E^2-LLM: Efficient and Extreme Length Extension of Large Language Models

Jiaheng Liu*^{†,1}, Zhiqi Bai*^{1}, Yuanxing Zhang^{1}, Chenchen Zhang^{1}, Yu Zhang^{1}, Ge Zhang^{2}, Jiakai Wang^{1}, Haoran Que^{1}, Yukang Chen^{3}, Wenbo Su^{1}, Tiezheng Ge^{1}, Jie Fu^{4}, Wenhu Chen^{2}, Bo Zheng^{1}

^{1}Alibaba Group, ^{2}University of Waterloo, ^{3}The Chinese University of Hong Kong
^{4}The Hong Kong University of Science and Technology
`{ljh411989, baizhiqi.bzq}@taobao.com`

Abstract

Training Large Language Models (LLMs) to process extensive context lengths incurs prohibitive computational costs. Prevailing techniques for extending context capabilities in LLMs typically require not only additional training procedures but also access to datasets with long context (e.g., sequences of 32K tokens), presupposing substantial GPU expenditures. To address the aforementioned issues, we introduce a novel solution named Efficient and Extreme length extension for Large Language Models (E^2-LLM). E^2-LLM entails a singular training process over considerably short sequences (e.g., 4K tokens), which greatly mitigates the cost of continual-pretraining or fine-tuning. Within the training phase, we incorporate a dual augmentation strategy with Rotary Position Embeddings (RoPE) that adjusts the scale and position indices across distinct training samples. E^2-LLM is meticulously designed to enhance the model's robustness to diverse relative positions. The experimental results on multiple benchmark datasets demonstrate the superior performance of E^2-LLM on demanding tasks of processing long contexts.

1 Introduction

Large language models are inherently constrained by a predetermined context window length. For instance, in the case of the Llama architecture (Touvron et al., 2023a,b), input sequences are restricted to a maximum of either 2,048 or 4,096 tokens. This preset context window limit could be readily surpassed in applications such as multi-turn dialogues, summarizations of long documents, or intricate reasoning chains (Zheng et al., 2023; Chen et al., 2023a). Thus, LLMs capable of accommodating extensive context windows are thereby highly demanding. Nevertheless, the process of training an LLM with long context windows from scratch

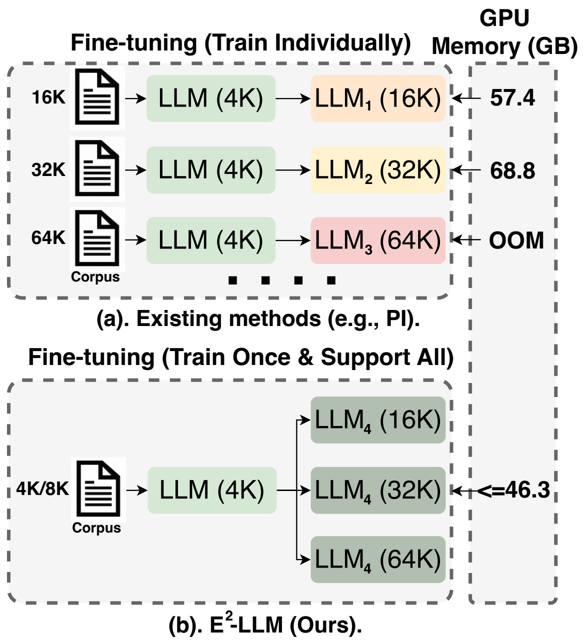

Figure 1: Comparison of existing long-context extension methods (*e.g.*, PI) and E^2-LLM. "LLM (4K)" means the LLM with default context window (*e.g.*, Llama2 with 4K). "LLM$_*$ (16K/32K/64K)" means the LLM with extended context windows (16K/32K/64K) after fine-tuning. (a) Existing methods require individual training over the corresponding long-context data for different target context windows; (b) E^2-LLM only needs a singular training phase with acceptable GPU memory usage and supports different evaluation context windows in inference.

incurs considerable computational overhead and resource expenditure. As a result, prevailing strategies (Peng et al., 2023; Chen et al., 2023b) tend to enhance the context window capacity of an existing LLM as a more resource-efficient alternative.

A commonly employed technique, termed direct extrapolation, fine-tunes an existing pretrained LLM with a more expansive context window. However, the authors of Position Interpolation (PI) (Chen et al., 2023a) observe that this method often leads to slow convergence and can fail when

* First two authors contributed equally.
† Corresponding Author: Jiaheng Liu.

adapting the model to substantially larger context windows. Existing long-context extension methods necessitate the predefinition of an enlarged context length to perform continual-pretraining or fine-tuning (Wang et al., 2024), so that the model could process the extended context through interpolation rather than extrapolation. It is imperative to curate a large collection of data with the designated context length. The expansion of training sequence lengths can precipitate GPU memory to become the main bottleneck for executing the long-context post-training (as exemplified by $\text{LLM}_{1\sim3}$ in Fig. 1 (a)). Certain approaches resort to sliding-skip techniques (Xiao et al., 2023) or quantization (Yang, 2023) to alleviate the GPU memory demands, yet they trade off precision during training, which may be unacceptable for particular tasks demanding precise long-context interpretation.

To address the aforementioned issues, we introduce E^2-LLM which enhances the ability of LLMs to process almost arbitrary sequence lengths through a singular and lossless training phase, thereby obviating the need for pre-specifying a target context length. As demonstrated in Fig. 1(b), E^2-LLM remains the training on the commonly-used data with short lengths (e.g., 4K tokens). The model would be exposed to a wide range of relative positions due to the strategic design of our training paradigm. We also employ positional offsets to prevent LLMs from focusing on a narrow range of absolute positional indices within a prefix span. Thanks to the off-the-shelf inference acceleration technologies (Hong et al., 2023), we can harness the superior performance of our lossless training.

