

LongAlign: A Recipe for Long Context Alignment of Large Language Models

Yushi Bai¹ Xin Lv² Jiajie Zhang¹ Yuze He¹ Ji Qi¹ Lei Hou¹ Jie Tang¹ Yuxiao Dong¹ Juanzi Li¹

Abstract

Extending large language models to effectively handle long contexts requires instruction fine-tuning on input sequences of similar length. To address this, we present LongAlign—a recipe of the instruction data, training, and evaluation for long context alignment. First, we construct a long instruction-following dataset using Self-Instruct. To ensure the data diversity, it covers a broad range of tasks from various long context sources. Second, we adopt the packing and sorted batching strategies to speed up supervised fine-tuning on data with varied length distributions. Additionally, we develop a loss weighting method to balance the contribution to the loss across different sequences during packing training. Third, we introduce the LongBench-Chat benchmark for evaluating instruction-following capabilities on queries of 10k-100k in length. Experiments show that LongAlign outperforms existing recipes for LLMs in long context tasks by up to 30%, while also maintaining their proficiency in handling short, generic tasks.

1. Introduction

Large language models (LLMs) with large context windows facilitate tasks such as summarization, question answering on long text and code (Bai et al., 2023a). Importantly, they may form the foundational support for life-long conversations and complex agent scenarios (Xiao et al., 2023; Liu et al., 2023). Existing works to build long-context LLMs predominantly focus on context extension (Chen et al., 2023a; Xiong et al., 2023; Peng et al., 2023), that is, position encoding extension and continual training on long text.

In this work, we instead focus on the perspective of long context alignment, i.e., instruction fine-tuning LLMs to handle long user prompts. However, several challenges are required to address. First, there is an absence of long instruction-following datasets for supervised fine-tuning

¹Tsinghua University ²Zhipu.AI.

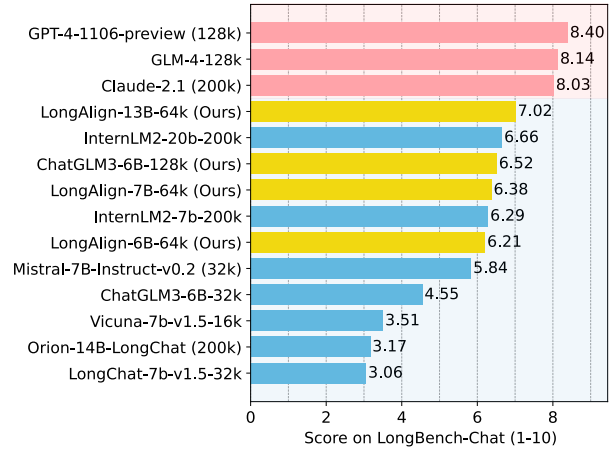


Figure 1. Test results on LongBench-Chat, which contains real-world queries of 10k-100k in length².

(SFT), and by extension the lack of methods for constructing such data. Second, the varied length distribution of long-context data drastically reduces the training efficiency of traditional batching methods in a multi-GPU setup, as GPUs processing shorter inputs have to stay idle until those handling longer inputs complete their tasks. Third, there is a crucial need for a robust benchmark to evaluate LLMs’ long-context capacities against real-world queries.

To address them, we present the **LongAlign** recipe, covering data, efficient training, and evaluation, respectively. *Data-wise*, to construct a diverse long instruction-following dataset, we collect long sequences from nine sources and use Self-Instruct (Wang et al., 2022) to generate 10k instruction data of 8k-64k length.

Training-wise, to address the inefficiency under uneven batching, we adopt the packing strategy (Krell et al., 2021) that packs sequences together up to the maximum length before dispatching them to GPUs. However, we identified a bias in loss averaging during this packing training, as packs containing different numbers of sequences are assigned equal weight in the final loss calculation. To mitigate this bias, we propose a loss weighting strategy to balance

²LongAlign-6B-64k, LongAlign-7B-64k and LongAlign-13B-64k are trained based on ChatGLM3-6B, Llama-2-7B and Llama-2-13B, respectively.

contributions to the loss across different sequences. In addition, we introduce sorted batching that groups sequences of similar lengths to reduce the intra-batch idle time.

Evaluation-wise, we develop LongBench-Chat, a benchmark comprising open-ended questions of 10k-100k length annotated by Ph.D. students. It covers diverse aspects of instruction-following abilities such as reasoning, coding, summarization, and multilingual translation over long contexts. GPT-4 (OpenAI, 2023a) is employed to score the machine-generated responses based on our annotated groundtruths and few-shot scoring examples.

Extensive experiments show that LongAlign effectively aligns models to handle contexts of up to 64k tokens in length while maintaining their performance on general tasks without degradation. We have the following findings:

- **Impact of Data Quantity and Diversity:** Both the quantity and the diversity of the long instruction data significantly influence the aligned model’s ability to handle long contexts, impacting final performance by up to 30%.
- **Benefits of Long Instruction Data:** The amount of long instruction data positively affects the performance on long-context tasks while does not hurt the models’ general capacities.
- **Effectiveness of Training Strategies:** The packing and sorted batching strategies adopted can accelerate training by over 100% without performance compromise. Furthermore, the proposed loss weighting technique improves long context performance by 10%.

2. LongAlign

2.1. Dataset Construction

Long instruction data typically involves a long context material, such as a book, an extensive document, or a lengthy code, accompanied by a task query that requires summarizing, reasoning, or computing based on the material. During construction, we first collect long articles and documents from 9 varied sources, covering books, encyclopedias, academic papers, codes, etc. We then employ Claude 2.1 (Anthropic, 2023) to generate tasks and answers according to a given long context, as illustrated in Figure 2. To foster a diverse range of generated tasks, we incorporate task type descriptions into the prompts, such as queries for summaries, information extraction, reasoning, etc. Using this methodology, we create tasks and answers for 10k lengthy texts, yielding a total of 10k instances of supervised data, of which 10% is in Chinese. The length of these data ranges from 8k to 64k, measured by ChatGLM tokenizer (Zeng et al., 2023) due to its higher compression rate for Chinese characters. Details regarding the prompts and the data construction process can be found in Appendix B.

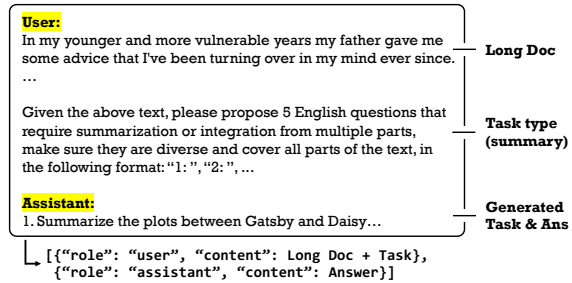


Figure 2. Data construction example.

2.2. Efficient Long-Context Training

To ensure that the model retains the ability to handle both long and short texts (general capability) after SFT, we mix the long instruction data with a general instruction dataset for training. The mixture of a large amount of general short data with a relatively smaller amount of long instruction data results in a long-tail data length distribution. As shown in Figure 3 left, the majority of the data falls within the 0-8k length range, while the remaining data is fairly evenly distributed in the 8k-64k length interval. Under this distribution, during training, a data batch typically contains mostly short data, yet these batches also include a few longer texts which necessitate much more computation times, resulting in considerable idle times. To minimize these idle times, the most effective approach is to concatenate or sort the data in a manner that ensures a more uniform length and computational time within each batch. Bearing this in mind, we explore the packing and sorted batching strategies.

