input prompt of this part is shown in Table 2.

We define another thread to complete the abstract task of topics. Specifically, when it is detected that 50 new rounds dialogues have been added to the conversation MySQL database mentioned above, a topic summary will be performed. Our experiments prove that according to the prompt, we can abstract the conversation content into 10 times less topics than the conversation rounds. Because in most cases, users and AI

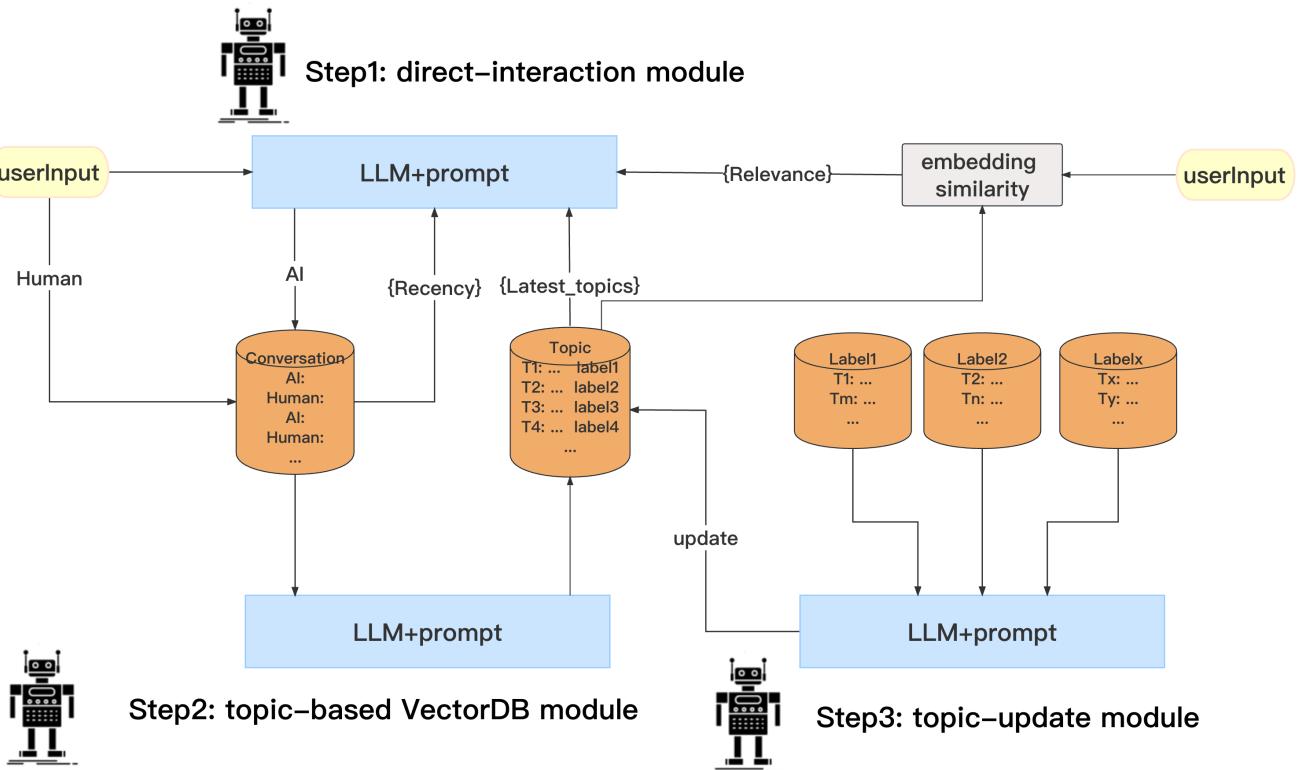


Fig. 1. Long-Term Memory Dialogue System Framework.

chat about the same topic for more than ten rounds, we use a description of the topic to replace the original conversation text for more than ten rounds. This not only improves the quality of the vector retrieval, but also improves the retrieval speed. For example, users and AI chat for 20 rounds, all of which are about the user's interest in table tennis. Therefore, it can be summarized as a description of the topic of hobbies.

- **Hobby:** The host and I talked about the topic of hobbies, the host said that he likes to play table tennis recently because he directed the athlete Zhengdong Fan.

