# Long Context Understanding using Self-Generated Synthetic Data

Jerry Li [1] Subhro Das [2] Aude Oliva [1 2] Dmitry Krotov [2] Leonid Karlinsky [2] Rogerio Feris [2]

## Abstract

Can a large language model (LLM) learn to understand longer context using self-generated instruction tuning data? Such capability would be important not only for long context modeling, but also to avoid issues related to licensing or copyright that may arise when relying on a separate teacher model to generate long context training data. In this paper, we address this challenge by proposing a novel set of diverse synthetic tasks that enable an LLM to create long context instruction tuning data in a scalable manner. This data is then used by the same LLM to compress its activations (and hence extend its context length) by learning a battery of low-rank adapters (LoRA), where each adapter is trained to focus on a specific compression rate. During inference, the LoRA compression experts are dynamically selected according to the length of the input. We showcase the effectiveness of our approach in the LongBench evaluation, covering tasks such as question-answering, summarization, few-shot learning, and code completion. We also plan to make our long context instruction data available to the community, which will be a useful resource for practitioners.

## 1. Introduction

Numerous real-world applications require large language models to process long input sequences, including question-answering on long documents, chatbots that make use of historical user data, in-context learning with a large set of demonstrations, and repo-level code understanding. Transformer models, however, struggle to handle long context windows, as memory and computation complexity scale quadratically with the input length, due to the self-attention mechanism (Tay et al., 2022). While many promising solutions have achieved great strides in tackling this problem (Gu & Dao, 2023; Munkhdalai et al., 2024; Liu et al., 2023; Zhang et al., 2024; Peng et al., 2023; He et al., 2024), effectively extending the context length of LLMs while keeping their original capabilities for short context inputs remains an open problem (Liu et al., 2024).

In this work, we address the long context problem through the lens of context compression (Zhang et al., 2024; Mu et al., 2024; Chevalier et al., 2023) and study, for the first time, the question whether LLMs can learn to compress long input sequences (and therefore extend their own context window) using self-generated synthetic data.

Our motivation to study this problem stems from two reasons. First, synthetic data offers a scalable way to produce long context instruction-following data. In fact, such data, while commonly available for short context windows, is *scarce* for long input sequences, given that it is very costly to manually curate high-quality text data with long-range dependencies for instruction tuning. Second, using synthetic data from the *same LLM* avoids problems related to licensing, copyright, data protection, and ethical issues that may be present when distilling data from a separate, stronger teacher model. Additionally, using data from a teacher model with longer context abilities also introduces dependency on the teacher model and if an issue with it is discovered in the future (e.g., personal identifiable information crawled in its training) and the teacher needs to be deprecated, the student model needs to go down as well.

Traditional approaches for short context synthetic data generation based on a teacher model (Sudalairaj et al., 2024; Mitra et al., 2023) cannot be directly applied in our setting, since the base LLM does not have the extended context length we are seeking for. Our proposed approach consists of a novel set of synthetic tasks tailored for long-context instruction following, which aggregate multiple short context outputs synthetically generated by the base LLM into long sequences with respective instructions (section 2.1). The instructions cover diverse tasks related to knowledge, reasoning, and code, and are designed to make the model access information in a long context and reason about long-range dependencies. We plan to publicly release the data generated by our synthetic tasks (with up to 10 million context length).

Our pipeline for long context data generation enables us to generate synthetic data at arbitrary lengths. This allows

[1]Massachusetts Institute of Technology, Cambridge MA, USA [2]MIT-IBM Watson AI Lab, IBM Research, Cambridge MA, USA. Correspondence to: Rogerio Feris <rsferis@us.ibm.com>.

*Proceedings of the 1st Workshop on Long-Context Foundation Models*, Vienna, Austria. 2024. Copyright 2024 by the author(s).