The contributions are summarized as follows:

- Our investigation delves into the limitations of existing long-context extension methods of LLMs, i.e., the heavy fine-tuning costs and the necessity of a large collection of curated long sequences. We then propose E^2-LLM to undertake a singular training procedure on the general short-context data, which minimizes GPU memory occupation without compromising numerical precision.
- E^2-LLM applies a dual augmentation strategy upon RoPE on the scale and position indices across distinct training samples, so that the model accommodates almost arbitrary context length and enhances the long-context capabilities.
- Evaluation on multiple challenging long-context benchmark datasets highlights the effectiveness of the E^2-LLM in processing long contexts. Further analysis verifies the model's robust performance on contexts encompassing up to 120K tokens, despite being trained on sequences restricted to 4K tokens.

2 Related Works

Long-context transformers. Transformers have been used in many tasks (Liu et al., 2022; Guo et al., 2022, 2023b, 2024a). Augmenting the capability of transformer models to process longer text sequences has emerged as a prevalent topic for LLMs. Methodologies such as retrieval-based models have been engaged (Karpukhin et al., 2020; Izacard et al., 2022). Other methods (Wang et al., 2020; Beltagy et al., 2020; Kitaev et al., 2020; Bulatov et al., 2022; Ding et al., 2023) craft approximate alternatives to the attention mechanisms to mitigate the self-attention's intrinsic computational intensity. Wu et al. (2022) further resorts to the memory-based systems to condense previous inputs and recall pertinent components. However, these approaches tend to fall short of the effect of full attention, thereby impeding the refinement of the pretrained LLMs. Our approach diverges by a solution that preserves a close correspondence with the conventional form of attention, evidencing only marginal deviation in performance.

Extension via position encodings. LLMs (Guo et al., 2024b; Bai et al., 2024; Wu et al., 2024; Zhang et al., 2024; Du et al., 2024; Guo et al., 2023a; Sun et al., 2024; Wang et al., 2023; Zhou et al., 2023, 2024) are originally trained with predefined context sizes. Contemporary research has pivoted towards exploring to enhance the context window of LLMs during fine-tuning, due to the prohibitive cost of post-training LLMs on longer sequences. One notable suite of techniques, including Position Interpolation (Chen et al., 2023a), NTK-aware position embeddings (ntk, 2023), PoSE (Zhu et al., 2023), ABF (Xiong et al., 2023), and Yarn (Peng et al., 2023), would alter RoPE (Su et al., 2021) to facilitate Llama's processing of inputs of 32K tokens. These methods still require long-context training data (thereby significant training costs), yet they may perform poorly on the out-of-distribution (OOD) context length. Some other approaches observe that relative positional encodings become almost the same when token distances become substantial, as the result of numerical precision limitations. Such approaches strategically trade some degree of accuracy for enhanced efficiency, such as Landmark attention (Mohtashami and Jaggi, 2023) that compasses extended contexts into a set of retrieved tokens, and StreamingLLM (Xiao et al., 2023) that curtails computational demands on intermediate tokens. In contrast, E^2-LLM is trained via a mechanism that exposes the model to diverse relative positions, so that the context can be extended to nearly arbitrary length in a lossless manner.

3 Background

3.1 Rotary Position Embedding (RoPE)

Transformer models require explicit positional information to be injected, where the positional encodings are used to represent the order of inputs. We take RoPE as an example, which is widely-used in many Llama-style models. In RoPE, given a position index $m \in [0, L)$ and an embedding vector $\mathbf{x} := [x_0, x_1, \ldots, x_{d-1}]^\top$, where L is the context window and d is the dimension of the attention head, a vector-valued complex function $\mathbf{f}(\mathbf{x}, m)$ is defined as follows:

$$\mathbf{f}(\mathbf{x}, m) = [(x_0+\mathrm{i}x_1)e^{\mathrm{i}m\theta_0}, \ldots, (x_{d-2}+\mathrm{i}x_{d-1})e^{\mathrm{i}m\theta_{d/2-1}}]^\top, \tag{1}$$

where $\mathrm{i} := \sqrt{-1}$ is the imaginary unit and $\theta_j = 10000^{-2j/d}$. Based on RoPE, the attention score a is computed as follows:

$$\begin{aligned} a(m,n) &= \mathrm{Re}\langle \mathbf{f}(\mathbf{q}, m), \mathbf{f}(\mathbf{k}, n)\rangle \\ &=: a(m-n), \end{aligned} \tag{2}$$

where $\mathbf{q}$ and $\mathbf{k}$ are the query and key vectors for a specific attention head, respectively. In Eqn. (2), we observe that $a(m, n)$ is only dependent on relative position $m-n$ through trigonometric functions. Besides, RoPE is performed on both query and key embeddings to obtain attention scores at each layer.

3.2 Position Interpolation

Although the attention weight in Eqn. (2) depends only on the relative positions, the model struggles to extrapolate. Specifically, when applying direct extrapolation to larger context windows than the training context window, the perplexity (PPL) will rise significantly (i.e., $> 10^3$).

Let s denote the positional span between a query and a key, and L denote the size of the trained context window. Instead of direct extrapolation on the attention score to $s > L$, the attention score is defined as $\tilde{a}(s) = a(Ls/L')$ in PI, where L' is the extended longer context window. Formally, RoPE $\mathbf{f}$ is replaced by $\mathbf{f}_{\mathrm{PI}}$ as follows:

$$\mathbf{f}_{\mathrm{PI}}(\mathbf{x}, m) = \mathbf{f}\left(\mathbf{x}, \frac{mL}{L'}\right), \tag{3}$$

where the actual position indices are mapped from $[0, L)$ to $[0, L')$. We define the scale parameter g as $\frac{L'}{L}$. For example, g is set as 2 when $L' = 8,192$ for Llama2 with context window of $L = 4,096$. Thus, the Eqn. (3) can be reformulated as follows:

$$\mathbf{f}_{\mathrm{PI}}(\mathbf{x}, m) = \mathbf{f}\left(\mathbf{x}, \frac{m}{g}\right). \tag{4}$$

4 Dual Augmentation of E^2-LLM

In this section, we introduce the details of E^2-LLM for adapting diverse sizes of context windows by a singular training procedure on short-length data, which significantly reduces the tuning costs.