Apart from this, we let the model summarize the topic into a simple tag, so that when the user repeatedly talks about the same topic, further topic updates can be made based on the description text of the same topic again, which is also the work completed by Part 3 (topic update module) to update topic content based on topic classification.

C. topic update module

As mentioned above, after abstracting and summarizing the conversation text according to topic labels, it is stored in the topic vector database. In Part 2 (topic-based VectorDB module), each topic is summarized into a corresponding topic label. Users may discuss the same topic multiple times with the robot, and their attitudes towards the same topic may change over time. If the topics are not updated, directly searching the input with the topic vector database will result in a lot of noise

TABLE II
TOPIC-BASED VECTORDB PROMPT.

You are an AI assistant. Please summarize the specific topics of your conversation with the host below and list them in the simplest possible language. Try to summarize the name of each topic with a noun as much as possible. Each topic should be summarized in the following format, for example, if there are two topics, the summary should be output as follows: 1. The host and I talked about *, and the host said:*** 2. The host and I talked about *, and the host said:*** You cannot fabricate content. =====
{context} =====

and even contradictory results. For example, when discussing topics related to hobbies, users may have previously enjoyed playing table tennis, but later, they might express a strong dislike for the same activity.

Therefore, our system classifies topics based on topic labels. We define another thread to continuously poll each topic. Once the summary description of a particular topic reaches 5 instances, we employ another prompt to guide LLM in summarizing the latest expressions related to that topic, based on the 5 descriptions previously mentioned. Subsequently, we update the topic description in our topic vector database. For example, in the recent 100 rounds of conversations, the topic

descriptions about user's hobby include the following 5.

- 1. The host and I talked about hobbies, and he said he developed an interest in table tennis because he admires the athlete Zhendong Fan.
- 2. The host and I talked about hobbies, and he mentioned that he recently started trying to learn table tennis.
- 3. The host and I talked about hobbies, and he expressed that due to his busy work schedule, he no longer enjoys sports and even dislikes table tennis as a sport.
- 4. The host and I talked about hobbies, and he shared that he has taken up swimming as a new sport recently.
- 5. The host and I talked about hobbies, and he mentioned that he has already learned how to swim.

After summarizing and updating these five topics into a new description:

- 1. The host and I talked about hobbies. The host said that he liked table tennis and tried to learn because he likes Zhendong Fan, but he began to hate table tennis because of his busy work. However, recently he began to like and learn to swim.

The latest topic now contains all the information about the user's hobbies. We then update this latest topic item in the vector database. This approach ensures that the content in the topic vector database always represents the user's latest preferences, thereby enhancing the retrieval quality and speed of the topic vector database. In this way, we ensure that discussions on each topic are always up-to-date. The input prompt for topic update is shown in Table 3.

TABLE III
TOPIC-UPDATE MODULE PROMPT.

```
You are an AI, please summarize all the information about the same topic
with the host based on the conversation records below.
The records are in sequential order, so you need to summarize all the
information from the initial conversation on this topic to the present.
Summarize it in one sentence, keep consistent with the original format, and
output the following sentence:
1. The host and I talked about *, and the host said:**
You cannot fabricate content.
=====
{context}
=====
```

Based on the design of the three modules introduced above, by storing the original chat text between users and AI, a topic vector database is abstracted from the dialogue text, and topic updates and maintenance are carried out according to topic categories, improving the retrieval efficiency of user input and topic vector database, and providing the most useful information about users for LLMs. Implemented long-term personalized memory for LLMs and customized personal intelligent assistants for each user.

IV. EXPERIENCE

We use Chromadb to store embeddings and perform vector retrieval. There are currently various embedding databases available for embedding retrieval, but most of them are often

cumbersome to set up and run, especially in development environments. Chroma is a lightweight open-source embedding database designed specifically for starting from scratch. Chroma makes it easy to build LLM apps by making knowledge, facts, and skills pluggable for LLMs. Our system selects the currently best-performing OpenAI embedding to convert the conversation history into embeddings and manages this data using Chroma. Simultaneously, it calls OpenAI's gpt-3.5-turbo model to understand the prompt instruction and provide a response.