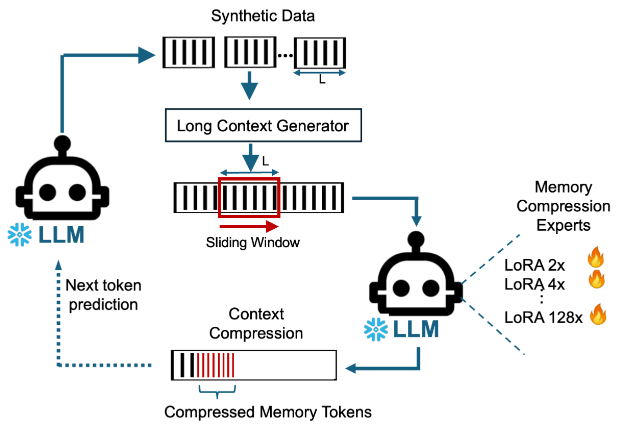

*Figure 1.* Our approach relies on LLM self-generated synthetic data to produce long context training examples, which are in turn compressed by the same LLM using a sliding window along with a battery of LoRA compression experts, which are dynamically selected according to the length of the input. The compressed long context (memory tokens) can fit in the original context window of the LLM while capturing long-range information and is used for next token prediction.

us to train specialized Low-Rank Adapter models (LoRA) (Hu et al., 2021) focusing on specific context length ranges, which we call *context experts* (section 2.2). We build upon the work of (Zhang et al., 2024) for context compression, and show that using multiple context experts spanning different compression rates yields more accurate results than using a single compression model. Additionally, since our architecture uses LoRA as opposed to a copy of the multi-head attention module, our network has much fewer parameters than the model of (Zhang et al., 2024). Figure 1 shows a high-level overview of our approach.

We show the effectiveness of our proposed approach on LonBench (Bai et al., 2023), including tasks for single and multiple document question-answering, summarization, few-shot learning, and code completion. In summary, the main *contributions* of our work are listed as follows: (1) We investigate the problem of self-compression, where the context length of an LLM is compressed using a model trained on its own self-generated synthetic data. (2) A set of diverse synthetic tasks is proposed to generate instruction-following data for long context modeling. We will release our generated data (with up to 10 million context length) to the community. (3) We perform compression of long inputs through a battery of LoRA context experts trained on synthetic data at multiple compression rates, which are dynamically selected during inference according to the length of the input.

## 2. Approach

### 2.1. Synthetic Tasks for Long Context Modeling

We investigate the ability for an LLM to learn to compress long contexts by training on its own self-generated synthetic data. When designing the synthetic training dataset, we target two objectives: task diversity and length flexibility. Specifically, our dataset encompasses a set of six diverse tasks which train the model to not only remember fine details but also reason across a long context, and our approach can easily be adapted to synthesize data of arbitrary desired length.

To construct the synthetic samples, we begin with a list containing 119,034 nouns in the English language mined from WordNet (Fellbaum, 2010). Next, we prompt the base LLM to generate a passage about each noun. The noun serves as a random seed to produce a wide variety of passages, and we also vary the prompt to cover diverse genres from academic papers to sci-fi stories. Let this corpus of synthetic passages be $P = \{p_i\}$ for each noun $i$. The remaining steps to synthesizing the long-context training samples are task-specific:

1. **Single-Doc QA**: We sample a passage $p_i \in P$ and prompt the base LLM to generate a highly specific question-answer pair $(q_i, a_i)$ about the passage. We then construct the long context $c_i$ by sampling a set of some $n$ additional distracting passages $\{p_{distract}\} \subset P$ and placing the relevant passage $p_i$ somewhere at random: $c_i = \text{Shuffle}(p_i, \{p_{distract}\})$. The training sample is then constructed as follows:

   Prompt: "Answer the question based on the passages."

   Input: Concat$(c_i, \text{Prompt}, q_i)$

   Output: $a_i$

2. **Summarization**: Instead of working with a single passage as in QA, we ask the LLM to summarize multiple passages. Specifically, we sample a set of passages $\{p_{sum}\} \subset P$ and prompt the base LLM to generate a summary for each passage, $\{s_{sum}\}$. Distracting passages are added at random to form the long context as before, $c_{sum} = \text{Shuffle}(\{p_{sum}\}, \{p_{distract}\})$.

