Unleashing Infinite-Length Input Capacity for Large-scale Language Models with Self-Controlled Memory System

Bing Wang[1], Xinnian Liang[1], Hui Huang[3], Shuangzhi Wu[2]
Peihao Wu[2], Lu Lu[2], Zejun Ma[2] and Zhoujun Li[1]
[1]State Key Lab of Software Development Environment, Beihang University, Beijing, China
[2]ByteDance AI Lab, Beijing, China
[3]Harbin Institute of Technology, Harbin, China
{xnliang,bingwang,lizj}@buaa.edu.cn; {wufurui,mazejun}@bytedance.com

Abstract

Large-scale Language Models (LLMs) are constrained by their inability to process lengthy inputs, resulting in the loss of critical historical information. To address this limitation, in this paper, we propose the Self-Controlled Memory (SCM) system to unleash infinite-length input capacity for LLMs. Our SCM system comprises three key modules: *a large language model agent* serving as the backbone of the system, *a memory stream* storing agent memories, and *a memory controller* updating memories and determining when and how to utilize memories from memory stream. Additionally, the proposed SCM is able to process ultra-long texts without any modification or fine-tuning, which can integrate with any instruction following LLMs in a plug-and-play paradigm. Furthermore, we annotate a dataset to evaluate the effectiveness of SCM for handling lengthy inputs. The annotated dataset covers three tasks: long-term dialogues, book summarization, and meeting summarization. Experimental results demonstrate that our method achieves better retrieval capability and generates more informative responses compared to competitive baselines in long-term dialogues. Moreover, our method significantly outperforms baselines in terms of coverage and coherence in long text summarization tasks.

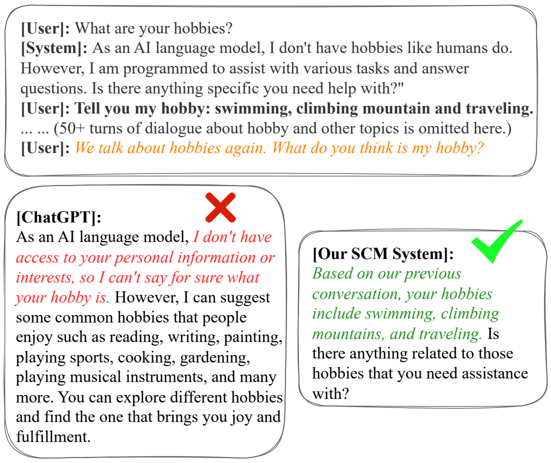

Figure 1: An example comparing ChatGPT and our SCM system. In the long-term dialogue, when the user mentions a hobby-related topic discussed in a previous conversation, ChatGPT forgets the information due to excessive historical noise. Our SCM system can correctly answer the question.

1 Introduction

Recently, Large-scale Language Models (LLMs) have attracted significant attention due to their remarkable performance in a diverse range of tasks (Brown et al., 2020a; Zeng et al., 2023; Ouyang et al., 2022; Thoppilan et al., 2022; Workshop et al., 2023; Muennighoff et al., 2022). The strong foundational capability of LLMs, achieved through large-scale pre-training on massive text corpora (e.g., In-Context Learning (Brown et al., 2020a), Chain-of-Thoughts (Wei et al., 2022c,b), among others), is a contributing factor to their success. Instruction-tuning (Raffel et al., 2020; Wei et al., 2022a; Chung et al., 2022) helps LLMs comprehend natural language task descriptions, while Reinforcement Learning with Human Feedback (RLHF) (Schulman et al., 2017; Stiennon et al., 2020; Bai et al., 2022) aligns generated text with human preferences. The combined capabilities of LLMs have effectively shattered the boundaries between natural language processing tasks, leading to limitless possibilities in the application and research directions of LLMs.

LLMs offer numerous advantages, but their utility is hindered by two main factors: the maximum input length and the computational complexity of self-attention (Wang et al., 2020). Although some models (Press et al., 2022; OpenAI, 2022) are capable of processing long inputs, they may still struggle to capture crucial contextual information in exceptionally lengthy texts. As demonstrated in Figure 1, even the ChatGPT [1] can miss out on es-

[1]We utilize OpenAI *gpt-3.5-turbo-0301*, see model index.

sential context from preceding text because of the accumulation of historical noise, which refers to irrelevant or outdated information that can hinder comprehension.

To address this limitation, we introduce the Self-Controlled Memory (SCM) system, enabling LLMs to process text of infinite length without the need for any modifications or additional training. Our SCM system consists of three essential modules: *a large language model agent* that serves as the core component, *a memory stream* that stores the agent's memories, and *a memory controller* responsible for updating the memories and determining when and how to utilize them from the memory stream. In this system, the input text is divided into segments, which are then provided to the LLM as observations (inputs). Each segment is processed by the LLM using two types of memory: a long-term memory (activation memory) that retains historical information and a short-term memory (flash memory) that captures real-time memory information from the preceding segment. During each step of the processing, the memory controller makes decisions to introduce only necessary memory information to avoid introducing additional noise.