4.1 Notations

Apart from the notations defined in Sec. 3, we also define the following notations. First, let R denote the trained length. In E^2-LLM, R is the maximum length of the data in fine-tuning, i.e., 4K by default. Therefore, it is easy to collect the training data with a length of R and the used GPU memory in fine-tuning is also acceptable. In contrast, R should be equal to the extension length L' (*e.g.*, 16K/32K) in many long-context extension methods (*e.g.*, PI), which requires high GPU memory usage in training. Second, we also introduce the position offset t in RoPE, and we can reformulate Eqn. (4) to compute the position embeddings as:

$$\mathbf{f}'(\mathbf{x}, m; t, g) = \mathbf{f}\left(\mathbf{x}, \frac{m+t}{g}\right). \tag{5}$$

In standard RoPE, by default, t is set as 0. In E^2-LLM, t is selected from a range of indices $\mathcal{T} = \{0, ..., T_{max}\}$, where T_{max} is the maximum position offset. Third, we also define a set of scale parameters used in E^2-LLM as $\mathcal{G} = \{1, 2, ..., G_{max}\}$, where G_{max} is the maximum scale parameter.

4.2 E^2-LLM

We then introduce the dual augmentation strategy of E^2-LLM, which augments on the hyperparameters (*i.e.*, the scale g and the position offset t) of Eqn. (5). Note that we take the pretrained LLM $\mathcal{H}$ with the default context window L as 4,096 and the trained length R as 4,096.

4.2.1 Augmentation on g

The advancements of conventional interpolation algorithms are attributed to the exposure to a varied range of relative positions during post-training, along with a sufficient number of strongly associated token pairs across these relative positions. In this paper, we conceptualize RoPE and accompanying interpolation algorithms as a series of fitting behaviors. These interpolation algorithms often necessitate retraining to adapt to OOD context length, partly due to the overfitting to the relative positions observed during training (analogous to the Runge phenomenon in numerical analysis). Consequently, we augment the training that incorporates a broader spectrum of relative positions, thereby generalizing LLMs to accommodate longer context.

We illustrate the augmentation procedure on the scale parameter g in Fig. 2. Specifically, to make the model $\mathcal{H}$ cover different position densities in training, for the i-th iteration, we sample the scale g_i from $\mathcal{G}$ at different iterations following a predefined probability distribution P as follows:

$$g_i = \mathcal{S}_g(P, \mathcal{G}), g_i \in \mathcal{G}, \tag{6}$$

where $\mathcal{S}_g(P, \mathcal{G})$ denotes the sampling operation on g. Therefore, different scale parameters are used across distinct iterations in training. Note that P is set to uniform distribution by default.

Figure 2 illustrates three different iterations in the training phase, where we fix the position offset t to 0. Suppose that the three randomly selected gs are 2, 5, and 10, respectively. It is obvious that within the singular training phase, the maximum interpolated context windows (horizontal axis), as well as the densities of the trained position indices (blue dots), are different among the three iterations. Note that, as the training context window R is less than the interpolated maximum context windows, only a certain proportion of the position indices (blue dots) would be trained in this setting.

4.2.2 Augmentation on t

As we can only focus on a small range of position indices when we start from zero index (i.e., $t = 0$), to improve the robustness and generalization ability of E^2-LLM, we further introduce the augmentation on the position offset t by changing the absolute position indices of RoPE. Inspired by recent works (Han et al., 2023; Xiao et al., 2023) which allocate a high proportion of attention weights to the initial tokens (i.e., attention sinks (Xiao et al., 2023)), we preserve several initial tokens and set the position offsets of these tokens as 0. Note that the number of initial tokens is set to 4 by default. For the remaining position indices, in the i-th training iteration, we sample the position offset t_i following a pre-set probability distribution, denoted as Q. Accordingly, t_i is set as follows:

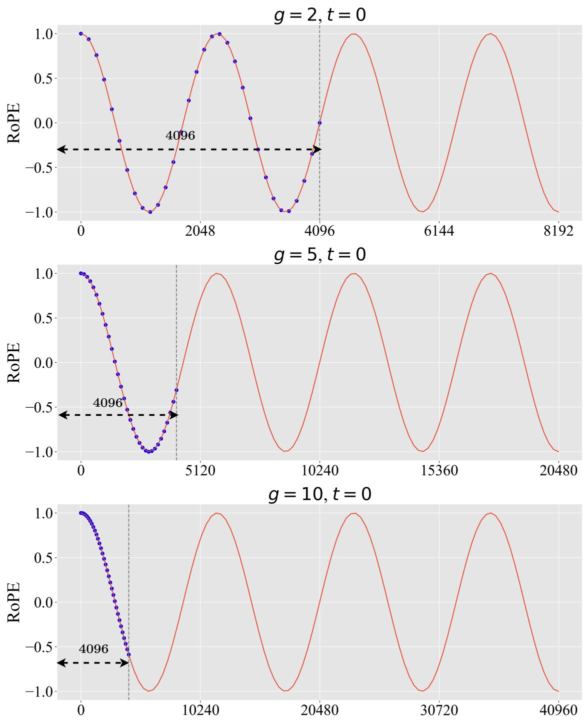

Figure 2: The trained position indices (blue dots) when using different scale parameters (i.e., $g = 2, 5, 10$). The maximum length of the fine-tuning data (i.e., R) is 4,096 and the position offset t is set to 0 for illustration.

$$t_i = \begin{cases} 0, & m \in [0, 3] \\ \mathcal{S}_t(Q, \mathcal{T}), & m \in (3, R) \end{cases}, \tag{7}$$

where $\mathcal{S}_t(Q, \mathcal{T})$ denotes the sampling operation on t. Note that Q is set as a uniform distribution and T_{max} is assigned by the difference between the maximum interpolated context window and the trained context window in the current iteration. Based on Eqn. (7), for $n \in [0, 3]$ and $m \in (3, R)$, the Eqn. (2) can be depicted by:

$$\begin{aligned} a(m, n) &= \text{Re}\langle \mathbf{f}'(\mathbf{q}, m; t_i, g_i), \mathbf{f}'(\mathbf{k}, n; t_i, g_i) \rangle \\ &=: a(m + \mathcal{S}_t(Q, \mathcal{T}) - n). \end{aligned} \tag{8}$$