   Prompt: "Write a summary for Passage $\{index_1\}$, the passage about $\{noun_1\}$; a summary for Passage $\{index_2\}$..."

   Input: Concat$(c_{sum}, \text{Prompt})$

   Output: Concat$(\{s_{sum}\})$

3. **Multi-Doc Sim-Diff**: For this task, we prompt the LLM to generate two very similar passages $r_i^1, r_i^2$ about a noun $i$ which differ in a few key details, and then prompt it again to identify similarities and differences $d_i$. The long context is formed by $c_i = \text{Shuffle}(r_i^1, r_i^2, \{p_{distract}\})$.

   Prompt: "Identify the similarities and differences in the two passages about $\{noun\}$."

   Input: Concat$(c_i, \text{Prompt})$

   Output: $d_i$

4. **Multi-Doc QA**: We prompt the LLM to generate two different passages $s_i^1, s_i^2$ about a noun $i$ which are linked by a few key details, and then prompt it again to generate a question-answer pair $(q_i, a_i)$ which requires knowledge of both passages to answer correctly. The long context is formed by $c_i = \text{Shuffle}(s_i^1, s_i^2, \{p_{distract}\})$.

   Prompt: "Answer the question based on the passages."

   Input: $\text{Concat}(c_i, \text{Prompt}, q_i)$

   Output: $a_i$

5. **Multi-Doc Logic**: We prompt the LLM to generate two facts $f_i^1, f_i^2$ about a noun $i$ which follow the logical structure A implies B and B implies C, respectively, and then generate a question-answer pair $(q_i, a_i)$ about the relationship between A and C. The training sample is constructed identically to the Multi-Doc QA task with $f_i^1, f_i^2$ replacing $s_i^1, s_i^2$.

6. **Code**: We prompt the LLM to generate Python code via alternating between 'invent & initialize a set of variables', 'invent a function', and 'call a function' action steps. Instructions are generated by prompting the same LLM following the same sequence of random policy steps for each synthetic code and providing grounded code snippets with the prompts (recorded during code synthesis). Example prompts for different steps include: 'Explain how the variable {v} is initialized?', 'Ask 5 diverse questions about this function {f}', answer the generated 5 questions in the function context, 'Summarize/Explain the function {f}'. Additionally, we follow the recorded tree of function calls to produce mechanistically QA of the form: 'Which function produced the variable {v}?' and 'List all the variables that participated in computing {v}' requiring long-context trace of a variable to its source.

Observe that by simply adjusting the number of distracting passages $n$, our method can be adapted to easily synthesize data of arbitrary length. For our experiments, we generate 6K synthetic training samples - 1K per task - where each sample has a context length between 1K and 8K tokens. The length distribution is uniform to prevent any training bias. We will also release additional datasets to the community covering the following context lengths: 0-8K, 8K-16K, 16K-32K, 32K-64K, 64K-128K, 128K-256K, 256K-512K, 512K-1M, 1M-2M, 2M-4M, 4M-10M.

### 2.2. Compression using LoRA Context Experts

We use the synthetic data described in the previous section to learn a context compression model that condenses the LLM's raw activations into a compact set of *memory tokens*, which allow the LLM to perceive a longer context within a limited context window. Compared to fine-tuning the model with long context data, this idea has the advantage of mantaining the parameters of the LLM unchanged and therefore retaining its original capabilities.

In this vein, our work builds upon the activation beacon method (Zhang et al., 2024), which first splits the long context into multiple intervals, and then employs a sliding window to sequentially process each interval at a time (see Figure 1). For each interval, a set of memory tokens (or beacon tokens) are added, which are prompted to compress the raw LLM activations of that particular interval. When moving to the next interval, the raw activations of the previous interval are discarded and replaced by the corresponding memory tokens with compressed activations. As a result of this compression, the memory tokens and the prompt can fit in the context window of the base LLM while capturing information from a much longer input.