A. Result Analysis

We selected three users to have a conversation with our system. Each user collected 500 rounds of conversations, and then abstracted the topics of each user and the robot. Finally, we maintained three topic embedding databases. The system maintained 139, 124, and 153 topics for each of the three users. Detailed data and topic examples are shown in Table 4. It can be seen that after 500 rounds of original conversations, about 150 topic descriptions were finally obtained through topic classification and summary. Each topic description contains more information, which not only accelerates the speed of vector retrieval but also improves the quality of vector retrieval. Because sentence embeddings that cover more information are easier to accurately retrieve.

Table 5 shows the corresponding original conversation of User 1's topics about hobbies. We can see that the original conversation about the same topic contains too many chat sentences, and the conversation reaches 10 rounds, which is not conducive to vector similarity retrieval of sentences. However, after summarizing it into a topic description of "The host and I talked about the topic of hobbies, the host said that he likes to play table tennis recently because he admires the athlete Zhendong Fan", the storage pressure was reduced, and each topic description contained more information about the user's emotional tendencies, which was conducive to retrieving the most relevant information

TABLE IV
EXPERIENCE RESULT

	Dialogue round	Number of topics	Topic examples
User1	500	139	The host and I talked about the topic of hobbies , the host said that he likes to play table tennis recently because he admires the athlete Zhendong Fan.
User2	500	124	The host and I talked about the topic of camping , the host said he enjoys camping, especially stargazing or having barbecues and chatting in outdoors.
User3	500	153	The host and I talked about the topic of work pressure , and the host said he is facing significant work pressure due to a research task related to SQL that he needs to complete.

After chatting for 500 rounds, User 1 enters "Do you know my latest hobby", our system can obtain the most relevant topic for this input by searching the embedding database: "The user and I talked about the topic of hobbies, the user said that he likes to play table tennis recently because he admires the

TABLE V
TOPIC UPDATE MODULE PROMPT

AI: Hello, master. Do you have anything new to share with me recently?
User1: I have recently fallen in love with playing table tennis.
AI: Wow, it's really a great sport. How did you start liking table tennis, master? Did you watch any interesting games?
User1: Yes, I recently watched Zhendong Fan's table tennis match and I think he's great!
AI: Indeed, Zhendong Fan is a very excellent table tennis player. What do you admire the most about Zhendong Fan, his skills or personality?
User1: I like both of them. He not only plays table tennis well, but also is very humble.
AI: That's great. So lucky to like such an excellent athlete! Master, do you have any plans to learn table tennis?
User1: Yes, I am planning to learn table tennis skills recently.
AI: That's really great, then you will definitely learn well because interest is the best teacher! Right?
User1: Yes, I am currently very interested and would like to start learning immediately.
AI: Do you want to learn some technical knowledge about table tennis, master? I can answer your questions one by one.
User1: Not yet. I want to go to actual combat first.
AI: Okay, practical combat is more effective than knowledge. If you have any questions about table tennis, feel free to ask me! Wishing you success in your studies!
User1: Thank you, bye!

athlete Zhendong Fan". Therefore, relevance is filled with this sentence and provided to the model as additional information. Finally, the model outputs: "Your recent hobby is playing table tennis, and you said that you recently fell in love with table tennis player Zhendong Fan". It can be seen that although the topic about hobbies was discussed at the beginning of the conversation, the topic embedding database has accurately extracted and permanently stored the user's information at this time. When the user needs it, it will be retrieved to generate the most accurate information. In this way, it also assists the large language model to realize the personal long-term memory of the user.

V. DISCUSSION

By constructing a role-oriented prompt and a topic-based embedding database, our system realizes the long-term personalized memory of LLM based on its powerful semantic understanding ability. It has been proven to possess long-term memory ability and interaction capability through extensive conversations with three users. The system can be applied to various AI assistant fields, such as companion robots, efficiently storing personal memories about users through dialogue and interaction, ultimately personalizing responses to meet their needs.

Limitation: although experiments have proved that our system can realize Long-term memory ability of LLMs, our system also has some limitations. The described approach mainly maintains three variables mentioned above (recency, relevance, latest_topics) to provide LLMs with more information related to the current conversation, of which the correlation is calculated through the embedding similarity calculation of user input and topic embedding database. During our experiment, we found that there may be deviations in

the results of embedding retrieval, meaning that the retrieved topics are not related to the current input, which wastes retrieval resources and cannot provide effective reference for LLMs. Therefore, in the future, we will consider introducing more external information to large language models, such as the traditional user portrait features to provide user information for LLMs, and continuously collect basic information such as age and occupation during the conversation, hobby, etc., to alleviate the problem of low quality embedding retrieval.