When $\mathcal{S}_t(Q, \mathcal{T})$ yields a high value, the range of relative position differences (i.e., $(m + \mathcal{S}_t(Q, \mathcal{T}) - n)$) between m and n would become significantly large. Fig. 3 illustrates the trained position indices (blue

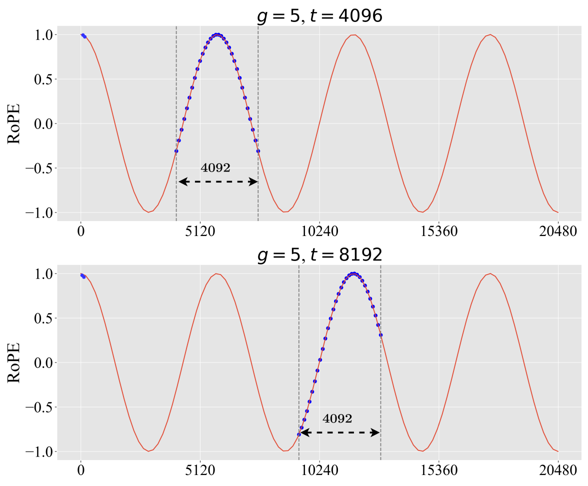

Figure 3: The trained position indices (blue dots) when using different position offsets and g is 5. The position indices of the first four tokens are not changed.

Algorithm 1 Training of E^2-LLM

```
Input: Pretrained LLM model $\mathcal{H}$ with default context window of $L$ (e.g., 4K); The trained context window $R$ (e.g., 4K/8K); The evaluation context window $L'$ (e.g., 32K/64K);
1: for the $i$-th iteration in training do
2:     Get the scale $g_i$ based on Eqn. (6);
3:     Get the position offset $t_i$ based on Eqn. (7);
4:     Modify the RoPE position embeddings based on Eqn. (5);
5:     Train model $\mathcal{H}$ on training window $R$;
6:     Compute the next token prediction loss;
7:     Update parameters of model $\mathcal{H}$;
8: end for
Output: The optimized long context model $\mathcal{H}'$. (Note that $\mathcal{H}'$ can extend to different context windows at inference.);
```

dots) with the position offset t, indicating that E^2-LLM can easily make use of the position indices with different absolute values. Hence, the argumentation on t makes LLMs generalize to diverse range of relative differences.

4.2.3 Training and Inference

Training. The training procedures of E^2-LLM method are summarized in Alg. 1. We replace g and t with g_i and t_i in Eqn. (5) for the i-th iteration in the training phase. $\mathcal{H}$ is fine-tuned by a short context window R under the next token prediction task, leveraging the modified position encodings. **Inference**. E^2-LLM also does not introduce extra trainable parameters, nor modify the network architecture. This is essential in practical applications as most infrastructure and optimization strategies for the original model remain available after context extension. During inference, we can extend to different context windows by merely setting the corresponding scale parameters for interpolation. For example, we set $g = 8$ for extending to $32,768$ and $g = 16$ for extending to $65,536$, called as E^2-LLM-32K and E^2-LLM-64K, respectively. Here, the parameters of E^2-LLM-32K and E^2-LLM-64K are the same, where the only difference is that the scale parameters are set to 8 and 16, respectively. In practice, we can thereby deploy only one LLM on devices, and automatically change the scale parameter of RoPE based on the designated length of input context to support different context windows.

5 Experiments

5.1 Experimental Settings

Models. We take the pretrained 7B and 13B Llama2 (Touvron et al., 2023b) as the base models to demonstrate the effectiveness of E^2-LLM.

Training procedure. All models are fine-tuned via the next token prediction objective based on two 8×A100-80GB GPU machines. We use AdamW (Loshchilov and Hutter, 2019) with $\beta_1 = 0.9$ and $\beta_2 = 0.95$. The learning rate is set to 1×10^{-5}, and the number of training iterations is set to 30,000 with a global batch size of 16.

Datasets. The training dataset includes pretrain dataset (i.e., Pile (Gao et al., 2020)), and fine-tuning datasets (i.e., ShareGPT (Zheng et al., 2023) and the long summarization datasets (Cohan et al., 2018)). Note that the fine-tuning datasets are used to improve the question-answer abilities of long-context LLMs following Vicuna and LongChat (Zheng et al., 2023) to generate reasonable results on LongBench. We evaluate the long-sequence language modeling performance on the LongBench (Bai et al., 2023) and Arxiv proof-pile (Azerbayev et al., 2022) datasets.

5.2 Results on LongBench

We evaluate several representative LLMs with long context capability, including GPT-3.5-Turbo-16K (OpenAI, 2022), Llama2-7B-chat-4K (Touvron et al., 2023b), LongChat-v1.5-7B-32K (Li et al., 2023), Vicuna-v1.5-7B-16K (Zheng et al., 2023), Longlora-7B-16K (Chen et al., 2023b), Llama2-13B-chat-4K (Touvron et al.,

Table 1: Results (%) on single-doc QA, multi-doc QA and summarization tasks from LongBench dataset.