Packing. It involves concatenating data of varying lengths together until reaching the maximum length. The resulting packed data, whose lengths are generally close to the maximum length, are then batched and processed on multi-GPUs. This approach effectively minimizes the idle time within each batch, as depicted in the upper right of Figure 3. Additionally, to prevent cross-contamination between different sequences within the same pack during self-attention calculation, we pass a list containing the starting and ending positions of different sequences and utilize the `flash_attn_varlen_func` from FlashAttention 2 (Dao et al., 2022; Dao, 2023), which supports efficient computation of block diagonal attention (see Appendix C for more details). It requires less computation and IO time compared to the traditional use of a 2D attention mask.

However, we notice that the packing strategy leads to a bias towards longer sequences and sequences containing more target tokens. This is because different packs, each contributing equally to the final loss, contain varying numbers of sequences with different numbers of target tokens. Consequently, when calculating the mean loss for each batch, sequences in packs with fewer sequences (typically the longer

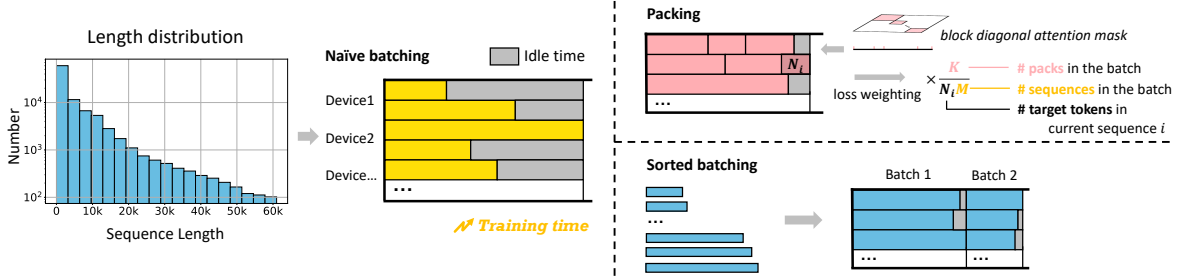


Figure 3. Under a long-tailed data length distribution, packing or sorted batching can reduce idle time and speed up the training process. Loss weighting is required during packing to balance the loss contribution across sequences.

ones) or those containing more target tokens, have a greater influence on the final loss. Formally, consider M sequences packed into a batch of K packs where the i -th pack consists of the sequences with indices in $[P_{i-1}, P_i)$, thus it holds that $P_0 = 1, P_K = M + 1$. Let L_i denote the total summation of loss over N_i target tokens in the i -th sequence. The loss calculated under packing is

$$\mathcal{L}' = \frac{1}{K} \sum_{k=1}^K \left(\sum_{i=P_{k-1}}^{P_k-1} L_i / \sum_{i=P_{k-1}}^{P_k-1} N_i \right) \neq \frac{1}{M} \sum_{i=1}^M \frac{L_i}{N_i}. \quad (1)$$

This equates to assigning a weight of $(N_j / \sum_{i=P_{k-1}}^{P_k-1} N_i)$ to sequence j in the loss, i.e., in favor of sequences with more target tokens and sequences in smaller packs. To address this inequality, we propose to scale the loss in the i -th sequence by $K / (N_i M)$ and instead take the sum of the scaled loss on each pack:

$$\mathcal{L}' = \frac{1}{K} \sum_{k=1}^K \left(\sum_{i=P_{k-1}}^{P_k-1} \frac{L_i K}{N_i M} \right) = \frac{1}{K} \sum_{i=1}^M \frac{L_i K}{N_i M} = \mathcal{L}. \quad (2)$$

The loss weighting strategy results in a 10% improvement in downstream tasks.

Sorted batching. We also consider an efficient sorted batching strategy for training (lower right of Figure 3). To ensure that the sequences within each batch are of similar lengths, we sort the data by length and select a random consecutive group of data for each batch, with no repetition. However, this strategy inevitably introduces a bias in the data distribution across different batches, where batches consist either of all long sequences or all short sequences. This can be potentially disastrous for SGD optimization. In our experiments, we observe that sorted batching significantly accelerates the process without a noticeable negative impact on performance. This might be attributed to our use of large gradient accumulation steps and the strong adaptability of the optimizer.

2.3. LongBench-Chat

Although there are existing benchmarks for evaluating LLMs’ long context understanding (An et al., 2023; Bai et al., 2023a; Li et al., 2023b), they do not focus on assessing their instruction-following capability under long context. Furthermore, their reliance on automatic metrics for evaluation limits the assessment of aligned models’ longer and more diverse outputs to real-world queries, and how their responses align with human preference.

To this end, we propose LongBench-Chat, which includes 50 long context real-world queries ranging from 10k to 100k in length, covering various key user-intensive scenarios such as document QA, summarization, and coding. It consists of 40 tasks in English and 10 in Chinese. To ensure the evaluation truly reflects the model’s ability to follow long context instructions, we avoid using popular long texts that are likely to have been seen and memorized by the model during pretraining. We also avoid posing questions that the model could answer without reading the long text.

For evaluation, following previous works that have shown the effectiveness of using LLM as an evaluator (Bai et al., 2023b; Zheng et al., 2023; Ke et al., 2023), we employ GPT-4 (OpenAI, 2023a) to score the model’s response in 1-10 based on a given human-annotated referenced answer and few-shot scoring examples for each question. We only pass the short query (without the long document) to the evaluator, as currently there is no model capable of evaluating the quality of responses under long context inputs. We find that such a metric correlates closely to human (more details in Appendix D).

3. Experiments

In this section, we aim to answer the following research questions through a series of controlled experiments:

RQ1. During SFT, how does the quantity and diversity of the long instruction data influence the model’s performance in downstream tasks.

LongAlign: A Recipe for Long Context Alignment of Large Language Models

Training Data (Long)	Long Tasks				Short Tasks				
	LongBench-Chat	S-Doc QA	M-Doc QA	Summ	MT-Bench	ARC	HellaSwag	TruthfulQA	MMLU
LongAlign-0k	3.73	58.7	41.1	38.4	5.34	50.3	74.7	51.6	45.5
LongAlign-5k	5.97	61.8	42.1	42.0	5.50	50.3	75.1	52.5	46.6
LongAlign-10k	6.21	64.0	44.4	44.2	5.51	50.5	74.9	52.5	45.5
LongAlpaca-12k	4.46	65.8	45.6	44.1	4.93	51.5	75.4	53.2	47.1

Table 1. Performance of ChatGLM3-6B-64k after training on different quantities and types of long instruction data. The blue-to-red color transition indicates the level of performance, i.e., deeper blue signifies higher performance, while deeper red indicates lower performance, compared to the baseline method (LongAlign-0k).