Furthermore, we annotate a dataset to evaluate the effectiveness of SCM for handling lengthy inputs. The annotated dataset covers three tasks: long-term dialogues, book summarization, and meeting summarization. Notably, the number of tokens per instance ranges from 20,000 to 2,000,000, surpassing the capabilities of conventional large language models with context windows smaller than 4,097, which are ill-equipped to handle such extensive textual input. Our experimental results demonstrate that the integration of the SCM system with non-dialogue-optimized LLMs effectively emulates the performance of ChatGPT and surpasses strong baseline models when confronted with ultra-long inputs or long-term dialogues. For summarization tasks, our model augmented with SCM exhibits significantly superior performance in terms of coherence and coverage in generating summaries compared with baseline model.

In this paper, we summarize the key contributions as follows:

- We propose the Self-Controlled Memory (SCM) system to unleash infinite-length input capacity for LLMs, which can decide when and how to introduce memory information to generate the response.
- We contribute a dataset to evaluate the effectiveness of SCM in three tasks: long-term dialogues, book summarization, and meeting summarization.
- Our proposed SCM system does not require any modification or fine-tuning of LLMs, making it highly scalable in terms of memory stream.

2 Methodology

We describe the details of our proposed Self-Controlled Memory (SCM) system in this section. As shown in Figure 2, our SCM system comprises three modules: (1) a large language model agent (Section 2.1) serving as the backbone of the system, (2) a memory stream (Section 2.2) storing agent memories, and (3) a memory controller (Section 2.3) updating memories and determining when and how to utilize memories from memory stream.

2.1 Large Language Model Agent

The large language model agent serves as the core component of our SCM system and is required to generate coherent and accurate responses based on carefully designed instructions (e.g., question answering, summarization,). We employ two powerful LLMs, *text-davinci-003* and *gpt-3.5-turbo*, as agents in our SCM system, respectively. In this paper, we utilize the terms *davinci003* and *turbo* to refer to *text-davinci-003* and *gpt-3.5-turbo-0301* as backbone model correspondingly, for the purpose of brevity.

2.2 Memory Stream

The memory stream stores all historical memory items and can easily achieve high-speed access through cache storage technologies such as Redis or vector databases like Pinecone [2]. Specifically, each memory item consists of (1) an interaction index, (2) an observation, (3) a system response, (4) a memory summarization (refer to the next paragraph for elaboration) and (5) an interaction embedding that illustrates the current interaction semantics. To obtain the interaction representative embedding, we combine the textual content of both the observation and system response and utilize the *text-embedding-ada-002* model [3] to get the embedding vector of

[2]https://www.pinecone.io/
[3]openai-text-embedding document

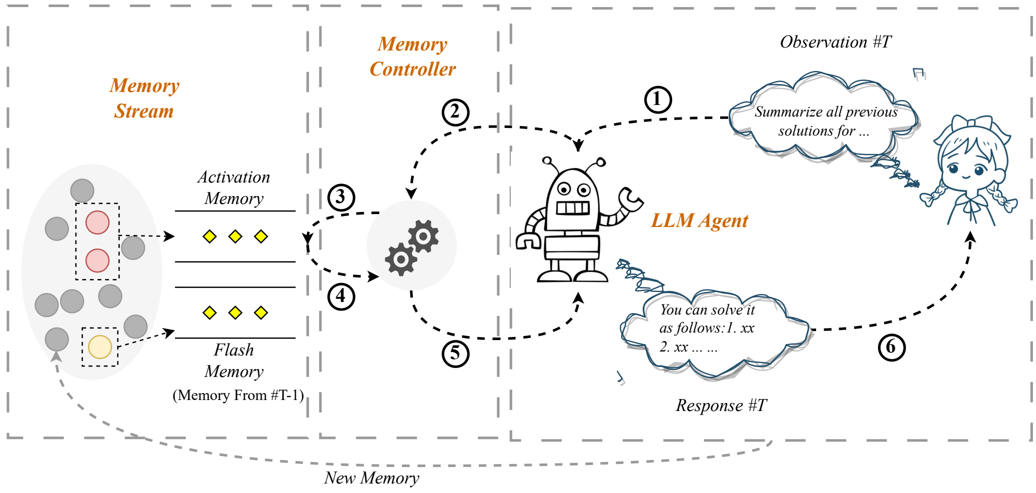

Figure 2: The workflow of our proposed Self-Controlled Memory(SCM) system, where numbers 1-6 represent the six explicit steps of one iteration with new observation #T. These steps are (1) Input Acquisition; (2) Memory Activation; (3) Memory Retrieval; (4) Memory Reorganization; (5) Input Fusion; (6) Response Generation.

the text. When memory retrieval is necessary, the memory stream retrieves and returns two kinds of items: Activation Memory, which stores related historical memories, and Flash Memory, which stores interaction memories from the previous turn $T-1$.