Model	Single-Doc QA				Multi-Doc QA				Summarization	
	NarrativeQA	Qasper	MultiField QA-en	MultiField QA-zh	Hotpot QA	2WikiMulti hopQA	MuSi Que	Du Reader	Gov Report	QMSum
GPT-3.5-Turbo-16K	23.6	43.3	52.3	61.2	51.6	37.7	26.9	28.7	29.5	23.4
Llama2-7B-chat-4K	18.7	19.2	36.8	11.9	25.4	32.8	9.4	5.2	27.3	20.8
LongChat-v1.5-7B-32K	16.9	27.7	41.4	29.1	31.5	20.6	9.7	19.5	30.8	22.7
Vicuna-v1.5-7B-16K	19.4	26.1	38.5	43.0	25.3	20.8	9.8	19.3	27.9	22.8
LongLora-7B-16K	19.8	29.1	37.2	8.5	37.0	30.3	17.1	15.3	31.5	24.1
E^2-LLM-Llama2-7B-16K	16.4	34.7	39.1	43.6	37.1	34.4	17.9	18.6	29.4	23.0
E^2-LLM-Llama2-7B-32K	12.3	35.6	40.4	46.6	43.7	34.8	22.0	22.6	29.7	23.8
Llama2-13B-chat-4K	19.2	25.8	36.9	33.3	36.1	32.4	14.5	26.8	26.6	20.2
Vicuna-v1.5-13B-16K	18.9	29.9	46.2	28.4	38.1	36.0	10.7	20.9	27.9	22.1
PI-Llama2-13B-16K	19.2	33.3	42.7	47.9	44.9	34.8	19.5	17.4	27.9	23.7
E^2-LLM-Llama2-13B-16K	25.4	35.3	46.5	49.1	46.4	38.3	25.2	19.3	29.9	22.7
E^2-LLM-Llama2-13B-32K	24.1	36.2	49.0	52.5	49.2	37.6	23.1	20.4	29.9	23.1

Table 2: Results (%) on summarization, few-shot learning, synthetic, and code tasks from LongBench dataset. 'Overall' is computed by the macro-average over major task categories. This is computed on English (EN) tasks, Chinese (ZH) tasks, and all (All) tasks, code tasks are included in both languages.

Model	Summarization		Few-shot Learning				Code		Overall		
	MultiNews	VCSUM	TREC	TriviaQA	SAMSum	LSHT	LCC	RepoBench-P	EN	ZH	All
GPT-3.5-Turbo-16K	26.7	16.0	68.0	91.4	41.7	29.2	54.7	53.6	44.60	33.78	42.19
Llama2-7B-chat-4K	25.8	0.2	61.5	77.8	40.7	19.8	52.4	43.8	35.17	15.45	20.79
LongChat-v1.5-7B-32K	26.4	9.9	63.5	82.3	34.2	23.2	53.0	55.3	36.86	20.43	33.21
Vicuna-v1.5-7B-16K	27.2	15.1	74.0	86.2	40.8	28.8	51.0	43.5	36.49	26.55	34.28
LongLora-7B-16K	27.7	0.5	63.5	85.7	41.9	26.0	57.6	54.5	39.79	14.55	34.18
E^2-LLM-Llama2-7B-16K	25.9	9.6	68.5	89.2	38.2	35.0	65.8	58.1	41.26	26.70	38.03
E^2-LLM-Llama2-7B-32K	25.4	11.7	70.5	88.4	32.5	40.0	64.5	60.9	**41.74**	**30.23**	**39.18**
Llama2-13B-chat-4K	26.1	17.2	66.0	85.2	36.5	20.3	51.9	52.8	37.87	24.38	34.87
Vicuna-v1.5-13B-16K	27.1	16.4	74.0	84.9	27.8	29.8	44.1	45.6	38.08	23.86	34.92
PI-Llama2-13B-16K	25.9	9.1	72.5	86.5	27.9	31.0	62.5	51.1	40.88	26.35	37.65
E^2-LLM-Llama2-13B-16K	27.0	9.8	73.5	87.9	40.6	36.0	65.4	59.1	**44.73**	28.56	41.13
E^2-LLM-Llama2-13B-32K	26.8	10.2	75.0	87.8	40.9	44.5	63.8	57.5	44.55	**31.93**	**41.74**

2023b), Vicuna-v1.5-13B-16K (Zheng et al., 2023), PI-Llama2-13B-16K. LongChat-v1.5-7B-32K, Vicuna-v1.5-7B-16K, and LongLora-7B-16K are fine-tuned from Llama2-7B with PI. Vicuna-v1.5-13B-16K (Zheng et al., 2023) and PI-Llama2-13B-16K are fine-tuned with Llama2-13B with PI, where PI-Llama2-13B-16K are fine-tuned via the abovementioned training datasets. We follow the settings in LongBench, i.e., either zero-shot or few-shot for each specific subtask. When the input length surpasses the maximum context length of a model (indicated by the suffix of its name), we truncate the input sequence from the middle. The measurement score of each subtask ranges from 0 to 100, the higher the better.

Tables 1 and 2 present the performance metrics on the LongBench dataset. The salient observations are as follows: (1) E^2-LLM-Llama2-13B-32K demonstrates comparable performance to GPT-3.5-Turbo-16K, with our model achieving an overall accuracy of 44.55% versus 44.60% for the latter in English tasks. (2) Using varying context sizes (16K and 32K tokens), we find that the model exhibits enhanced performance with an increased context window size. (3) To ensure a fair comparison, we meticulously fine-tune the Llama2-13B with canonical PI (denoted as PI-Llama2-13B-16K), employing identical training settings and datasets. The results reveal that E^2-LLM-Llama2-13B-16K consistently surpasses PI-Llama2-13B-16K by an average margin of 9%, thereby underscoring the efficacy of E^2-LLM.

5.3 Results on Proof-Pile

In Table 3, we report the perplexity scores as an indicator of model performance over extended context using the curated ArXiv Mathematics Proof-Pile dataset (Azerbayev et al., 2022). We randomly select 128 documents from the dataset, each containing no fewer than 64K tokens, and assess the perplexity of these samples. We apply the sliding window method with a window size of 256 tokens to compute perplexity values. Notably, the models Vicuna-v1.5-16K and LongChat-v1.5-32K are fine-tuned on the Llama2 architecture, wherein RoPE is scaled linearly. As previously highlighted, E^2-LLM can be easily adapted to the 64K context scenarios. Moreover, the results in Table 3 reveal a remarkable reduction in perplexity for models augmented with E^2-LLM, particularly noticeable with larger context window sizes.