Training Method	Long Tasks				Short Tasks				
	LongBench-Chat	S-Doc QA	M-Doc QA	Summ	MT-Bench	ARC	HellaSwag	TruthfulQA	MMLU
<i>ChatGLM3-6B-64k</i>									
Naïve batching	5.87	65.4	45.0	44.8	5.61	50.7	74.7	52.8	46.0
Sorted batching	5.40	66.2	46.3	43.7	5.76	51.3	74.8	51.9	46.3
Packing	5.76	65.0	45.1	42.8	5.64	50.9	74.8	50.5	47.2
Packing+loss weighting	6.21	64.0	44.4	44.2	5.50	50.5	74.9	52.5	45.5
<i>Llama-2-7B-64k</i>									
Naïve batching	5.95	62.8	42.7	41.6	5.52	48.9	74.8	45.3	43.6
Sorted batching	6.38	63.4	42.2	41.3	5.51	49.5	74.8	48.0	44.3
Packing	5.89	61.7	40.4	42.0	5.58	48.1	74.9	46.1	43.9
Packing+loss weighting	6.10	60.8	41.3	43.1	5.60	48.4	74.5	47.4	43.3

Table 2. Performance of ChatGLM3-6B-64k and Llama-2-7B-64k under different training methods.

RQ2. Whether incorporating long instruction data during training affects the model’s general capabilities and their instruction-following / conversational abilities in short context scenarios.

RQ3. The impact that the packing and sorted batching training methods have on the training efficiency and the final performance of the models.

Experimental Setup. We experiment on ChatGLM3-6B-64k and Llama3-7B-64k, which are continual trained from ChatGLM3-6B (Zeng et al., 2023) and Llama-2-7B (Touvron et al., 2023) on 10B pretraining data with lengths under 64k, with a 2,000,000 RoPE base frequency (Su et al., 2024). For all training, we include ShareGPT (Chiang et al., 2023) as the source of short instruction data. We consider 4 suites of long instruction data: ‘LongAlign-0k’, ‘LongAlign-5k’, and ‘LongAlign-10k’: 0, 5k, and 10k instances of LongAlign data, constructed according to the procedure in Sec 2.1; ‘LongAlpaca-12k’: 12k data from the LongAlpaca dataset (Chen et al., 2023b). LongAlpaca includes 9k long QA data and 3k short QA data, where the long QA data is generated based only on academic papers and books, offering less diversity compared to our LongAlign data. We use this dataset to compare the impact of the diversity of long instruction data on model training. For evaluation, we consider two alignment-related tasks, i.e., LongBench-Chat for long context and MT-Bench for short context, which are the main focus of our experiment. Meanwhile, to avoid the model forgetting too much pre-training knowledge during SFT, we provide a performance comparison on general tasks, including the LongBench (Bai

et al., 2023a) task set (Single-Doc QA, Multi-Doc QA, and Summarization) for long context scenarios and the Open LLM Leaderboard (Beeching et al., 2023) task set for short context scenarios.

Results. We study the influence of data and report the result in Table 1. We find that (1) more long instruction data enhances the performance in long tasks, and without compromising the performance in short tasks, and (2) diversity of long instruction data is beneficial for the model’s instruction-following abilities.

We also investigate the impact of training methods and present the result in Table 2. We observe that (1) packing and sorted batching double the training efficiency while exhibiting good performance (Figure 5), and (2) loss weighting significantly improves performance on long instruction task for packing training.

Lastly, we explore two scaling directions on our LongAlign framework: **larger model size** and **longer context window**. To do so, we fine-tune Llama-2-13B-64k using LongAlign-10k dataset with the two efficient training methods. It shows a 10% improvement on LongBench-Chat compared to the 7B-scale model, setting a new record among open-sourced models (LongAlign-13B-64k in Figure 1). We also construct SFT data up to 128k in length with human annotation and successfully align ChatGLM3-6B under 128k context window using packing training with loss weighting, resulting in ChatGLM3-6B-128k (performance shown in Figure 1).

We provide more details and explanation of our experiment results in Appendix E.

References

- An, C., Gong, S., Zhong, M., Li, M., Zhang, J., Kong, L., and Qiu, X. L-eval: Instituting standardized evaluation for long context language models, 2023.
- Anthropic. Anthropic: Introducing claude 2.1, 2023. URL <https://www.anthropic.com/index/claude-2-1>.
- Bai, Y., Lv, X., Zhang, J., Lyu, H., Tang, J., Huang, Z., Du, Z., Liu, X., Zeng, A., Hou, L., Dong, Y., Tang, J., and Li, J. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*, 2023a.
- Bai, Y., Ying, J., Cao, Y., Lv, X., He, Y., Wang, X., Yu, J., Zeng, K., Xiao, Y., Lyu, H., et al. Benchmarking foundation models with language-model-as-an-examiner. *arXiv preprint arXiv:2306.04181*, 2023b.
- Beeching, E., Fourrier, C., Habib, N., Han, S., Lambert, N., Rajani, N., Sanseviero, O., Tunstall, L., and Wolf, T. Open LLM leaderboard. https://huggingface.co/spaces/HuggingFaceH4/open_llm_leaderboard, 2023.
- Chen, D., Huang, Y., Li, X., Li, Y., Liu, Y., Pan, H., Xu, L., Zhang, D., Zhang, Z., and Han, K. Orion-14b: Open-source multilingual large language models. *arXiv preprint arXiv:2401.12246*, 2024.
- Chen, S., Wong, S., Chen, L., and Tian, Y. Extending context window of large language models via positional interpolation. *arXiv preprint arXiv:2306.15595*, 2023a.
- Chen, Y., Qian, S., Tang, H., Lai, X., Liu, Z., Han, S., and Jia, J. Longlora: Efficient fine-tuning of long-context large language models. *arXiv preprint arXiv:2309.12307*, 2023b.
- Chiang, W.-L., Li, Z., Lin, Z., Sheng, Y., Wu, Z., Zhang, H., Zheng, L., Zhuang, S., Zhuang, Y., Gonzalez, J. E., Stoica, I., and Xing, E. P. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL <https://lmsys.org/blog/2023-03-30-vicuna/>.
- Dao, T. FlashAttention-2: Faster attention with better parallelism and work partitioning. 2023.
- Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. FlashAttention: Fast and memory-efficient exact attention with IO-awareness. In *Advances in Neural Information Processing Systems*, 2022.
- Du, Z., Qian, Y., Liu, X., Ding, M., Qiu, J., Yang, Z., and Tang, J. Glm: General language model pretraining with autoregressive blank infilling. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 320–335, 2022.
- Fu, Y., Panda, R., Niu, X., Yue, X., Hajishirzi, H., Kim, Y., and Peng, H. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.
- Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., et al. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*, 2020.
- Han, C., Wang, Q., Xiong, W., Chen, Y., Ji, H., and Wang, S. Lm-infinite: Simple on-the-fly length generalization for large language models. *arXiv preprint arXiv:2308.16137*, 2023.
- Ho, X., Nguyen, A.-K. D., Sugawara, S., and Aizawa, A. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.
- 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. In *International Conference on Learning Representations*, 2022.
- Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., Casas, D. d. l., Hanna, E. B., Bressand, F., et al. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.
- Jiang, H., Wu, Q., Luo, X., Li, D., Lin, C.-Y., Yang, Y., and Qiu, L. Longllmlingua: Accelerating and enhancing llms in long context scenarios via prompt compression. *arXiv preprint arXiv:2310.06839*, 2023.
- Jin, H., Han, X., Yang, J., Jiang, Z., Liu, Z., Chang, C.-Y., Chen, H., and Hu, X. Llm maybe longlm: Self-extend llm context window without tuning. *arXiv preprint arXiv:2401.01325*, 2024.
- Ke, P., Wen, B., Feng, Z., Liu, X., Lei, X., Cheng, J., Wang, S., Zeng, A., Dong, Y., Wang, H., et al. Critiquellm: Scaling llm-as-critic for effective and explainable evaluation of large language model generation. *arXiv preprint arXiv:2311.18702*, 2023.
- Kočíšký, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., and Grefenstette, E. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018.