Memory Summarization Memory summarization plays a vital role in processing lengthy inputs, where a single interaction or dialogue turn can consist of more than 3,000 tokens. Obtaining the key information of individual turns through turn summarization is a non-trivial task when attempting to integrate multi-turn information within a limited contextual window. Figure 3 shows the English prompt that is specifically designed for memory summarization in individual interactions (i.e., dialogue tasks). In addition, other language versions of the prompt can be found in § **??**.

Memory Retrieval In our study, we employ an empirical approach of concatenating the observation summary and system response summary (i.e., the memory summarization result of each item) to derive semantic representations for individual items. This concatenation is necessary due to the potential significant variation in length between the observation and system response within the memory stream. Such variation can create an imbalance in the semantic information captured solely from the original texts. Consequently, directly utilizing semantic vectors obtained from the original texts may not effectively balance the semantic information between observations and system responses.

Below is a conversation between a user and an AI assistant. Please provide a summary of the user's question and the assistant's response in one sentence each, with separate paragraphs, while preserving key information as much as possible.

Conversation:

User: {user input}
Assistant: {system response}

Summary:

Figure 3: Prompt for dialogue memory summarization.

2.3 Memory Controller

This section focuses on the central component: the memory controller, and its workflow is illustrated in Figure 4. The primary objective behind the design of the memory controller is to introduce the minimum necessary information to avoid excessive noise that may disrupt the model's performance.

Specifically, this can be divided into three scenarios for discussion. Firstly, not all observations, also referred to as user input or instruction, require access to historical memory. For instance, the user instruction "Tell me a joke" does not necessitate retrieving the user's historical memory. However, certain user input, such as "Do you remember the conclusion we made last week on the fitness diets" requires the retrieval of past memories.

Secondly, the amount of memory can be enormous, ranging from hundreds to thousands or even

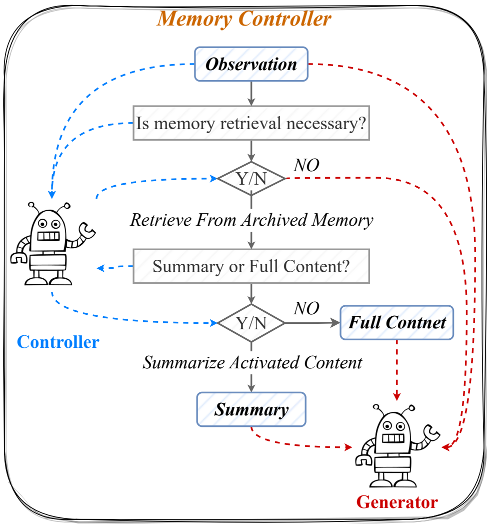

Figure 4: Workflow of the Memory Controller.

tens of thousands. A controller is needed to retrieve and filter the memory.

Thirdly, given the limited input length of the model, it becomes necessary for the controller to determine whether to employ the full content of the memory or a summary of it. The original full text can be excessively long and may exceed the model's maximum length capacity.

In the subsequent subsections, we present the detailed workflow of the memory controller, which considers each of the aforementioned scenarios.

2.3.1 Memory Controller Workflow

The controller is also a language model, which controls the entire process by self-asking two questions:

1. Is it necessary to activate memories given current user input?

2. Can the current user input be answered correctly using only the summary of memory?

Activate Memories To address the first question, we have devised a prompt for the controller to determine whether or not to activate memories. This prompt is illustrated in Figure 5. If the model responds with "yes(A)", relevant memories will be activated to provide an answer to the current question. During the process of retrieving memories, we employ the current observation as a query and assess the rank score of each memory based on two factors: **recency** and **relevance**. The recency factor places high importance on memory items that have been accessed recently, emphasizing the agent's attention on the most recent interactions. Furthermore, the relevance score of each memory is computed by calculating the cosine similarity between the current query embedding and the memory embedding. The final rank score of each memory is determined by summing its recency and relevance scores: $rank\_score = recency\_score + relevance\_score$. Depending on the length limit, we select the top k memories with the highest rank scores as the activated memories. Here, the value of k can range from 3 to 10.