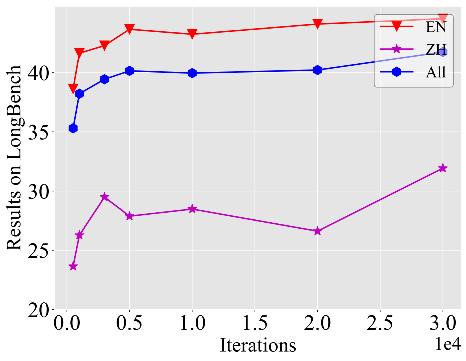

Figure 4: Performance along with training iterations.

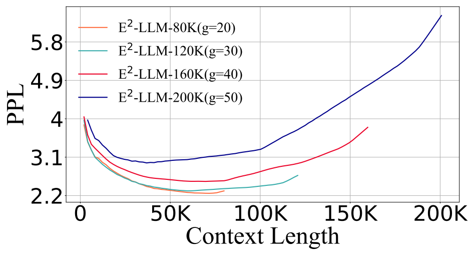

Figure 5: Generalization abilities on the unseen scales.

5.4 Ablation Study

Effect of the augmentation strategies. We provide two variants of E^2-LLM to verify the standalone results of the augmentation strategies as shown in Section 4: “E^2-LLM(no offset)”, in which the positional offset is omitted (i.e., $t \equiv 0$), and “E^2-LLM(fixed scale)”, which represents the standard PI (target context length as 16K) with positional offset (i.e., $g \equiv 4$). As shown in Table 4, these two alternatives still outperform PI (in Table 2) on LongBench. Meanwhile, the full E^2-LLM exhibits additional improvements, indicating the effectiveness of the two augmentation strategies.

Effect of the number of initial fixed tokens. By design, we set the number of initial fixed tokens as four. The results in Table 5 show the accuracy on LongBench of deviations from this predefined number of initial tokens. The setting involving four initial tokens seems to sufficiently bootstrap the model's performance, and the further increase will not produce any improvements.

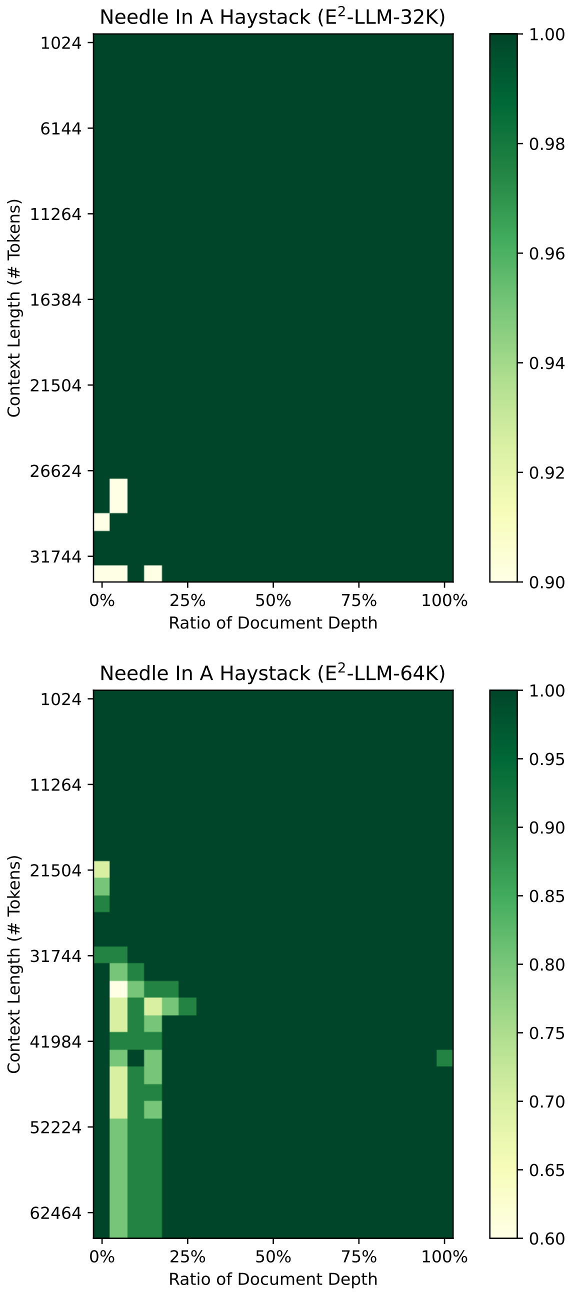

Figure 6: Pressure Test (“Needle In A Haystack”) of E^2-LLM.

Effect of the number of fine-tuning steps. Fig. 4 illustrates the performance of E^2-LLM-Llama2-13B-32K on LongBench across varying fine-tuning iterations. During the initial 5K iterations, there is a remarkable rise in accuracy, suggesting that the model swiftly acquires long-context process-

Table 3: Evaluation of PPL on Arxiv Proof-pile based on Llama2 7B and 13B, where lower perplexity means better performance. "PI" denotes Position Interpolation (Chen et al., 2023a). The open-sourced Vicuna-v1.5-16K and LongChat-v1.5-32K are extended based on PI. Note that weights of E^2-LLM-16K, E^2-LLM-32K, and E^2-LLM-64K are the same at inference, and the only difference is that the scales are set as 4, 8 and 16, respectively.

Model			Evaluation Context Window Size				
Size	Training Context Window	Method	4,096	8,192	16,384	32,768	65,536
7B	4K	None	2.92	-	-	-	-
7B	16K	Vicuna-v1.5-16K (PI)	3.48	3.17	3.95	-	-
	32K	LongChat-v1.5-32K (PI)	3.54	3.18	2.91	2.73	-
7B	4K	E^2-LLM-16K	2.96	2.71	2.54	-	-
		E^2-LLM-32K	2.99	2.74	2.56	2.46	-
		E^2-LLM-64K	3.06	2.81	2.62	2.51	2.56
13B	4K	None	2.78	-	-	-	-
13B	16K	Vicuna-v1.5-16K (PI)	3.27	2.97	2.78	-	-
13B	4K	E^2-LLM-16K	2.82	2.59	2.43	-	-
		E^2-LLM-32K	2.85	2.61	2.44	2.34	-
		E^2-LLM-64K	2.91	2.67	2.49	2.39	2.44

Table 4: Ablation on different augmentation strategies.

