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

Xinnian Liang[1]*†, **Bing Wang**[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. To address this limitation, we propose the Self-Controlled Memory (SCM) system to unleash infinite-length input capacity for large-scale language models. Our SCM system is composed of three key modules: the language model agent, the memory stream, and the memory controller. The language model agent iteratively processes ultra-long inputs and stores all historical information in the memory stream. The memory controller provides the agent with both long-term memory (archived memory) and short-term memory (flash memory) to generate precise and coherent responses. The controller determines which memories from archived memory should be activated and how to incorporate them into the model input. Our SCM system can be integrated with any LLMs to enable them to process ultra-long texts without any modification or fine-tuning. Experimental results show that our SCM system enables LLMs, which are not optimized for multi-turn dialogue, to achieve multi-turn dialogue capabilities that are comparable to ChatGPT, and to outperform ChatGPT in scenarios involving ultra-long document summarization or long-term conversations. Additionally, we will supply a test set, which covers common long-text input scenarios, for evaluating the abilities of LLMs in processing long documents. [1][2]

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.

Large Language Models (LLMs) offer numerous advantages, but their utility is hindered by two main factors: the maximum input length and the com-

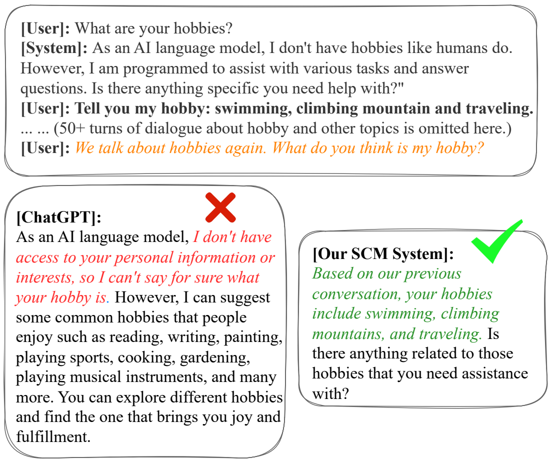

Figure 1: An example comparing ChatGPT and our SCM system. The orange text indicates the user's current question, bold text denotes related historical information, red text indicates an erroneous response from ChatGPT, and green text denotes the correct response from our system.

*The authors contribute equally and the order is random.
†Contribution during internship at ByteDance Inc.
‡Corresponding Author
[1]Working in progress.
[2]https://github.com/wbbeyourself/SCM4LLMs

putational complexity of self-attention during the pre-training phase (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 [3] can miss out on essential 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 present the Self-Controlled Memory (SCM) system, which enables Large Language Models (LLMs) to process text of infinite length without any modification or additional training. The input is partitioned into segments and fed to the LLM as observations (inputs). The SCM expands the LLM with a long-term memory (archived memory), a short-term memory (flash memory), and a memory controller. The archived memory preserves all historical information, while the flash memory captures real-time memory information from previous rounds. The memory controller determines when and how to introduce archived information, allowing the LLM to efficiently handle ultra-long text without sacrificing any essential information.

To evaluate the performance of our system, we integrate the SCM with non-dialogue-optimized LLMs and simulate ChatGPT with success. Our findings indicate that our system outperforms ChatGPT in handling ultra-long inputs or conversations. For summarization tasks, we generate a hierarchical summary of the entire archived memory until the summary length meets the user's specifications. By incorporating information from preceding text into local summaries within the memory, our approach preserves the correlations among the original content, in contrast to the conventional approach of directly generating a hierarchical summary of the entire text. Furthermore, our work is still in progress, and we plan to release a comprehensive evaluation dataset designed for long-text tasks, along with standardized human evaluations to evaluate the effectiveness of different methods.

2 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). 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), Galactica (Taylor et al., 2022), and Pangu (Zeng et al., 2021; Ren et al., 2023). One of the most notable works in this series of research that has attracted widespread industry attention and can be considered a milestone towards Artificial General Intelligence (AGI) is ChatGPT (OpenAI, 2022), which is based on InsctuctGPT (Ouyang et al., 2022) and optimized for multi-turn dialogue ability. ChatGPT 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. This problem has become even more prominent with the advent of pre-training and LLMs, as the fixed input length during pre-training and the high costs of expanding it during the pre-training stage limit the ability to process longer inputs. 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) to handle long texts by decomposing a complex prob-

[3] In this study, we utilize OpenAI *gpt-3.5-turbo-0301*.