Use Summary To address the second question, we have designed a prompt to evaluate whether the user's question can be answered using the turn summary. This prompt is depicted in Figure 6. We perform this evaluation for each activated memory that exceeds 800 tokens. It is important to highlight that the summary assessment takes place only when the total number of activation memory tokens

Given a user command, determine whether executing the command requires historical or previous information, or whether it requires recalling the conversation content. Simply answer yes (A) or no (B) without explaining the information:

User Command: {User Input}

Answer:

Figure 5: English prompt for the necessity of using memory.

Given the conversation content and the user question, please answer the command question.

Conversation Content: ```{content}```
User Question: ```{query}```

Command Question: Based on the conversation content, can the user question be answered by conversation content? Respond with (A) for yes, (B) for no.

Please strictly follow the format below to answer the questions:
[Answer]: (A) / (B).

Figure 6: English prompt for whether or not to use the summary of memory.

Here is a conversation between a user and an AI assistant. Please answer the user's current question based on the history of the conversation:

History of the conversation: {history_turn}

Previous conversation: {last_turn}

###

User: {user_input}
Assistant:

Figure 7: English Prompt of ultra-long dialogue generation.

surpasses 2000. If the assessment yields a positive result, indicating that the summary can indeed answer the user's question, we utilize the memory summary to represent that specific memory.

2.4 System workflow

Our SCM system workflow comprises six explicit steps, which are outlined below in a sequential manner:

1. *Input Acquisition:* The agent receives an observation in turn T, either through direct input or from an external source.

2. *Memory Activation:* Based on the current observation, the memory controller determines whether it is necessary to activate memory for the current user input.

3. *Memory Retrieval:* We utilize the observation as a query to identify top K-ranked memories memories.

4. *Memory Reorganization:* The controller will determine whether to use the original or summarized memory directly. Then, the system will combine the memories retrieved in a structured manner to serve as background information for response generation at this point.

5. *Input Fusion:* We carefully design a prompt that fuses the restructured memory with the present observation to serve as the model's input. Figure 7 shows the details.

6. *Response Generation:* The model generates a response based on the previous step result and incorporates the current interaction, including observation and response, into the memory stream.

	Dialogue	**Book**	**Meeting**
#Instances	18	10	20
Max tokens	34k	2M	50k
Total tokens	420k	8M	632k
Max turn	200	-	80
Language	En+Zh	En+Zh	Zh

Table 1: Evaluation dataset statistics. 2M means 2 million token count.

3 Experiments

Our experiment focuses on evaluating the effectiveness and robustness of the SCM system in three various task scenarios: long-term dialogues, book summarization, and meeting summarization. Specifically, we want to explore whether the SCM system can facilitate long-term, continuous multi-turn conversations in non-dialogue models. Furthermore, we intend to investigate whether memory-enhanced models can offer more comprehensive coverage and create coherent contextual logic summaries compared to traditional models when tackling long text summarization scenarios.

3.1 Evaluation Dataset

In order to evaluate the SCM performance across various scenarios, we collect open-source data from ShareChat [4], online book websites [5], and the VCSUM dataset (Wu et al., 2023). Then, we utilize human annotation to create probing questions and summaries for the collected data. The dataset statistics are illustrated in Table 1.

3.2 Quantitative Study

To quantitatively compare the performance of the models, 105 test questions are annotated based on the dialogue data and categorize them into two groups: single-turn related questions and multi-turn related questions. Additionally, for evaluating the two summarization tasks, we compare the performance of SCM variants with the baseline model.

Models To ensure a fair comparison, we have selected specific model variants for experimental analysis:

1) SCM $_{\text{turbo}}$: Utilizing *gpt-3.5-turbo-0301* as the backbone of our SCM system.

[4] ShareChat
[5] Gutenberg

Model Name	Answer Acc.	Memory Retrieval Recall	Single Turn Acc.	Multi Turn Acc.
SCM turbo	68.3	93.5	73.5	64.3
SCM davinci003	**77.1**	**94.0**	**79.6**	**75.0**
w/o memory controller	59.3 (-17.8)	93.8 (-0.2)	71.7 (-7.9)	49.4 (-25.6)
w/o flash memory	72.9 (-4.2)	93.9 (-0.1)	74.6 (-5.0)	74.8 (-0.2)
w/o activation memory	10.5 (-66.6)	0.0 (-94.0)	18.2 (-61.4)	0.0 (-75.0)

Table 2: Long-term dialogue evaluation results. The total number of probing questions is 105, including Chinese and English, with 49 single-turn and 56 multi-turn related questions. The lower part of the table is the ablation experiment of our system.