Methods	EN	ZH	All
E^2-LLM	**44.55**	**31.93**	**41.74**
E^2-LLM (no offset)	42.28	29.49	39.44
E^2-LLM (fixed scale)	41.66	28.33	38.77

Table 5: Ablation on the number of initial tokens.

# Initial Tokens	0	2	4	6
EN	42.93	43.94	44.55	44.23
ZH	29.55	30.65	31.93	32.82
All	39.95	40.99	41.74	41.69

ing capabilities in E^2-LLM. When proceeding with training, the accuracy still presents a consistent upward tendency, indicating possibly further gains in the model's ability to handle long-range dependencies.

5.5 Further Analysis

Extension to unseen scales. By default, we set G_{max} as 20 to support the maximum interpolated context window of 80K. In Fig. 5, the interpolation scales are experimentally adjusted to 20, 30, 40, and 50 during inference to evaluate the generalization ability of E^2-LLM. The results demonstrate that PPL maintains a satisfactory level for contexts comprising fewer than 120K tokens. Nonetheless, when we continue to increase the scale, a discernible deterioration in performance occurs. It suggests that E^2-LLM possesses robust generalization ability for unseen scales within a certain range.

Visualization on pressure test. Meanwhile, we take the E^2-LLM-32K and E^2-LLM-64K based on Llama2-13B as examples to provide the pressure test on E^2-LLM. The pressure test (i.e., "Needle In A Haystack") denotes to test the long-context abilities under different evaluation context windows, as shown in Fig. 6. Specifically, in the pressure test, we first prepare the documents truncated to 32K and 64K, respectively. Then, we randomly insert the reference information at different ratios (e.g., 5%, 75%) of the document depth to generate the input context. The reference information provided herein is an intentionally fabricated fact. It is designed to ensure consistency and non-conflict with the context, representing a minute detail. The model requires a robust ability to resist interference to accurately identify the correct answer. After that, the model takes the input context as input and generates the answer. Finally, we calculate the accuracies of the ten documents under different ratios of document depth and different evaluation context lengths. From Fig. 6, we observe that E^2-LLM-32K and E^2-LLM-64K achieve superior performance for the pressure test. It indicates that despite being trained once within 4K sequences, E^2-LLM presents reliable capabilities with the appropriate interpolation scale.

6 Conclusion

In this study, we propose E^2-LLM to extend the context windows with a singular training phase and minimal computational overhead. E^2-LLM harnesses RoPE-based position embeddings to implement a novel dual augmentation strategy that manipulates the interpolation scale and positional offset in training examples of short lengths (e.g., 4K/8K). Comprehensive experimental results demonstrate the superior performance of E^2-LLM on long-context processing tasks.

Ethical Issues and Limitations

E^2-LLM is designed to enhance the capability of LLMs to process long context via technical solutions. There are no ethical issues for applying E^2-LLM to existing LLMs. However, the major limitation of deploying E^2-LLM comes from the inference phase. Although E^2-LLM introduces lossless training, the best performance still depends on the highly efficient inference infrastructure.

References

2023. Ntk-aware scaled rope.

Zhangir Azerbayev, Edward Ayers, and Bartosz Piotrowski. 2022. Proof-pile.

Ge Bai, Jie Liu, Xingyuan Bu, Yancheng He, Jiaheng Liu, Zhanhui Zhou, Zhuoran Lin, Wenbo Su, Tiezheng Ge, Bo Zheng, and Wanli Ouyang. 2024. Mt-bench-101: A fine-grained benchmark for evaluating large language models in multi-turn dialogues. *arXiv*.

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2023. Longbench: A bilingual, multitask benchmark for long context understanding. *arXiv preprint arXiv:2308.14508*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. 2022. Recurrent memory transformer. In *NeurIPS*.

Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023a. Extending context window of large language models via positional interpolation. *CoRR*, abs/2306.15595.

Yukang Chen, Shengju Qian, Haotian Tang, Xin Lai, Zhijian Liu, Song Han, and Jiaya Jia. 2023b. Longlora: Efficient fine-tuning of long-context large language models. *arXiv:2309.12307*.

Arman Cohan, Franck Dernoncourt, Doo Soon Kim, Trung Bui, Seokhwan Kim, Walter Chang, and Nazli Goharian. 2018. A discourse-aware attention model for abstractive summarization of long documents. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 615–621, New Orleans, Louisiana. Association for Computational Linguistics.

Jiayu Ding, Shuming Ma, Li Dong, Xingxing Zhang, Shaohan Huang, Wenhui Wang, Nanning Zheng, and Furu Wei. 2023. Longnet: Scaling transformers to 1, 000, 000, 000 tokens. *CoRR*, abs/2307.02486.

Xinrun Du, Zhouliang Yu, Songyang Gao, Ding Pan, Yuyang Cheng, Ziyang Ma, Ruibin Yuan, Xingwei Qu, Jiaheng Liu, Tianyu Zheng, Xinchen Luo, Guorui Zhou, Binhang Yuan, Wenhu Chen, Jie Fu, and Ge Zhang. 2024. Chinese tiny llm: Pretraining a chinese-centric large language model.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, et al. 2020. The pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

Hongcheng Guo, Jiaheng Liu, Haoyang Huang, Jian Yang, Zhoujun Li, Dongdong Zhang, Zheng Cui, and Furu Wei. 2022. Lvp-m3: language-aware visual prompt for multilingual multimodal machine translation. *EMNLP*.

Hongcheng Guo, Jian Yang, Jiaheng Liu, Jiaqi Bai, Boyang Wang, Zhoujun Li, Tieqiao Zheng, Bo Zhang, Qi Tian, et al. 2024a. Logformer: A pretrain and tuning pipeline for log anomaly detection. *AAAI*.