Different from (Zhang et al., 2024), which clones and unfreezes the parameters of the multi-head self-attention (MHA) for compressing the activations, we instead posit that having multiple activation compression experts operating at different context length ranges is a more sensible design choice. In other words, we propose to learn a battery of compression models, each one tailored for a particular compression rate (2x, 4x, 8x, ..., 1024x). During inference, one of these models is selected according to the length of the input, so that the compressed activations fit in the LLM context window. As verified by our experiments, better compressor models can be obtained if their weights are specialized to particular compression rates, as opposed to having a single model for all compression rates as in (Zhang et al., 2024). On the other hand, cloning multiple MHA parameters would not be feasible in terms of memory footprint. We address this issue by training multiple LoRA adapters, with each adapter focused on a particular compression rate. The LoRA compression experts are trained by auto-regression, where the next token is predicted based on the compressed activations from the memory tokens and the raw activations from the input ordinary tokens.

## 3. Experimental Results

### 3.1. Experimental Setup

In addition to the 6K synthetic data we generate, our training dataset is also comprised of 150K pretraining samples from RedPajama and 10K human-annotated instruction-tuning samples from LongAlpaca to mitigate forgetting. We train for one epoch of the whole dataset on a 8xA100 GPU node with training parameters following (Zhang et al., 2024).

We apply our methodology to two different LLM backbones, Llama-2-7B-chat and Llama-2-13B-chat, which have native context lengths of 4K. Our approach is evaluated on the

| | Llama-2-7B-chat | | | Llama-2-13B-chat | | |
|---|---|---|---|---|---|---|
| Dataset | MHA | Experts | Experts+Synth. | MHA | Experts | Experts+Synth. |
| NarrativeQA | 22.85 | 21.03 | 17.67 | 21.31 | 23.63 | 22.83 |
| Qasper | 17.86 | 23.77 | 28.77 | 23.52 | 27.86 | 29.83 |
| MultiFieldQA | 30.97 | 33.49 | 33.82 | 36.16 | 40.64 | 41.36 |
| **Single-Doc QA Avg** | 23.89 | 26.1 | **26.75** | 27.0 | 30.71 | **31.34** |
| HotpotQA | 35.9 | 36.22 | 37.85 | 43.13 | 43.7 | 48.57 |
| 2WikiMultihopQA | 30.97 | 30.02 | 33.56 | 37.25 | 36.89 | 41.28 |
| MuSiQue | 15.89 | 18.69 | 19.38 | 27.01 | 27.82 | 31.57 |
| **Multi-Doc QA Avg** | 27.59 | 28.31 | **30.26** | 35.8 | 36.14 | **40.47** |
| GovReport | 24.89 | 26.84 | 27.73 | 27.46 | 30.56 | 29.66 |
| QMSum | 21.68 | 23.21 | 23.02 | 23.06 | 23.68 | 23.13 |
| MultiNews | 25.27 | 25.3 | 25.27 | 25.63 | 25.91 | 26.03 |
| **Summarization Avg** | 23.95 | 25.12 | **25.34** | 25.38 | **26.72** | 26.27 |
| TREC | 52.5 | 55.5 | 59.0 | 66.5 | 66.5 | 67.5 |
| TriviaQA | 82.04 | 76.96 | 82.05 | 83.51 | 82.94 | 81.88 |
| SAMSum | 42.11 | 43.52 | 43.49 | 42.64 | 42.04 | 40.12 |
| **Few-shot Avg** | 58.88 | 58.66 | **61.51** | **64.22** | 63.83 | 63.17 |
| LCC | 60.12 | 60.25 | 60.78 | 58.79 | 60.99 | 61.04 |
| RepoBench-P | 51.26 | 52.44 | 51.72 | 57.55 | 56.66 | 56.78 |
| **Code Completion Avg** | 55.69 | **56.34** | 56.25 | 58.17 | 58.83 | **58.91** |
| **All Dataset Avg** | 36.74 | 37.66 | **38.87** | 40.97 | 42.13 | **42.97** |

*Table 1.* Accuracy on LongBench for LoRA compression experts and training with synthetic data. The two modifications lead to additive performance improvements over the MHA baseline for both Llama-2-7B-chat and Llama-2-13B-chat.

five real-world long-context tasks in the LongBench dataset: single-document QA, multi-document QA, summarization, few-shot learning, and code completion, totaling 3,350 test samples. Following previous work, we truncate all samples to 16K tokens for evaluation.