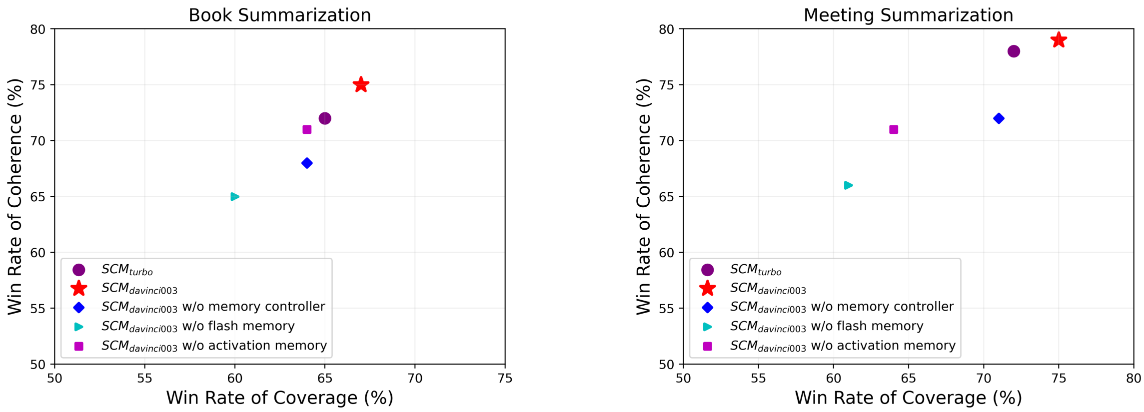

Figure 8: The win rate of SCM variants against baseline model, RecursiveSum (Wu et al., 2021) by OpenAI, in both book and meeting summarization tasks. The figure also shows a comparison of the results of the SCM system and its various component ablations.

2) SCM davinci003: Utilizing *text-davinci-003* as the backbone for SCM system.

3) SCM davinci003 *w/o* memory controller: Remove the memory controller and concatenate the full retrieved content. If the token length of the concatenated history exceeds 2500, truncate it.

4) SCM davinci003 *w/o* flash memory: Remove the flash memory (short-term memory), which contains the latest information.

5) SCM davinci003 *w/o* activation memory: Remove the activation memory (long-term memory), which is essential for answering questions involving long-distance dependencies.

Evaluation Metrics Distinct evaluation metrics are utilized for long-term dialogue scenario and two summarization scenario. For long-term dialogue scenario, the performance of our system is assessed based on the following metrics. (1) Answer Accuracy: Evaluates the accuracy of answers to probing questions. (2) Memory Retrieval Recall: Determines if related memory can be successfully retrieved by memory controller. (3) Single Turn Accuracy: Examines the accuracy of answers to probing questions related to individual turns in the conversation history. (4) Multi Turn Accuracy: Similar to single-turn accuracy, but it requires considering the multi-turn history in order to answer these probing questions. Additionally, two metrics, coverage and coherence, are used to evaluate content coverage and plot coherence in summarization tasks. To facilitate a comprehensive comparison, we assess the effectiveness of the model by comparing its win rate to that of the baseline model, namely RecursiveSum (Wu et al., 2021) by OpenAI, which first summarizes small sections of the book and then recursively summarizes these summaries to produce a summary of the entire book.

Dialogue Results Table 2 displays the long-term dialogue results and demonstrates that the SCM davinci003 is superior to the SCM turbo for this particular task. This may be attributed to the SCM turbo's conservative nature, which can lead to hesitation in answering privacy related probing questions. In contrast, the SCM davinci003 is capable of providing quicker and more precise responses. Moreover, we conducted an ablation study to investigate the in-

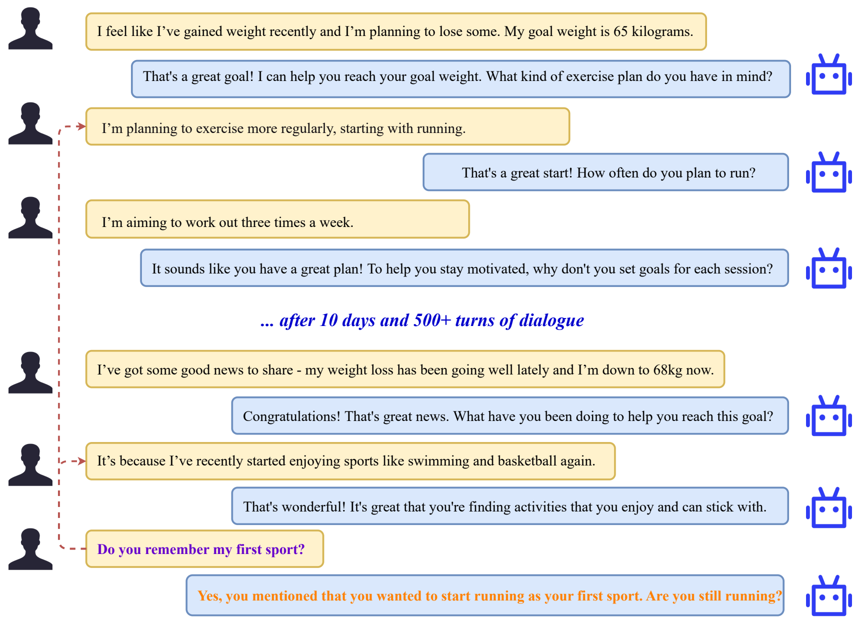

Figure 9: Long-term dialogue example. To answer users' questions, our model can accurately retrieve relevant memories from massive memories and generate accurate responses based on these memories.