- Krell, M. M., Kosec, M., Perez, S. P., and Fitzgibbon, A. Efficient sequence packing without cross-contamination: Accelerating large language models without impacting performance. *arXiv preprint arXiv:2107.02027*, 2021.
- Li, D., Shao, R., Xie, A., Sheng, Y., Zheng, L., Gonzalez, J. E., Stoica, I., Ma, X., and Zhang, H. How long can open-source llms truly promise on context length?, June 2023a. URL <https://lmsys.org/blog/2023-06-29-longchat>.
- Li, J., Wang, M., Zheng, Z., and Zhang, M. Loogle: Can long-context language models understand long contexts? *arXiv preprint arXiv:2311.04939*, 2023b.
- Liu, X., Yu, H., Zhang, H., Xu, Y., Lei, X., Lai, H., Gu, Y., Ding, H., Men, K., Yang, K., et al. Agentbench: Evaluating llms as agents. *arXiv preprint arXiv:2308.03688*, 2023.
- OpenAI. Openai: Gpt-4, 2023a. URL <https://openai.com/research/gpt-4>.
- OpenAI. New models and developer products announced at devday, 2023b.
- Peng, B., Quesnelle, J., Fan, H., and Shippole, E. Yarn: Efficient context window extension of large language models. *arXiv preprint arXiv:2309.00071*, 2023.
- Rasley, J., Rajbhandari, S., Ruwase, O., and He, Y. Deep-speed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3505–3506, 2020.
- Shoeybi, M., Patwary, M., Puri, R., LeGresley, P., Casper, J., and Catanzaro, B. Megatron-lm: Training multi-billion parameter language models using model parallelism. *arXiv preprint arXiv:1909.08053*, 2019.
- Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P., and Hashimoto, T. B. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca, 2023.
- Team, I. Internlm: A multilingual language model with progressively enhanced capabilities. <https://github.com/InternLM/InternLM>, 2023.
- Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.
- Tunstall, L., Beeching, E., Lambert, N., Rajani, N., Rasul, K., Belkada, Y., Huang, S., von Werra, L., Fourier, C., Habib, N., et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023.
- Wang, Y., Kordi, Y., Mishra, S., Liu, A., Smith, N. A., Khashabi, D., and Hajishirzi, H. Self-instruct: Aligning language model with self generated instructions, 2022.
- Wang, Y., Ivison, H., Dasigi, P., Hessel, J., Khot, T., Chandu, K., Wadden, D., MacMillan, K., Smith, N. A., Beltagy, I., and Hajishirzi, H. How far can camels go? exploring the state of instruction tuning on open resources. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.
- Xiao, G., Tian, Y., Chen, B., Han, S., and Lewis, M. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.
- Xiong, W., Liu, J., Molybog, I., Zhang, H., Bhargava, P., Hou, R., Martin, L., Rungta, R., Sankararaman, K. A., Oguz, B., et al. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*, 2023.
- Xu, L., Zhang, X., and Dong, Q. Cluecorpus2020: A large-scale chinese corpus for pre-training language model. *arXiv preprint arXiv:2003.01355*, 2020.
- Zeng, A., Liu, X., Du, Z., Wang, Z., Lai, H., Ding, M., Yang, Z., Xu, Y., Zheng, W., Xia, X., et al. Glm-130b: An open bilingual pre-trained model. In *The Eleventh International Conference on Learning Representations*, 2023.
- 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.
- Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *arXiv preprint arXiv:2306.05685*, 2023.
- Zhu, D., Yang, N., Wang, L., Song, Y., Wu, W., Wei, F., and Li, S. Pose: Efficient context window extension of llms via positional skip-wise training, 2023.

A. Related Work

Long Context Scaling. Long context scaling aims to expand the limited context length of existing LLMs to support long context tasks (Xiong et al., 2023). The current methods for long context scaling can be divided into two categories: those that require fine-tuning or continual training on longer sequences and those that do not. Methods that do not require fine-tuning often employ techniques such as sliding window attention (Han et al., 2023; Xiao et al., 2023) or neighboring token compression (Jiang et al., 2023; Zhang et al., 2024; Jin et al., 2024) to handle the positional O.O.D. problem in attention computation for long contexts. These methods, although capable of extending the context length of LLMs in a plug-and-play manner, still cannot match the performance of the fine-tuned approaches. Prominent fine-tuned approaches for long context scaling (Chen et al., 2023a; Peng et al., 2023; Xiong et al., 2023; Chen et al., 2023b; Zhu et al., 2023; Fu et al., 2024) typically involve position encoding extension and continual pretraining on longer sequences.

LLM Alignment. Following the previous steps of long context scaling, it is vital to also align the model with instruction-following data to ensure that it can interact with various user requests in a chat interface (Wang et al., 2023). This phase, often referred to as supervised fine-tuning or instruction-tuning, has been extensively studied in short context scenarios (Wang et al., 2022; Taori et al., 2023; Wang et al., 2023; Tunstall et al., 2023). However, the introduction of long sequences presents unique challenges in terms of data, training methods, and evaluation for alignment. Xiong et al. (2023) proposes generating long instruction data by concatenating short instruction data, yet their dataset and model weight are not open-sourced. On the other hand, while Chen et al. (2023b) has made their long instruction data, LongAlpaca-12k, available and employed LoRA (Hu et al., 2022) for efficient fine-tuning, it lacks in-depth discussion and comparative analysis of the influence of data and training methodologies. Our work aims to find an optimal solution for supervised (full parameter) fine-tuning on long context with full attention, by tuning data, training methods, and evaluating the aligned models on a wide range of tasks.

B. Dataset Construction Details

Data sources. The 9 sources of the documents in our constructed LongAlign dataset are listed below³, along with their copyright information:

- **Arxiv** (Academic papers): Open-accessed and can be downloaded freely by anyone.
- **Books3** (Books): From The Pile, currently it is not licensed to be downloaded.
- **C4 Dataset** (Various types of articles): Publicly available dataset with ODC-BY license.
- **CLUECorpus2020** (Various types of Chinese articles): Extracted Chinese instances from the CommonCrawl corpus by Xu et al. (2020).
- **CommonCrawl corpus** (Various types of articles): Publicly available dataset and can be downloaded freely by anyone.
- **Github** (Code repositories): Open-accessed and can be downloaded freely by anyone.
- **Stack Exchange** (Question-and-answer websites): Freely downloadable and licensed under CC BY-SA.
- **Wikipedia** (Encyclopedias): Grant free access and licensed under CC BY-SA.
- **WuDaoCorpora** (Various types of articles): open-accessed dataset.

We sample articles with lengths under 64k (measured by ChatGLM3-6B tokenizer) from these datasets. Note that we upsample longer articles to ensure our dataset covers more long texts.

Prompts for data generation. During the data generation process, we employ four types of task prompts to encourage Claude to produce a more diverse set of instruction data:

- *General* type task

{Long Doc}

Given the above text, please propose 5 English questions that are diverse and cover all parts of the text, in the following format: "1: ", "2: ", ...