## 3.2. Main Results

**Multiple LoRA compression experts yields stronger performance with fewer parameters.** In Table 1, we present the performance comparison between cloning and finetuning a single set of MHA parameters for all compression ratios vs. training a battery of LoRA compression experts which each specialize in a single compression ratio. The results show that using the compression experts significantly improves performance across all LongBench tasks except for few-shot and across both LLM backbones.

Furthermore, using specialized LoRA experts comes with the additional benefit of a substantially reduced memory footprint. The previous MHA cloning method increased the number of parameters of the base LLM by about 1/3 ($\sim$2B additional parameters for Llama-2-7B), whereas our battery of LoRA experts only requires a few hundred million additional parameters. The smaller memory requirement leads to practical deployment benefits.

**Training with synthetic long-context data provides an orthogonal accuracy improvement.** We present the results of training on 6K additional synthetic data in Table 1, where we observe further performance gains on LongBench for both backbones. The accuracy increase is reflected across nearly all tasks but is particularly large for multi-doc QA, which suggests that the synthetic tasks we designed helped the model learn to reason across multiple passages. These results indicate that careful design of synthetic long-context data can lead to performance benefits on related real-world tasks and reveal the potential for further gains by introducing additional, more diverse synthetic tasks. Overall, our findings demonstrate that an LLM can indeed leverage its own synthetically generated training data to learn to extend its context.

## 4. Conclusion

We proposed a novel set of synthetic tasks for creating long context instruction tuning data, along with a context extension method that relies on a set of compression experts, i.e., multiple LoRA adapters trained at particular compression rates. Our long context instruction tuning data will be released to the community. As future work, we plan to explore the use of our data for model fine-tuning and extension to multiple modalities.

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

## References

Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., et al. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023.

Chevalier, A., Wettig, A., Ajith, A., and Chen, D. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*, 2023.

Fellbaum, C. Wordnet. In *Theory and applications of ontology: computer applications*, pp. 231–243. Springer, 2010.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

He, Z., Karlinsky, L., Kim, D., McAuley, J., Krotov, D., and Feris, R. Camelot: Towards large language models with training-free consolidated associative memory. *arXiv preprint arXiv:2402.13449*, 2024.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Liu, H., Zaharia, M., and Abbeel, P. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023.

Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., and Liang, P. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Mitra, A., Del Corro, L., Mahajan, S., Codas, A., Simoes, C., Agarwal, S., Chen, X., Razdaibiedina, A., Jones, E., Aggarwal, K., et al. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*, 2023.

Mu, J., Li, X., and Goodman, N. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36, 2024.

Munkhdalai, T., Faruqui, M., and Gopal, S. Leave no context behind: Efficient infinite context transformers with infini-attention. *arXiv preprint arXiv:2404.07143*, 2024.

Peng, B., Quesnelle, J., Fan, H., and Shippole, E. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.

Sudalairaj, S., Bhandwaldar, A., Pareja, A., Xu, K., Cox, D. D., and Srivastava, A. Lab: Large-scale alignment for chatbots. *arXiv preprint arXiv:2403.01081*, 2024.

Tay, Y., Dehghani, M., Bahri, D., and Metzler, D. Efficient transformers: A survey. *ACM Computing Surveys*, 55(6): 1–28, 2022.

Zhang, P., Liu, Z., Xiao, S., Shao, N., Ye, Q., and Dou, Z. Soaring from 4k to 400k: Extending llm's context with activation beacon. *arXiv preprint arXiv:2401.03462*, 2024.