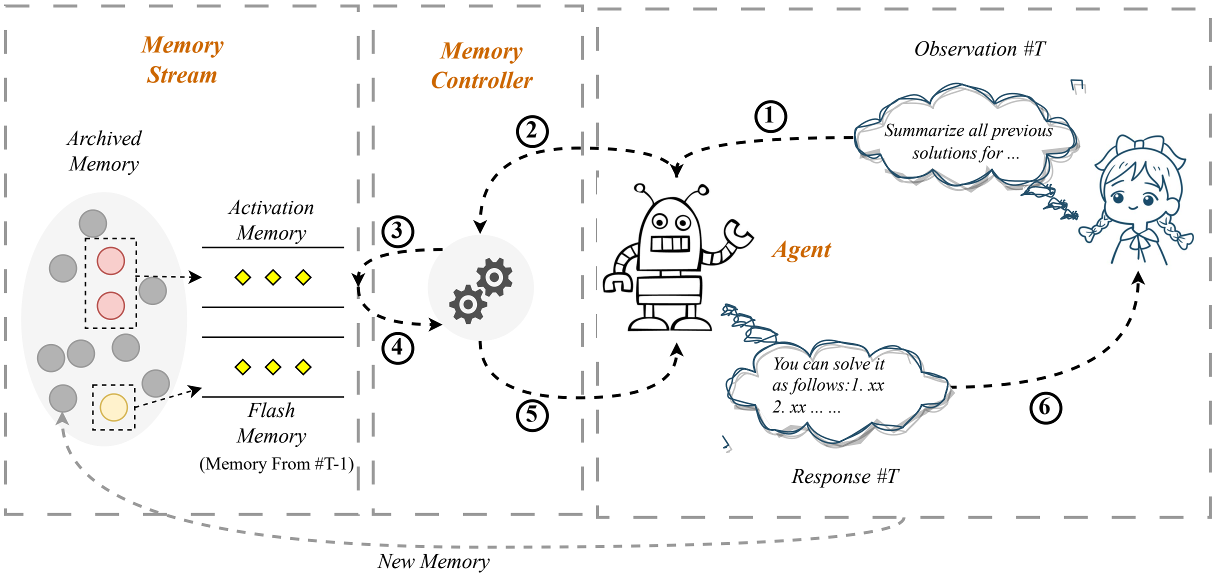

Figure 2: The workflow of our proposed Self-Controlled Memory(SCM) system, where numbers 1-6 represent the sequential process of one iteration with new observation #T.

lem into multiple sub-problems. However, these methods fail to capture the relationships among sub-problems.

3 Methodology

The Self-Controlled Memory (SCM) system proposed in this paper aims to give large-scale language models (LLMs) the capability to store long-term memories, allowing them to process lengthy inputs and retain information after multiple interactions with the user.

3.1 System Overview

In this section, we introduce the workflow of our proposed SCM system. As illustrated in Figure 2, our SCM system comprises three modules, including a language model agent, a memory stream, and a memory controller. The three modules work together to process lengthy documents and provide more accurate and coherent responses. Our system workflow consists of six explicit steps, which are presented as follows:

1. Input Acquisition: The agent receives an observation in turn T (i.e., ultra-long document input or a user question), 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. In the case where memory activation is warranted, relevant memories (for detailed memory information, refer to section § 3.2) is retrieved by executing steps 3 and 4. Otherwise, the process moves directly to step 5. § 3.3.1 provides a comprehensive explanation of the control flow of the memory controller.

3. Memory Retrieval: In this step, we utilize the observation as a query to identify related memories. The score ranking of each memory is computed by considering two dimensions: relevance and recency. With respect to relevance, we evaluate how similar the content of the memory is to the observation. With respect to recency, we consider the time elapsed since the memory was last accessed. Subsequently, we retain the top K-ranked memories.

4. Memory Reorganization: In this step, the controller will determine whether to use the original memory directly or the summarized memory. If summarized memory is chosen, the original memory the will be compressed. § 3.3.2 provides a detailed explanation of the state compression process. Then, the system will combine the memory retrieved in a structured manner to serve as background information for response generation at this point.

5. Input Fusion: In this step, we carefully design a prompt that fuses the restructured memory with the present observation to serve as the model's input. A thorough description is given in § 3.4.