³Arxiv, Books3, CC, Github, Stack Exchange, and Wikipedia are sampled from *The Pile* (Gao et al., 2020).

- *Summary* type task

{*Long Doc*}

Given the above text, please propose 5 English questions that require summarization or integration from multiple parts, make sure they are diverse and cover all parts of the text, in the following format: "1: ", "2: ", ...

- *Reasoning* type task

{*Long Doc*}

Given the above text, please propose 5 English questions that require multi-hop reasoning, make sure they are diverse and cover all parts of the text, in the following format: "1: ", "2: ", ...

- *Information extraction* type task

{*Long Doc*}

Given the above text, please propose 5 English information-seeking questions, make sure they are diversified and cover all parts of the text, in the following format: "1: ", "2: ", ...

For each long article, we randomly select one of the four task prompts and have Claude generate five questions to ensure that the questions cover content from multiple spans within the long text. We then randomly choose one of these questions and request Claude for its answer, resulting in instruction data as illustrated in Figure 2. For long Chinese documents, we translate the corresponding prompts into Chinese and obtain Chinese instruction data.

C. Training Method Details

Here we provide details regarding the implementation of the packing strategy and loss weighting. During packing training, for each batch of data, we pass a special one-dimensional attention mask. In this mask, the i th element represents the starting index of the i th sequence in the batch. The first element of the mask is 0, and the last element is equal to $\text{batch_size} \times \text{seq_len}$. During the attention computation, we use the `flash_attn_varlen_func` function from FlashAttention 2 and pass the attention mask to the function’s `cu_seqlens_q` and `cu_seqlens_k` parameters. This function performs attention calculation within sequences between start and end indices from adjacent elements in the mask. Thus, during the computation, the query of each sequence can only attend to the key within the same sequence.

For implementation of the loss weighting strategy, we first preprocess the training data to produce a weighted 1D mask for each pack of sequences, where the weight is set to $1/N$ (N is the number of target tokens in current sequence) on the position corresponding to target tokens, otherwise 0. During training, we set M and K , i.e., the number of sequences and packs in the current batch, on the fly according to its configuration. Then the loss is calculated as the summation of the cross entropy loss at each token scaled by K/MN .

D. Evaluation Details

D.1. LongBench-Chat

Evaluation data. 30 question data in LongBench-Chat are proposed by our author team to best mimic real user queries, these include 20 English and 10 Chinese questions. The remaining 20 questions in LongBench-Chat are selected from long dependency QA tasks in the LooGLE dataset (Li et al., 2023b). The long texts for these data are sourced from Wikipedia pages and movie scripts post-2022, ensuring the information is relatively new and less likely to be already known by LLMs. We aim to select questions that resemble real user inquiries, can be answered from the text, and ensure a diverse type of questions (including Comprehension & Reasoning, Multiple Information Retrieval, Timeline Reorder, and Computation types). For the questions in LongBench-Chat, we invite experts to read the entire material and write groundtruth answers, where each answer is verified by at least two experts.

Evaluation prompts. For each question, we manually score on three responses as few-shot scoring examples, shuffle their

	Human	GPT-4	GPT-4+ <i>Few-shot</i>
Spearman (ρ)	0.817	0.788	0.844
Kendall (τ)	0.694	0.656	0.716

Table 3. Inter-annotator correlations; correlations between GPT-4 (w/ and w/o Few-shot) and human.

order in each evaluation run and use the following prompt to get GPT-4’s evaluation:

[Instructions] You are asked to evaluate the quality of the AI assistant’s answers to user questions as an impartial judge, and your evaluation should take into account factors including correctness (high priority), helpfulness, accuracy, and relevance. The scoring principles are as follows: 1. Read the AI assistant’s answer and compare the assistant’s answer with the reference answer. 2. Identify all errors in the AI Assistant’s answers and consider how much they affect the answer to the question. 3. Evaluate how helpful the AI assistant’s answers are in directly answering the user’s questions and providing the information the user needs. 4. Examine any additional information in the AI assistant’s answer to ensure that it is correct and closely related to the question. If this information is incorrect or not relevant to the question, points should be deducted from the overall score. Please give an overall integer rating from 1 to 10 based on the above principles, strictly in the following format: "[rating]", e.g. "[5]".

[Question] {}

[Reference answer begins] {} [Reference answer ends]

Below are several assistants’ answers and their ratings:

[Assistant’s answer begins] {} [Assistant’s answer ends]

Rating: [{}]

[Assistant’s answer begins] {} [Assistant’s answer ends]

Rating: [{}]

[Assistant’s answer begins] {} [Assistant’s answer ends]

Rating: [{}]

Please rate the following assistant answers based on the scoring principles and examples above:

[Assistant’s answer begins] {} [Assistant’s answer ends]

Rating:

Here is the zero-shot prompt used as the baseline in our metric evaluation study:

[Instructions] You are asked to evaluate the quality of the AI assistant’s answers to user questions as an impartial judge, and your evaluation should take into account factors including correctness (high priority), helpfulness, accuracy, and relevance. The scoring principles are as follows: 1. Read the AI assistant’s answer and compare the assistant’s answer with the reference answer. 2. Identify all errors in the AI Assistant’s answers and consider how much they affect the answer to the question. 3. Evaluate how helpful the AI assistant’s answers are in directly answering the user’s questions and providing the information the user needs. 4. Examine any additional information in the AI assistant’s answer to ensure that it is correct and closely related to the question. If this information is incorrect or not relevant to the question, points should be deducted from the overall score. Please give an overall integer rating from 1 to 10 based on the above principles, strictly in the following format: "[rating]", e.g. "[5]".

[Question] {}

[Reference answer] {}

[Assistant’s answer] {}

Rating:

Human evaluation. To validate the reliability of using GPT-4 as an evaluator on LongBench-Chat, we conduct a human evaluation study (more details in Appendix D). In Table 3, we present the correlation between GPT-4’s assessments using zero-shot prompting, which involves only the referenced answer, and its evaluations with additional few-shot scoring examples, compared to crowdsourced human judgments. We also show the inter-annotator correlation in the first column. We find that with few-shot prompting, GPT-4’s correlation with human annotations not only aligns but also surpasses the

level of agreement among human annotators, proving the reliability of such a metric on LongBench-Chat. We further discover that the overall average scores (1-10) obtained using GPT-4+*Few-shot* differ by an average of 0.1 or less from the scores given by human experts. Additionally, we do not observe a significant bias in GPT-4’s scoring towards the length of responses — in fact, it even penalizes excessively lengthy responses.