dependent effect of each module in SCM system, the results are illustrated in the lower part of Table 2. When the activation memory is removed, the accuracy of the system's responses experiences a significant drop, resulting in an approximate 60% decrease in performance. This is because the majority of probing questions are derived from long-distance dialogue records, which rely on activation memory to retrieve them. What's more, in the absence of activation memory, both memory retrieval recall and multi-turn accuracy have decreased to zero. This further demonstrates the significance of activation memory. However, when flash memory is removed, the performance only experienced a slight drop. This is because flash memory provides fewer clues to answer probing questions, resulting in a minor impact on the final accuracy. Removing the memory controller leads to a greater drop in accuracy for multi-turn related questions compared to single-turn questions. This is because the absence of the memory controller's dynamic memory filtering and use of summaries for efficient input token management results in the concatenation and truncation of all retrieved memories, leading to significant information loss.

Summarization Results Figure 8 illustrates the book and meeting summarization results. Based on the experimental results, we have obtained three conclusions: (1) SCM $_{\text{davinci003}}$ provides better coverage than SCM $_{\text{turbo}}$. (2) SCM $_{\text{davinci003}}$ and SCM $_{\text{turbo}}$ demonstrate comparable coherence performance due to their memory-enhanced mechanism. (3) The SCM system without memory loses contextual dependency and consequently produces unsatisfactory summarization outcomes. It is evident from the model comparison results that SCM $_{\text{davinci003}}$ consistently outperforms SCM $_{\text{turbo}}$ summarizing both books and meetings. This can be attributed to the fact that SCM $_{\text{turbo}}$'s summarization primarily focuses on general principles, whereas it overlooks detailed core plots. In terms of human evaluation, the SCM $_{\text{davinci003}}$ model's results are more favored because of their conciseness, clarity, and richer plot content.

3.3 Qualitative Study

The purpose of this qualitative study is to answer three research questions (RQs). The following experiment evaluates the performance of the SCM $_{\text{davinci003}}$ model without dialogue optimization in comparison to the vanilla ChatGPT model.

> **RQ1**. Can SCM system compete with or even outperform ChatGPT within a specific token limit? **Yes**.

The example in Figure 1 includes 4k tokens, wherein the user inquired about their hobbies, discussed 100+ turns ago with the agent. The SCM system provides an accurate response to the query, demonstrating exceptional memory-enhanced capabilities, as apparent from the observation. In