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

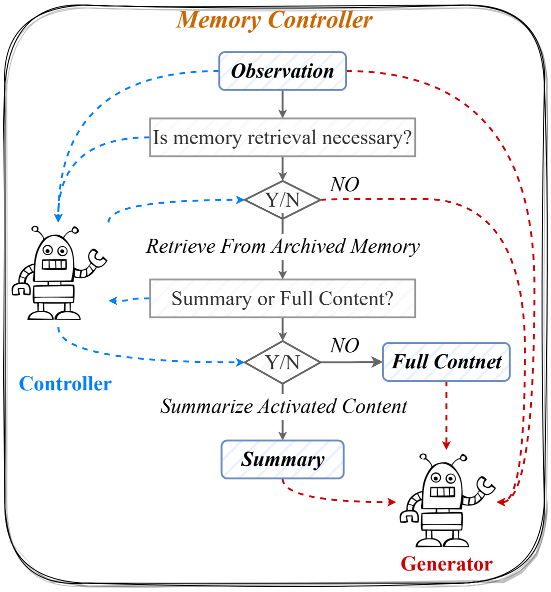

Figure 3: Workflow of the Memory Controller.

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:

Command: [User Input]

Figure 4: English prompt for the necessity of using memory.

Given a user command, determine if it can be executed correctly based solely on the summary historical information provided. Simply answer yes (A) or no (B), without explaining the information.

Command: [User Input]

Figure 5: English prompt for whether or not to use the summary of memory.

and response, into the memory stream. Please refer to § 3.4 for further details.

3.2 Memory Stream

This section provides an overview of the internal structure of memory stream. The memory stream stores all historical memory items in a designated location named as the archived memory center, which can easily achieve high-speed access through cache storage and access tools such as Redis or Pinecone[4]. Each memory item consists of an interaction index, an observation, a system response, and an interaction embedding that illustrates the current interaction semantics. In addition, The Activation Memory stores the retrieved memory set, and the Flash Memory indicates the memory of Turn $T-1$.

3.3 Memory Controller

This section discusses the reasons for using the memory controller and its workflow, as illustrated in Figure 3. There exist three fundamental reasons. Firstly, not all observations, also referred to as user input or instruction, require access to historical memory usage. For example, the user instruction "Tell me a joke" does not require the retrieval of the user's history memory. However, certain user input such as "Do you remember the conclusion we made last week on the fitness diets" requires retrieving past memories. The second reason is that the amount of memory can be enormous, ranging from hundreds to thousands or even tens of thousands. A controller is needed to retrieve and filter the memory. The third reason is that the input length of the model is limited, and a controller is needed to choose between using the full text of the memory or a summary of the memory, as the original text can be long and may exceed the maximum length of the model. The next two subsections present the details of the controller's workflow and state compression implementation, respectively.

3.3.1 Memory Controller Workflow

The core of the controller in terms of process control is to ask two questions of the agent:

1. Is it necessary to use memory to accurately answer when executing user commands?

2. Can user commands be executed normally using only the summary of memory?

The first question prompt is shown in Figure 4, while the prompt for the second question is shown in Figure 5. Other language versions of the prompt can be found in § A.1.

If the controller determines the necessity of utilizing historical memory, memory retrieval should be carried out. While retrieving memories, we use the current observation (i.e. user instruction) as a query and evaluate each memory's rank score based on two factors: Recency and Relevance. Recency highly prioritizes memory items accessed

[4] Pinecone: https://www.pinecone.io/

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 6: Prompt for dialogue memory summarization.

recently, reinforcing the idea that the agent's attention remains on the states of latest interactions. The relevance factor assigns a higher score to memory items that are related to the current observation. In our implementation, we created an embedding vector for the text description of every memory through the use of a language model[5]. The cosine similarity between the embedding vector of the memory and that of the query observation is calculated to determine relevance. The rank score of each memory is the sum of its recency and relevance scores: $rank\_score = recency\_score + relevance\_score$. Depending on the length limit, we designate the top k memories with the highest rank scores as activated memories, where k varies between 3 and 10.

If the controller determines that the employment of a summary can allow for the normal execution of instructions, then it is necessary to summarize the current extracted memories. The specific information regarding the memory summarization is located in the following subsection.

3.3.2 Memory Summarization

Memory summarization is a crucial aspect in scenarios such as document summarization, where a single interaction or dialogue turn can have a token length exceeding 3000. It enables stacking multiple memories into an activated memory section. Figure 6 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 § A.3.

[5]We used OpenAI embedding model *text-embedding-ada-002*

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 of Related Turn]

Previous conversation:

User: [previous user input]
Assistant: [previous system response]

###

User: [current user input]
Assistant:

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

3.4 Response Generation

As mentioned in the introduction, SCM allows non-dialogue-optimized LLMs to emulate ChatGPT. This necessitates providing explicit instructions to prompt LLMs to use activated and flash memory to generate the desired response. Figure 7 shows an English prompt intended for extremely long multi-interaction dialogues, with other language versions included in the § A.2.