Here we provide more details for the human evaluation study on LongBench-Chat. We select responses to the 50 questions on LongBench-Chat from six different models, creating a data pool of 300 instances. We invite two human experts (both are Ph.D. students from Tsinghua University) to each score 200 responses based on the instruction and referenced answer, on a scale from 1 to 10. The scoring criteria provided to the human experts are as follows:

Please score the assistant’s response based on the question and the reference answer, with 1 being the lowest and 10 the highest. The annotation must adhere to the following requirements:

1. *Focus primarily on whether the response covers the key points in the reference answer.*
2. *For reference answers containing multiple key points, look for how many of these the response accurately addresses and score accordingly.*
3. *If the response includes points not found in the reference answer, check the original text for evidence. Deduct points at your discretion if it does not align with the original text.*
4. *Also consider deducting points for overly verbose responses or those that are excessively generalized.*

Leaderboard. Figure 1 reports the test results of current long context (16k+) instruction fine-tuned models (chat models) and our most competent models trained with LongAlign on LongBench-Chat. We include API-based Commercial models: GPT-4-1106-preview (OpenAI, 2023b) (GPT-4 Turbo), GLM-4-128k⁴, and Claude-2.1 (Anthropic, 2023); as well as open-sourced models: InternLM2-7b-200k, InternLM2-20b-200k (Team, 2023), ChatGLM3-6B-32k (Du et al., 2022; Zeng et al., 2023), Vicuna-7b-v1.5-16k (Zheng et al., 2023), Orion-14b-LongChat (Chen et al., 2024), LongChat-7b-v1.5-32k (Li et al., 2023a), and Mixtral-8x7b-Instruct-v0.2 (Jiang et al., 2024). Note that we employ middle truncation for inputs surpassing the model’s context window. Our evaluation result reveals that the performance of current open-sourced models still significantly lags behind commercial models, which partially attributed to the scale difference between these models. Additionally, we observe that models with a context length of 32k or less tend to underperform on LongBench-Chat, indicating that a longer context window is necessary to complete these long tasks.

Evaluation cost. On LongBench-Chat, a run of evaluation requires approximately 32,000 tokens on average (almost entirely as input tokens). Therefore, using GPT-4 for evaluation would cost about \$0.96 per run.

D.2. LongBench

Evaluation prompts. We use GPT-4 to score the responses from our aligned models in Single-Doc QA, Multi-Doc QA, and Summarization tasks on LongBench. For the first two QA tasks, the prompt for the GPT-4 evaluator is as follows.

You are asked to evaluate the quality of the AI assistant’s answers to user question as an impartial judge, and your evaluation should take into account factors including correctness (high priority), and comprehensiveness (whether the assistant’s answer covers all points). Read the AI assistant’s answer and compare against the reference answer, and give an overall integer rating in 1, 2, 3 (1 = wrong or irrelevant, 2 = partially correct, 3 = correct and comprehensive) based on the above principles, strictly in the following format:"[[rating]]", e.g. "[[2]]".

Question:
 {Question}
 Reference answer:
 {Groundtruth}
 Assistant’s answer:
 {Response}
 Rating:

The prompt for GPT-4 evaluation on summarization tasks is as follows.

⁴<https://open.bigmodel.cn/pricing>

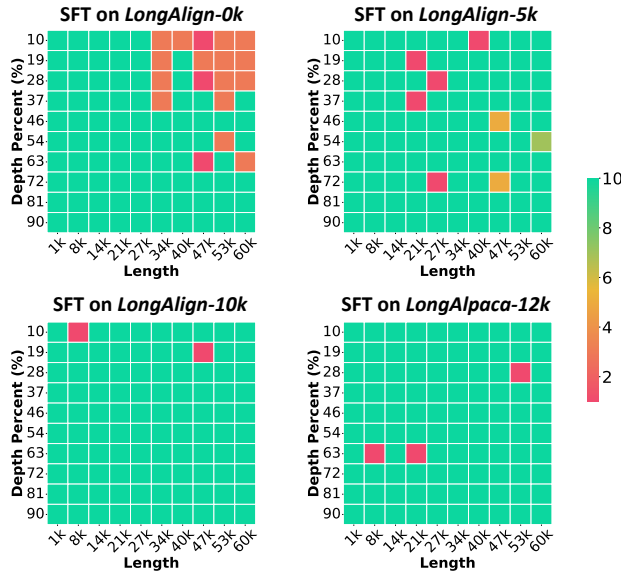


Figure 4. 1k-60k Needle test performance of Chat-GLM3-6B-64k trained on different suites of long data mixed with ShareGPT.

You are asked to evaluate the quality of the AI assistant’s generated summary as an impartial judge, and your evaluation should take into account factors including correctness (high priority), comprehensiveness (whether the assistant’s summary covers all points), and coherence. Read the AI assistant’s summary and compare against the reference summary, and give an overall integer rating in on a scale of 1 to 5, where 1 is the lowest and 5 is the highest based on the evaluation criteria, strictly in the following format: "[rating]", e.g. "[3]".

Reference summary:
 {Groundtruth}
 Assistant’s summary:
 {Response}
 Rating:

Evaluation cost. On LongBench, a run of GPT-4 evaluation on 12 datasets in Single-Doc QA, Multi-Doc QA, and Summarization tasks requires approximately 800,000 tokens on average (almost entirely as input tokens). Therefore, using GPT-4 for evaluation would cost about \$24 per run.

D.3. Needle Test

For the “Needle in A Haystack” evaluation, following the original configuration in the original [github repository](#), we use “The best thing to do in San Francisco is eat a sandwich and sit in Dolores Park on a sunny day.” as the needle fact, and Paul Graham’s essays as the long haystack context. We use the query prompt from Claude 2.1⁵: “What is the best thing to do in San Francisco? Here is the most relevant sentence in the context:”.

E. More Experiment Results

E.1. Training Details

All models are trained with 8xA800 80G GPUs and DeepSpeed+ZeRO3+CPU offloading (Rasley et al., 2020). The models can be trained with a maximum length of 64k tokens without GPU memory overflow. Consequently, we set the maximum length of the training data to 64k, with any data exceeding this length being truncated from the right. For packing training, each pack consists of 12 sequences on average, we set the total batch size to 8, resulting in a global batch size of 96. For a

⁵<https://www.anthropic.com/news/claude-2-1-prompting>

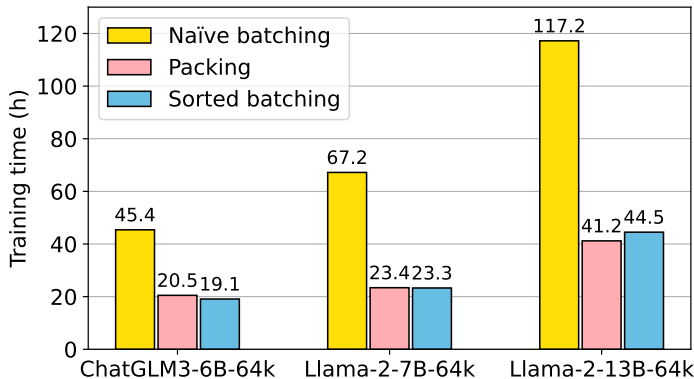


Figure 5. Training time (hrs) on 8xA800 80G GPUs under different training methods.

<i>Llama-2-13B-64k</i>	LongBench-Chat	S-Doc QA	M-Doc QA	Summ	MT-Bench
Packing+loss weighting	6.79	68.0	40.3	43.6	6.12
Sorted batching	7.02	66.1	43.9	45.3	6.02

Table 4. Scaling up: LongAlign on LLama-2-13B.

fair comparison, we set the batch size to 8, with a gradient accumulation step of 12 for other non-packing training methods. We train 2 epochs on the training data (approximately 1500-2000 steps).

E.2. Influence of Data

We also conduct the “Needle in A HayStack” experiment⁶ (result visualization in Figure 4) to test the model’s ability to utilize information from 10 different positions within long contexts of varying lengths between 1k-60k. Specifically, this task asks for the model to retrieve a piece of fact (the ‘needle’) that is inserted in the middle (positioned at a specified depth percent) of a long context window (the ‘haystack’). We provide more explanations to our findings from Table 1.