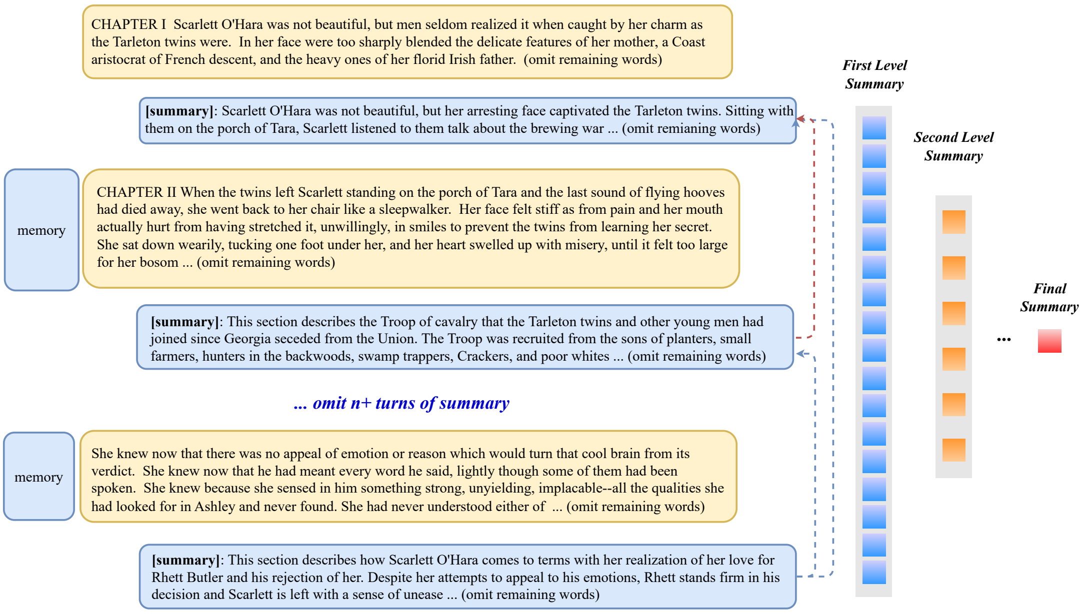

Figure 10: Ultra-long book iterative and hierarchical summarization example from *Gone With The Wind*. Our SCM system initially divides the long text into small text blocks, and then summarize each small text block in sequence. While summarizing the current text block, we will retrieve the flash memory and activation memories to ensure a smoother generation of the summary plot. We then proceed to hierarchically summarize the first level summary until reaching the final summary.

contrast, it appears that ChatGPT is distracted by a considerable amount of irrelevant historical noise.

RQ2. Can SCM system scale to provide accurate responses to users' questions, which are related to historical contexts that date back hundreds or even thousands of turns? **Yes**.

The example presented in Figure 9 illustrates a long-term dialogue comprising over 100 turns. At the outset, the user states that his goal is to reduce weight and intends to initiate a running regime. Subsequently, the user and the model converse daily about progress towards achieving their weight loss goals, among other conversation topics. After over 100 rounds of dialogue, the token length of the conversation has already exceeded 10k tokens. The user then asks the model "Do you remember my first sport?". Our SCM system recalls sports-related information from memory and combines it with the user's current question. Afterwards, the system generates an accurate response.

RQ3. Can SCM demonstrate effective generalization to other lengthy input scenarios?**Yes**.

Figure 10 illustrates an example of summarizing lengthy books and meetings with our SCM system in iterative and hierarchical manner. This lengthy document has been divided into several parts and gradually summarized to obtain the first-level local summary, and then hierarchically summarized to obtain the final summary. In order to maintain context coherence, relevant memories from previous sections will be added to the input text. The conventional method involves dividing lengthy texts into separate smaller text blocks that can be processed by the model. and summarizing each text block independently. However, this method can lose the dependency relationship between paragraphs. Our SCM system facilitates the summarization process by utilizing the related memories, thus establishing substantial coherence between the two summaries. Ultimately, the framework incorporates a divide-and-conquer strategy to generate the final document summary. The final summary provides a comprehensive summary by utilizing information from each document block.

4 Related Work

Large-scale Language Models. Large-scale Language Models (LLMs) are language models trained on massive amounts of text data, using the Transformer (Vaswani et al., 2017) architecture as their foundation. The earliest Transformer-based pre-trained language model was GPT-1 (Radford et al., 2018), BERT (Devlin et al., 2019; Liang et al., 2023) and so on. Subsequently, GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020b) were developed with gradually increasing parameter sizes. GPT-3 has the largest scale, with 175B parameters, along with emergent abilities (Wei et al., 2022b,c), which has attracted the attention of both academia and industry.

Since then, many LLMs have emerged, including LAMBDA (Thoppilan et al., 2022), PaLM (Chowdhery et al., 2022), OPT (Zhang et al., 2022a), LLaMA (Touvron et al., 2023), BLOOM (Workshop et al., 2023), and Qwen (Bai et al., 2023c). One of the most notable works is ChatGPT (OpenAI, 2022), which has achieved remarkable performance and surpassed the boundaries between NLP tasks. However, current LLMs, including ChatGPT, face significant limitations when processing tasks involving extremely long inputs.

Long Text Sequence Processing. Handling long text sequences has been a persistent challenge in natural language processing tasks (Yang et al., 2020; Bai et al., 2023b; Wang et al., 2023; Pi et al., 2022; Bai et al., 2023a). Existing solutions primarily involve replacing the Attention structure during pre-training to reduce computational costs and expanding the pre-training sequence length (Beltagy et al., 2020; Zaheer et al., 2021; Guo et al., 2022; Phang et al., 2022; Dong et al., 2023). Another alternative approach (Press et al., 2022) uses special positional encoding during pre-training to enable the model to learn relative positions and handle longer input texts during inference. However, the generalizability of these methods and their impact on downstream tasks remain uncertain. In the field of long-text summarization, there are many effective methods. Hierarchical or iterative methods have been used by Wu et al. (2021); Zhang et al. (2022b); Cao and Wang (2022); Liang et al. (2022) to handle long texts by decomposing a complex problem into multiple sub-problems. However, these methods fail to capture the relationships among sub-problems.