4 Experiments

4.1 Settings

Our framework is preliminarily evaluated in two scenarios: ultra-long dialogues and ultra-long document summarization. We conduct experiments to answer three research questions (RQs).

- **RQ1**. Can SCM system compete with or even outperform ChatGPT within a specific token limit?
- **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?
- **RQ3**. Can SCM demonstrate generalization to other scenarios, including long document summarization?

The following experiment evaluates the performance of the *text-davinci-003* model without dia-

Figure 8: Ultra-long dialogue example.

logue optimization in comparison to the *ChatGPT-Turbo* model. The evaluation focuses on the tasks of ultra-long dialogues and ultra-long document summarization.

4.2 Qualitative Study

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

The example in Figure 1 includes 4000 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 contrast, it appears that ChatGPT was distracted by a considerable amount of irrelevant historical data.

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 8 illustrates a ultra-long dialogue comprising over 500 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 ten days, the length of the dialogue reaches 10,000 tokens. The user then asks the model "Do you remember my fir". Our SCM system accurately responds to this question.

RQ3. Can SCM demonstrate generalization to other scenarios, including long document summarization? **Yes**.

Figure 9 illustrates an instance of an incredibly lengthy document summary. Specifically, the report pertains to the unveiling of GPT-4 by OpenAI. Summaries exceeding 4,000 characters pose a challenge for conventional models, thus necessitating the splitting and individual summarization of document parts, which are then united. Nonetheless, this method can lose the dependency relationship between paragraphs. Our framework utilizes a iterative summarization procedure. While summarizing paragraphs, our approach relies on earlier relevant summary memories to generate more precise summaries. Ultimately, the framework incorporates a divide-and-conquer strategy to generate the final document summary. The final summary obtained through the divide-and-conquer method provides a comprehensive summary by utilizing information from each document block. Furthermore, our iterative summary paradigm contains a memory-enhancement feature that allows topic-specific summaries to be generated by integrating a

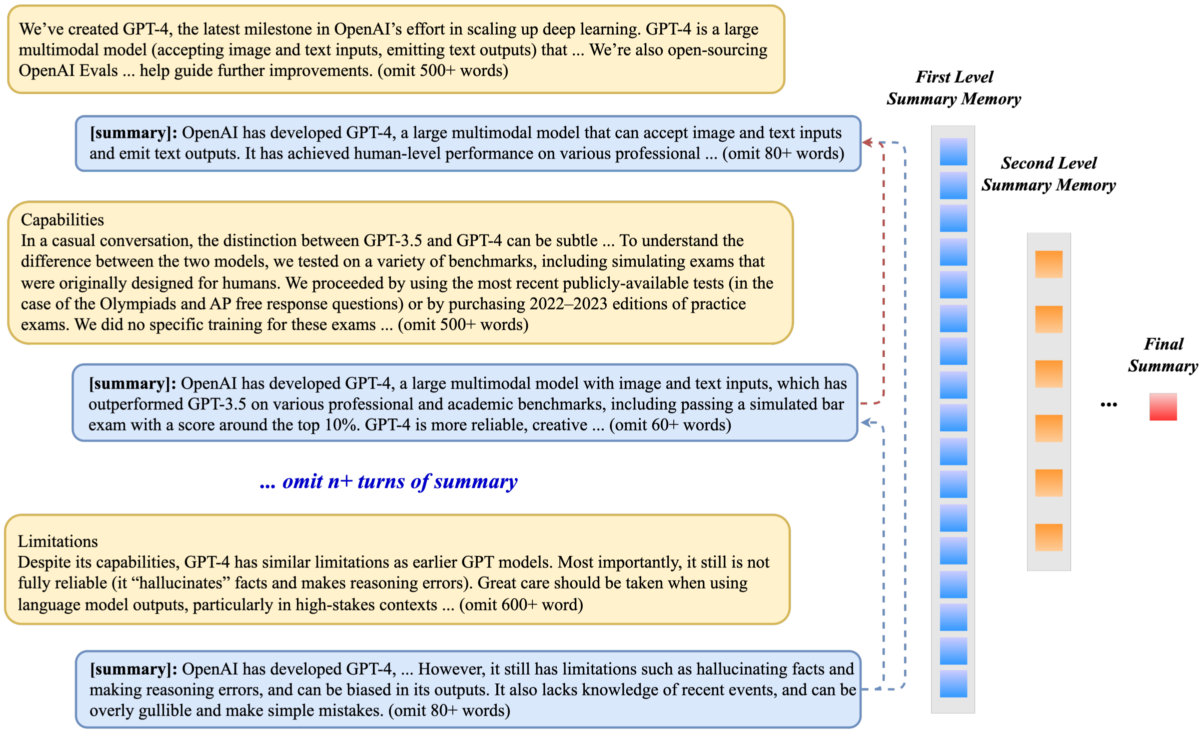

Figure 9: Ultra-long iterative and hierarchical summarization example.