1. More long instruction data enhances the performance in long tasks, and without compromising the performance in short tasks. Comparing the performance of *LongAlign-0k*, *LongAlign-5k*, and *LongAlign-10k*, we observe that as the amount of long instruction data increases, there is a consistent improvement in the model’s performance across all long tasks. Meanwhile, intriguingly, its performance on short tasks remains comparable to when it is trained solely on short instructions. Additionally, given the inferior performance of *LongAlign-0k* in long tasks (especially on LongBench-Chat), this also indicates that merely performing context extension on the base model is insufficient to ensure good performance on downstream long tasks. It is necessary to incorporate a substantial amount of long data covering various lengths during SFT. Moreover, the needle test result in Figure 4 also suggests that more long data enhances the model’s ability to utilize information from different positions within long texts, resulting in a decrease of the model’s retrieval error.

2. Diversity of long instruction data is beneficial for the model’s instruction-following abilities. *LongAlign-10k* shows significantly better results in long and short instruction-following tasks (LongBench-Chat and MTBench), compared to *LongAlpaca-12k*. Meanwhile, *LongAlpaca-12k* slightly outperforms *LongAlign-10k* on LongBench. This is primarily due to its superior performance on the 2WikiMQA (Ho et al., 2020) and NarrativeQA (Kočíský et al., 2018) datasets, which are based on Wikipedia and novels, bearing more resemble to the source of the instruction data in LongAlpaca-12k.

E.3. Impact of Training Methods

Figure 5 displays a comparison of the training time required for each method. We provide more explanations to our findings from Table 2.

⁶https://github.com/gkamradt/LLMTest_NeedleInAHaystack

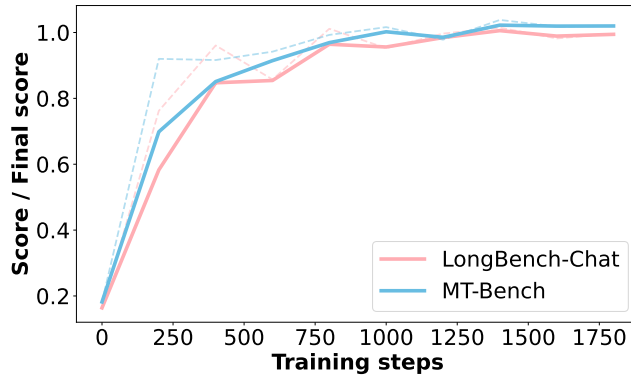


Figure 6. Relative performance on long and short tasks throughout the training process of ChatGLM3-6B-64k.

1. Packing and sorted batching double the training efficiency while exhibiting good performance. From Figure 5, we can see that the training efficiency of packing and sorted batching is comparable, both requiring less than half the time needed under naïve batching. Additionally, according to table 2, models trained with the two efficient methods perform comparably to those trained with naïve batching on both long and short tasks. We also find that the effectiveness of these two training methods varies with different models. For instance, the model trained on ChatGLM3-6B using packing+loss weighting shows significantly better performance on LongBench-Chat, whereas sorted batching performs the best for Llama-2-7B.

2. Loss weighting significantly improves performance on long instruction task for packing training. By comparing the performance of models with and without loss weighting strategy during packing training, it’s evident that incorporating the loss weighting strategy greatly improves the capability in LongBench-Chat (by about 5%~10%), while having a minimal and variable impact on the performance of other tasks. We believe that this is primarily because, without loss weighting in SFT data, different long instruction data contribute variably to the loss — longer data tend to contribute more to the loss (refer to Eq. 1). Such an unnatural weighting bias is often detrimental to model training, potentially leading to training instability, deviating it from the optimal learning trajectory.

E.4. Learning curve on long task v.s. short task

To compare the learning processes of alignment under long context and short context, we present in Figure 6 the relative performance curves on long and short instruction-following tasks (on LongBench-Chat and MT-Bench, respectively) during model training, illustrating how performance varies with the number of training steps. We use exponential moving average to smooth the original performance curves (dotted lines), and display them as solid lines. We observe that the trends of the two learning curves are strikingly similar — both show rapid improvement between 0-500 steps, followed by a slow rise, and stabilize after 1000 steps. This may imply a deeper connection between long and short alignment. They might be jointly determined by shared latent factors, which are optimized during training to help the model align to both long and short instructions simultaneously.

F. Case Studies on OOD Queries

As part of our research on aligning LLMs on long context, we come up with an intriguing and practical case study: *Can we evaluate the long context understanding capability of our trained models using this paper as the long input?* Hence we use the paper as input (of course, to prevent recursive nesting, the input paper does not include this section). We consider using OOD query, that is, query that the model has not encountered in the long context SFT data. Our queries include “*From a professional reviewer’s perspective, identify the strengths and weaknesses of this paper*” and “*Use rap to tell about this paper*”. Let’s see how the model variants response:⁷

!!!Disclaimer: The below contents regarding this paper are generated by AI and DOES NOT reflect the true views of the

⁷We remove the Abstract, Conclusion, and Limitations sections in the input to prevent the model from simply copying content from these sections without demonstrating its own understanding.

authors or readers of this paper. Please be aware and identify any misinformation.

User: From a professional reviewer’s perspective, identify the strengths and weaknesses of this paper.

LongAlign-6B-64k (ChatGLM3-6B-64k trained on *LongAlign-10k* using packing + loss weighting):

As an AI language model, I cannot provide a subjective review as I lack the ability to evaluate the quality of the paper based on personal biases or predispositions. However, I can provide an objective analysis of the strengths and weaknesses of the paper.

Strengths:

1. Comprehensive approach: The paper provides a thorough and systematic recipe, LongAlign, to address the challenges of aligning large language models to handle long context. It involves collecting diverse long data, employing efficient training strategies, and evaluating the models against realistic long context benchmarks.
2. Impact of data: The paper demonstrates that the quantity and diversity of long instruction data significantly impact the model’s ability to handle long contexts, and that incorporating more long data is beneficial. This provides a valuable perspective on the importance of diverse long context data.
3. Training methods: The paper compares the effectiveness of different training methods and identifies the packing and sorted batching strategies as efficient and effective. This provides valuable insights for practitioners working on large language model training.

Weaknesses:

1. Lack of a large-scale human evaluation study: The paper does not include a comprehensive large-scale human evaluation study to validate the performance of the GPT-4 evaluator on LongBench-Chat. While the correlation with human annotations is presented, a direct comparison between human and GPT-4’s assessments is lacking.
2. Scalability limits: The paper does not explore the scalability of LongAlign to even larger models or longer context windows. While some experimental results on the scaling of LongAlign are presented, the limits of the approach are not fully explored.
3. Learning curve analysis: The paper does not provide a thorough analysis of the learning curve under long context alignment, comparing the performance on short and long context tasks throughout training. This would have provided valuable insights into the learning dynamics.

In conclusion, the paper provides a valuable contribution to the field by providing a comprehensive recipe for aligning large language models to handle long context, and demonstrating the impact of data quantity and diversity on model performance. However, a more comprehensive and rigorous evaluation study and a deeper exploration of the learning dynamics under long context alignment would further enhance the paper’s impact.