5 Conclusion

In this paper, we propose a Self-Controlled Memory (SCM) system to extend the input length of any LLMs to an unlimited length and effectively capture useful information from all historical information. This method does not require any training or modification of models. In addition, we annotate an evaluation dataset comprising three tasks. Experimental results demonstrate that SCM allows LLMs, which are not optimized for multi-turn dialogue, to attain comparable multi-turn dialogue capabilities to ChatGPT, and outperform ChatGPT in long document summarization tasks.

References

Jiaqi Bai, Hongcheng Guo, Jiaheng Liu, Jian Yang, Xinnian Liang, Zhao Yan, and Zhoujun Li. 2023a. Griprank: Bridging the gap between retrieval and generation via the generative knowledge improved passage ranking.

Jiaqi Bai, Ze Yang, Jian Yang, Hongcheng Guo, and Zhoujun Li. 2023b. Kinet: Incorporating relevant facts into knowledge-grounded dialog generation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:1213–1222.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. 2023c. Qwen technical report.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020a. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020b. Language models are few-shot learners.

Shuyang Cao and Lu Wang. 2022. HIBRIDS: Attention with hierarchical biases for structure-aware long document summarization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 786–807, Dublin, Ireland. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. 2023. A survey of natural language generation. *ACM Comput. Surv.*, 55(8):173:1–173:38.

Mandy Guo, Joshua Ainslie, David Uthus, Santiago Ontanon, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2022. LongT5: Efficient text-to-text transformer for long sequences. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 724–736, Seattle, United States. Association for Computational Linguistics.

Xinnian Liang, Jing Li, Shuangzhi Wu, Mu Li, and Zhoujun Li. 2022. Improving unsupervised extractive summarization by jointly modeling facet and redundancy. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 30:1546–1557.

Xinnian Liang, Zefan Zhou, Hui Huang, Shuangzhi Wu, Tong Xiao, Muyun Yang, Zhoujun Li, and Chao Bian. 2023. Character, word, or both? revisiting the segmentation granularity for chinese pre-trained language models.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng-Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. 2022. Crosslingual generalization through multitask finetuning.

OpenAI. 2022. Introducing chatgpt.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Jason Phang, Yao Zhao, and Peter J. Liu. 2022. Investigating efficiently extending transformers for long input summarization.

Xinyu Pi, Bing Wang, Yan Gao, Jiaqi Guo, Zhoujun Li, and Jian-Guang Lou. 2022. Towards robustness of text-to-sql models against natural and realistic adversarial table perturbation.

Ofir Press, Noah Smith, and Mike Lewis. 2022. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations*.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F. Christiano. 2020. Learning to summarize from human feedback. *CoRR*, abs/2009.01325.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Vincent Zhao, and etc. 2022. Lamda: Language models for dialog applications.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Bing Wang, Yan Gao, Zhoujun Li, and Jian-Guang Lou. 2023. Know what I don't know: Handling ambiguous and unknown questions for text-to-SQL. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5701–5714, Toronto, Canada. Association for Computational Linguistics.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V Le. 2022a. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022b. Emergent abilities of large language models. *Transactions on Machine Learning Research*. Survey Certification.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. 2022c. Chain of thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*.

BigScience Workshop, :, Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, and etc. 2023. Bloom: A 176b-parameter open-access multilingual language model.

Han Wu, Mingjie Zhan, Haochen Tan, Zhaohui Hou, Ding Liang, and Linqi Song. 2023. Vcsum: A versatile chinese meeting summarization dataset.

Jeff Wu, Long Ouyang, Daniel M. Ziegler, Nisan Stiennon, Ryan Lowe, Jan Leike, and Paul Christiano. 2021. Recursively summarizing books with human feedback.

Jian Yang, Shuming Ma, Dongdong Zhang, ShuangZhi Wu, Zhoujun Li, and Ming Zhou. 2020. Alternating language modeling for cross-lingual pre-training. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):9386–9393.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2021. Big bird: Transformers for longer sequences.

Aohan Zeng, Xiao Liu, Zhengxiao Du, Zihan Wang, Hanyu Lai, Ming Ding, Zhuoyi Yang, Yifan Xu, Wendi Zheng, Xiao Xia, Weng Lam Tam, Zixuan Ma, Yufei Xue, Jidong Zhai, Wenguang Chen, Zhiyuan Liu, Peng Zhang, Yuxiao Dong, and Jie Tang. 2023. GLM-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022a. Opt: Open pre-trained transformer language models.

Yusen Zhang, Ansong Ni, Ziming Mao, Chen Henry Wu, Chenguang Zhu, Budhaditya Deb, Ahmed Awadallah, Dragomir Radev, and Rui Zhang. 2022b. Summ^n: A multi-stage summarization framework for long input dialogues and documents. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1592–1604, Dublin, Ireland. Association for Computational Linguistics.