question-asking methodology during single block summarization. For instance, if a user poses a question such as "Please provide a summary of the technical details and evaluation findings of GPT4 in image processing", the model will access prior summary memories and extract the relevant content. We will continue to improve this aspect in the future.

5 Limitations and Risks

Limitations A lack of appropriate datasets for evaluating the handling of extremely lengthy texts has resulted in our model being validated solely through manual verification. This method, however, is inadequate for evaluating different scenarios comprehensively and objectively. Therefore, we aim to construct a specific test set that incorporates various key indicators essential for processing long texts in diverse settings. This test set will be accompanied by a manual evaluation standard to enable a more equitable comparison with relevant methods. Moreover, we will assess the efficacy of our system on more open-source models that possess single-turn instruction comprehension capability.

Risks Our system has the capability to attach to any LLMs, which may be prone to factual errors, delusions, toxic language, and malicious responses. Consequently, we restrict the usage of our system to academic research purposes for now.

6 Conclusion and Future Work

In this paper, we propose a Self-Controlled Memory (SCM) system to extend the input length of any LLMs model to an unlimited length and effectively capture useful information from all historical information. This method does not require any training or modification of models and has strong applicability. We validated the effectiveness of our method through manual evaluation of the *ChatGPT* and the *Text-DaVinci-003* model based on our system, demonstrating superior performance in certain aspects of long-text scenarios compared to ChatGPT.

Our future work will focus on releasing a comprehensive test set and its manual evaluation criteria, and testing our system on various open-source models currently available.

References

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.

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.

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.

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.

Xiaozhe Ren, Pingyi Zhou, Xinfan Meng, Xinjing Huang, Yadao Wang, Weichao Wang, Pengfei Li, Xiaoda Zhang, Alexander Podolskiy, Grigory Arshinov, Andrey Bout, Irina Piontkovskaya, Jiansheng Wei, Xin Jiang, Teng Su, Qun Liu, and Jun Yao. 2023. Pangu-Σ: Towards trillion parameter language model with sparse heterogeneous computing.

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.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science.

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.

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.

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

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)*.

Wei Zeng, Xiaozhe Ren, Teng Su, Hui Wang, Yi Liao, Zhiwei Wang, Xin Jiang, ZhenZhang Yang, Kaisheng Wang, Xiaoda Zhang, Chen Li, Ziyan Gong, Yifan Yao, Xinjing Huang, Jun Wang, Jianfeng Yu, Qi Guo, Yue Yu, Yan Zhang, Jin Wang, Hengtao Tao, Dasen Yan, Zexuan Yi, Fang Peng, Fangqing Jiang, Han Zhang, Lingfeng Deng, Yehong Zhang, Zhe Lin, Chao Zhang, Shaojie Zhang, Mingyue Guo, Shanzhi Gu, Gaojun Fan, Yaowei Wang, Xuefeng Jin, Qun Liu, and Yonghong Tian. 2021. Pangu-α: Large-scale autoregressive pretrained chinese language models with auto-parallel computation.

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.

A Prompt List

A.1 Prompt For Memory Controller

给定一个用户指令，判断执行该指令是否需要历史信息或者上文的信息，或者需要回忆对话内容，只需要回答是(A)或者否(B)，不需要解释信息：

指令：[用户输入]

Figure 10: Chinese Prompt of memory controller.

A.2 Prompt for Dialogue Generation

以下是用户和人工智能助手的对话，请根据历史对话内容，回答用户当前问题：

相关历史对话：

[历史轮对话内容]

上一轮对话：

[上一轮对话内容]

###

用户：[用户问题]

助手：

Figure 11: Chinese Prompt of ultra-long dialogue generation.

A.3 Prompt for Dialogue State Compression

以下是用户和人工智能助手的一段对话，请分别用一句话写出用户摘要、助手摘要，分段列出，要求尽可能保留用户问题和助手回答的关键信息。

对话内容：

用户：[用户输入]
助手：[系统回复]

摘要：

Figure 12: Chinese Prompt of ultra-long dialogue summarization.