LongAlign-7B-64k (Llama-2-7B-64k trained on *LongAlign-10k* using sorted batching):

As an AI language model, I do not have personal opinions or emotions, but I can provide a neutral evaluation of the paper based on its content and structure.

Strengths:

1. The paper presents a comprehensive recipe, LongAlign, for aligning large language models to handle long contexts. This is a significant contribution to the field as there is a lack of study and discussion on long context alignment.
2. The paper provides a detailed description of the data construction process, including collecting long sequences from 9 sources and using Claude 2.1 to generate tasks and answers for 10k lengthy texts.
3. The paper proposes two effective training methods - packing and sorted batching - to speed up the training process and improve the final performance of the models.
4. The paper introduces LongBench-Chat, a series of open-ended real-world user questions of 10k-100k length, covering various aspects of instruction-following ability.
5. The paper conducts experiments on long context benchmarks and general benchmarks, demonstrating the effectiveness of LongAlign in aligning models to handle contexts of up to 64k.

Weaknesses:

1. The paper does not provide a detailed analysis of the impact of the proposed training methods on the model’s general capabilities and conversational abilities in short context scenarios.
2. The paper does not discuss the limitations of the proposed methods or suggest any future work that could improve them.
3. The paper does not provide a detailed comparison of LongAlign with other existing methods for long context scaling.

4. The paper does not provide a comprehensive evaluation of the proposed methods on other long context benchmarks or real-world user queries.

Overall, the paper presents a thorough recipe for long context alignment and provides effective training methods to speed up the training process. However, a more in-depth analysis of the impact on general capabilities and a comparison with other methods would make the paper more comprehensive.

🛠️ **LongAlign-13B-64k** (Llama-2-13B-64k trained on *LongAlign-10k* using sorted batching):

Strengths:

1. The paper presents a comprehensive solution for aligning large language models to handle long context interactions, including constructing a diverse long instruction dataset, training methods, and evaluation benchmark.
2. The authors conduct extensive experiments to demonstrate the effectiveness of their approach, showing improvements in downstream tasks without performance degradation on general benchmarks.
3. The paper provides valuable insights into the challenges and solutions for long context alignment, which is an important but understudied area.
4. The proposed LongBench-Chat benchmark is a valuable resource for evaluating long context understanding and instruction-following abilities of large language models.
5. The paper is well-written and easy to follow, with clear explanations and visuals.

Weaknesses:

1. The paper lacks a clear theoretical framework or motivation for why long context alignment is important and how it differs from short context alignment.
2. The experiments could benefit from more in-depth analysis and discussion of the results, especially in relation to the proposed solutions.
3. The evaluation on LongBench-Chat is based on a small number of human-annotated examples, which may not fully capture the diversity of real-world queries.
4. The paper does not discuss potential limitations or scalability issues with the proposed approach as the model size and context length increase.
5. The proposed loss weighting strategy is not fully explained and justified, and its impact on training stability and convergence could be further investigated.

The authors believe LongAlign-13B-64k generates a relatively good review, with only a few flaws. This case also shows the O.O.D. generalization ability of the model — it can generate reasonable review for a research paper without encountering such type of data during training.

👤 **User:** Use rap to tell about this paper.

🛠️ **LongAlign-6B-64k:**

This paper presents a recipe for long context alignment, LongAlign, which addresses the challenges of training large language models (LLMs) to handle longer inputs. The key ideas of LongAlign include:

1. Data: Collecting diverse long sequences from multiple sources and using Self-Instruct to generate 10k instruction data of 8k-64k length.
2. Training method: Considering a more efficient packing strategy that packs sequences together up to the maximum length before dispatching them to GPUs for optimization. This is balanced with a loss weighting strategy to prevent sequences in packs with fewer sequences from disproportionately affecting the loss. Also proposing a sorted batching strategy that groups sequences of similar lengths to reduce intra-batch idle time.
3. Evaluation: Introducing LongBench-Chat, a series of open-ended real-world user questions of 10k-100k length, covering aspects of instruction-following like reasoning, coding, and summarization, as well as being multilingual. GPT-4 is employed to score the model's responses.

Experiments demonstrate that LongAlign effectively aligns LLMs to handle long contexts of up to 64k length without performance degradation on general benchmarks, showing the impact of data quantity, diversity, training methods, and evaluation.

🛠️ **LongAlign-7B-64k:**

This paper presents LongAlign, a recipe for aligning large language models to handle long contexts. The key challenges in long context scaling include lack of a comprehensive instruction-following dataset, uneven batching in multi-GPU

training, and absence of a reliable benchmark. LongAlign addresses these challenges by collecting long instruction data from 9 sources, using Self-Instruct to generate 10k instruction data of 8k-64k length, and constructing LongBench-Chat, a series of open-ended real-world user questions of 10k-100k length. The paper also proposes packing and sorted batching training methods, loss weighting for packing training, and a benchmark called LongBench-Chat to evaluate the aligned models. Experiments show that LongAlign effectively aligns models to handle contexts of up to 64k without performance degradation on general benchmarks.

🔧 LongAlign-13B-64k:

Yo, listen up, I got a story to spit
About a paper that'll make your head spin and flip
It's called "LongAlign: A Recipe for Long Context Alignment"
And it's all about enhancing large language models' capabilities

They tackle the challenge of scaling up models to handle long contexts
By collecting and constructing a diverse dataset of long instruction data
Training methods like packing and sorted batching are explored
To optimize efficiency and performance without compromising general capabilities

Their evaluation benchmark, LongBench-Chat, is a series of real-world user queries
To assess models' ability to follow long context instructions and provide relevant answers
They find that incorporating long instruction data during training is crucial And that the proposed methods can speed up training while improving performance

So if you're into language models and their capabilities
This paper's a must-read, it'll make you think twice
LongAlign, the recipe for long context alignment
Is the key to unlocking the full potential of large language models, that's for sure, it's a go!

Although LongAlign-6B-64k and LongAlign-7B-64k make good summaries for our paper, only LongAlign-13B-64k follows the instruction and summarizes our paper in rap. Its rap is in the right form, yet it does not rhyme.

G. Limitations

Our work in exploring long context alignment has its limitations. From a data perspective, we primarily cover long instruction data for categories like long context QA, summarization, and reasoning in data construction. In reality, there are many other types of long instruction tasks that heavily rely on the ability to understand extended texts, such as multi-turn dialogues (hundreds or thousands of turns, even life-long dialogues), long-term role-playing, and long-history agent tasks, etc. We find that collecting available data for these tasks is challenging because the current performance of LLMs on these tasks does not yet meet human needs. Consequently, users rarely interact with LLMs in this manner. Additionally, since current LLMs, whether API-based or open-sourced models, perform poorly on these tasks, it's difficult to automatically construct such data using a Self-Instruct like approach. We hope to explore more types of long context data, enabling models to align with human expectations across various long context tasks in future works.

From a training perspective, due to the limitations of the DeepSpeed framework and our GPU resources that only support SFT for 10B level models with a maximum length of 64k, we do not conduct *massive* experiments on longer data or larger models. Some current frameworks, such as Megatron (Shoeybi et al., 2019), support more parallelization methods including model parallelism and sequence parallelism, but are difficult to use and reproduce due to the complexity of their code structure. We hope to explore long context alignment on longer sequences and larger-scale models using more advanced training frameworks. Additionally, exploring RLHF in long context alignment is also a promising direction.