Hongcheng Guo, Jian Yang, Jiaheng Liu, Liqun Yang, Linzheng Chai, Jiaqi Bai, Junran Peng, Xiaorong Hu, Chao Chen, Dongfeng Zhang, et al. 2023a. Owl: A large language model for it operations. *arXiv preprint arXiv:2309.09298*.

Jinyang Guo, Jiaheng Liu, Zining Wang, Yuqing Ma, Ruihao Gong, Ke Xu, and Xianglong Liu. 2023b. Adaptive contrastive knowledge distillation for bert compression. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 8941–8953.

Jinyang Guo, Jianyu Wu, Zining Wang, Jiaheng Liu, Ge Yang, Yifu Ding, Ruihao Gong, Haotong Qin, and Xianglong Liu. 2024b. Compressing large language models by joint sparsification and quantization. *ICML*.

Chi Han, Qifan Wang, Wenhan Xiong, Yu Chen, Heng Ji, and Sinong Wang. 2023. Lm-infinite: Simple on-the-fly length generalization for large language models. *CoRR*, abs/2308.16137.

Ke Hong, Guohao Dai, Jiaming Xu, Qiuli Mao, Xiuhong Li, Jun Liu, Kangdi Chen, Hanyu Dong, and

Yu Wang. 2023. Flashdecoding++: Faster large language model inference on gpus. *arXiv preprint arXiv:2311.01282*.

Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models. *CoRR*, abs/2208.03299.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *EMNLP*, pages 6769–6781.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *ICLR*.

Dacheng Li, Rulin Shao, Anze Xie, Ying Sheng, Lianmin Zheng, Joseph E. Gonzalez, Ion Stoica, Xuezhe Ma, and Hao Zhang. 2023. How long can open-source llms truly promise on context length?

Jiaheng Liu, Tan Yu, Hanyu Peng, Mingming Sun, and Ping Li. 2022. Cross-lingual cross-modal consolidation for effective multilingual video corpus moment retrieval. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1854–1862.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Amirkeivan Mohtashami and Martin Jaggi. 2023. Landmark attention: Random-access infinite context length for transformers. *CoRR*, abs/2305.16300.

OpenAI. 2022. Introducing chatgpt.

Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2023. Yarn: Efficient context window extension of large language models. *CoRR*, abs/2309.00071.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. 2021. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864.

Tao Sun, Linzheng Chai, Yuwei Yin Jian Yang, Hongcheng Guo, Jiaheng Liu, Bing Wang, Liqun Yang, and Zhoujun Li. 2024. Unicoder: Scaling code large language model via universal code. *ACL*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. 2020. Linformer: Self-attention with linear complexity. *CoRR*, abs/2006.04768.

Xindi Wang, Mahsa Salmani, Parsa Omidi, Xiangyu Ren, Mehdi Rezagholizadeh, and Armaghan Eshaghi. 2024. Beyond the limits: A survey of techniques to extend the context length in large language models. *arXiv preprint arXiv:2402.02244*.

Zekun Moore Wang, Zhongyuan Peng, Haoran Que, Jiaheng Liu, Wangchunshu Zhou, Yuhan Wu, Hongcheng Guo, Ruitong Gan, Zehao Ni, Man Zhang, Zhaoxiang Zhang, Wanli Ouyang, Ke Xu, Wenhu Chen, Jie Fu, and Junran Peng. 2023. Rolellm: Benchmarking, eliciting, and enhancing role-playing abilities of large language models. *arXiv preprint arXiv: 2310.00746*.

Yanan Wu, Jie Liu, Xingyuan Bu, Jiaheng Liu, Zhanhui Zhou, Yuanxing Zhang, Chenchen Zhang, Zhiqi Bai, Haibin Chen, Tiezheng Ge, Wanli Ouyang, Wenbo Su, and Bo Zheng. 2024. Conceptmath: A bilingual concept-wise benchmark for measuring mathematical reasoning of large language models. *arXiv*.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. In *ICLR*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv*.

Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, et al. 2023. Effective long-context scaling of foundation models. *arXiv preprint arXiv:2309.16039*.

Jianxin Yang. 2023. Longqlora: Efficient and effective method to extend context length of large language models. *arXiv preprint arXiv:2311.04879*.

Ge Zhang, Scott Qu, Jiaheng Liu, Chenchen Zhang, Chenghua Lin, Chou Leuang Yu, Danny Pan, Esther Cheng, Jie Liu, Qunshu Lin, Raven Yuan, Tuney

Zheng, Wei Pang, Xinrun Du, Yiming Liang, Yinghao Ma, Yizhi Li, Ziyang Ma, Bill Lin, Emmanouil Benetos, Huan Yang, Junting Zhou, Kaijing Ma, Minghao Liu, Morry Niu, Noah Wang, Quehry Que, Ruibo Liu, Sine Liu, Shawn Guo, Soren Gao, Wangchunshu Zhou, Xinyue Zhang, Yizhi Zhou, Yubo Wang, Yuelin Bai, Yuhan Zhang, Yuxiang Zhang, Zenith Wang, Zhenzhu Yang, Zijian Zhao, Jiajun Zhang, Wanli Ouyang, Wenhao Huang, and Wenhu Chen. 2024. Map-neo: Highly capable and transparent bilingual large language model series. *arXiv preprint arXiv: 2405.19327*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric. P Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena.

Zhanhui Zhou, Jie Liu, Zhichen Dong, Jiaheng Liu, Chao Yang, Wanli Ouyang, and Yu Qiao. 2024. Emulated disalignment: Safety alignment for large language models may backfire! *arXiv preprint arXiv:2402.12343*.

Zhanhui Zhou, Jie Liu, Chao Yang, Jing Shao, Yu Liu, Xiangyu Yue, Wanli Ouyang, and Yu Qiao. 2023. Beyond one-preference-for-all: Multi-objective direct preference optimization. *arXiv preprint arXiv:2310.03708*.

Dawei Zhu, Nan Yang, Liang Wang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2023. Pose: Efficient context window extension of llms via positional skip-wise training.