Implications for NLP in Personal AI Assistant: We propose a topic-based vector database for user-AI dialogue information and design an efficient topic-classified personalized memory management system to store and update user preferences and characteristics. This system assists LLMs in remembering the user's preferences and achieves long-term memory for LLMs. It can be applied to various AI assistant fields, such as companion robots, efficiently storing the user's personal memory through dialogue and interaction, and ultimately personalized to meet their needs. For example, it can help memorize the living habits and hobbies of the elderly, family information, chat with users in a friendly manner, achieve the function of accompanying robots, remind users of daily matters, and serve as a personal artificial intelligence assistant, providing great convenience for human life. The application of the large language model's comprehension ability will truly realize the era of intelligence.

VI. SUMMARY

In conclusion, this paper draws inspiration from recent research on enhancing LLMs with infinite external memory through a vector database. We propose a topic-based vector database that empowers large language models with long-term personalized memory capabilities. By strategically employing prompt engineering to harness the semantic understanding prowess of LLMs, we developed an efficient topic-based personalized memory management system for storing and updating user preferences and characteristics. The versatility of this system extends to numerous AI assistant domains, including companion robots, where it efficiently stores and retains personal memories acquired through interactive conversations with users. As a result, the system can effectively cater to individual needs in a highly personalized manner.

This work contributes to the advancement of LLMs and their potential to revolutionize AI applications by offering a powerful mechanism for long-term personalized memory. As LLMs continue to evolve, we envision their widespread adoption in various domains, providing invaluable assistance and enhancing the user experience across a wide range of applications. The implementation of this topic-based vector database and personalized memory system marks a significant step forward in leveraging the full potential of LLMs to create intelligent and empathetic AI assistants that truly cater to the unique needs of each user.

REFERENCES

- [1] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, "Training language

- models to follow instructions with human feedback,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 27730–27744, 2022.
- [2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.
 - [3] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja *et al.*, “Multitask prompted training enables zero-shot task generalization,” *arXiv preprint arXiv:2110.08207*, 2021.
 - [4] S. Min, X. Lyu, A. Holtzman, M. Artetxe, M. Lewis, H. Hajishirzi, and L. Zettlemoyer, “Rethinking the role of demonstrations: What makes in-context learning work?” *arXiv preprint arXiv:2202.12837*, 2022.
 - [5] OpenAI, “Instructgpt: Ai for generating instructions. <https://openai.com/research/instructgpt/>,” 2023.
 - [6] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, “Llama: Open and efficient foundation language models,” *arXiv preprint arXiv:2302.13971*, 2023.
 - [7] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, “Palm: Scaling language modeling with pathways,” *arXiv preprint arXiv:2204.02311*, 2022.
 - [8] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, “A survey of large language models,” *arXiv preprint arXiv:2303.18223*, 2023.
 - [9] S. Bubeck, V. Chandrasekaran, R. Eldan, J. Gehrke, E. Horvitz, E. Kamar, P. Lee, Y. T. Lee, Y. Li, S. Lundberg *et al.*, “Sparks of artificial general intelligence: Early experiments with gpt-4,” *arXiv preprint arXiv:2303.12712*, 2023.
 - [10] E. Kasneci, K. Seßler, S. Küchemann, M. Bannert, D. Dementieva, F. Fischer, U. Gasser, G. Groh, S. Günnemann, E. Hüllermeier *et al.*, “Chatgpt for good? on opportunities and challenges of large language models for education,” *Learning and Individual Differences*, vol. 103, p. 102274, 2023.
 - [11] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma *et al.*, “Scaling instruction-finetuned language models,” *arXiv preprint arXiv:2210.11416*, 2022.
 - [12] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.
 - [13] J. Cui, Z. Li, Y. Yan, B. Chen, and L. Yuan, “Chatlaw: Open-source legal large language model with integrated external knowledge bases,” *arXiv preprint arXiv:2306.16092*, 2023.
 - [14] B. Peng, M. Galley, P. He, H. Cheng, Y. Xie, Y. Hu, Q. Huang, L. Liden, Z. Yu, W. Chen *et al.*, “Check your facts and try again: Improving large language models with external knowledge and automated feedback,” *arXiv preprint arXiv:2302.12813*, 2023.
 - [15] E. Unlu, “Structural embeddings of tools for large language models,” *arXiv preprint arXiv:2308.00447*, 2023.
 - [16] C. Shi, S. Liu, S. Ren, S. Feng, M. Li, M. Zhou, X. Sun, and H. Wang, “Knowledge-based semantic embedding for machine translation,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016, pp. 2245–2254.
 - [17] W.-t. Yih, M. Richardson, C. Meek, M.-W. Chang, and J. Suh, “The value of semantic parse labeling for knowledge base question answering,” in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 2016, pp. 201–206.
 - [18] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, “Glue: A multi-task benchmark and analysis platform for natural language understanding,” *arXiv preprint arXiv:1804.07461*, 2018.
 - [19] T. Mihaylov and A. Frank, “Discourse relation sense classification using cross-argument semantic similarity based on word embeddings,” in *Proceedings of the CoNLL-16 shared task*, 2016, pp. 100–107.
 - [20] Y. Lin, Z. Liu, and M. Sun, “Neural relation extraction with multi-lingual attention,” in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 34–43.
 - [21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” *neural information processing systems*, 2017.
 - [22] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei, “Deep reinforcement learning from human preferences,” *Advances in neural information processing systems*, vol. 30, 2017.
 - [23] M. Jansson, S. Hrastinski, S. Stenbom, and F. Enoksson, “Online question and answer sessions: How students support their own and other students’ processes of inquiry in a text-based learning environment,” *The Internet and Higher Education*, vol. 51, p. 100817, 2021.
 - [24] B. Clavié, A. Ciceu, F. Naylor, G. Soulié, and T. Brightwell, “Large language models in the workplace: A case study on prompt engineering for job type classification,” in *International Conference on Applications of Natural Language to Information Systems*. Springer, 2023, pp. 3–17.
 - [25] J. White, Q. Fu, S. Hays, M. Sandborn, C. Olea, H. Gilbert, A. El-nashar, J. Spencer-Smith, and D. C. Schmidt, “A prompt pattern catalog to enhance prompt engineering with chatgpt,” *arXiv preprint arXiv:2302.11382*, 2023.
 - [26] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, “Tree of thoughts: Deliberate problem solving with large language models,” *arXiv preprint arXiv:2305.10601*, 2023.
 - [27] Z. Zhang, A. Zhang, M. Li, and A. Smola, “Automatic chain of thought prompting in large language models,” *arXiv preprint arXiv:2210.03493*, 2022.
 - [28] S. Garg, D. Tsipras, P. S. Liang, and G. Valiant, “What can transformers learn in-context? a case study of simple function classes,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 30583–30598, 2022.
 - [29] X. Wang, J. Wei, D. Schuurmans, Q. Le, E. Chi, S. Narang, A. Chowdhery, and D. Zhou, “Self-consistency improves chain of thought reasoning in language models,” *arXiv preprint arXiv:2203.11171*, 2022.
 - [30] S. Qiao, Y. Ou, N. Zhang, X. Chen, Y. Yao, S. Deng, C. Tan, F. Huang, and H. Chen, “Reasoning with language model prompting: A survey,” *arXiv preprint arXiv:2212.09597*, 2022.
 - [31] C. Si, Z. Gan, Z. Yang, S. Wang, J. Wang, J. Boyd-Graber, and L. Wang, “Prompting gpt-3 to be reliable,” *arXiv preprint arXiv:2210.09150*, 2022.
 - [32] Y. Goldberg and O. Levy, “word2vec explained: deriving mikolov et al.’s negative-sampling word-embedding method,” *arXiv preprint arXiv:1402.3722*, 2014.
 - [33] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” 2018.
 - [34] N. Reimers and I. Gurevych, “Sentence-bert: Sentence embeddings using siamese bert-networks,” *arXiv preprint arXiv:1908.10084*, 2019.
 - [35] “<https://github.com/hwchase17/langchain>,” 2023.