

More Context Needed: Expanding the Context Windows of LLMs with Minimal Finetuning

ALEXANDER HORATIO JEPHTHA

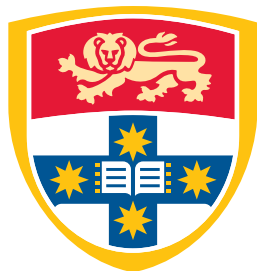
SID: 520446056

Supervisor: Dr. Jonathan K. Kummerfeld

This thesis is submitted in partial fulfillment of
the requirements for the degree of
Bachelor of Science (Honours)

School of Information Technologies
The University of Sydney
Australia

30 January 2026



THE UNIVERSITY OF
SYDNEY

Student Plagiarism: Compliance Statement


I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: Alexander Horatio Jephtha

Signature: 

Date: 7/11/2025

Abstract

Large language models (LLMs) are trained to process text within a fixed-size context window. In real-world uses of LLMs, this context window may not be large enough to span the entire input. Examples of such uses include document summarisation, information retrieval, and information synthesis. Extending the context window size enables an LLM to perform calculations on the entire input, hence producing more accurate outputs. However, LLMs have not been trained to handle these larger inputs and thus perform poorly without any additional modifications. This problem is known as the context extension problem.

There have been various approaches to solving the context extension problem, but none of which have produced standout results across long context benchmarks. Firstly, we argue that existing variants of Rotary Position Embeddings (RoPE) context extension methods make an unnecessary assumption that the magnitude of the RoPE angle between two tokens is directly proportional to their relative distance. From this, we derive and motivate Fractional RoPE as a method that breaks this assumption while satisfying other desirable constraints. Inspired by NRoPE’s early exploration of hybrid embeddings (Yang et al., 2025), we also propose hybrid context extension methods that employ two or more existing methods. Specifically, we define ARoPE, which alternates between ALiBi and RoPE positional embeddings between layers, and HyPE, which uses both RoPE and ALiBi in every layer. We evaluated our methods with and without 32000 finetuning examples on the CommonPile filtered PG-19 dataset, against LongBench (Bai et al., 2024), perplexity against the training set, and manual example analysis.

Our results demonstrate that fractional RoPE outperforms baseline context extension methods in summarisation and few-shot learning tasks, validating our rejection of the linear RoPE angle assumption. In contrast, our hybrid methods do not produce any significant results. This reveals the importance of minimising changes to model architecture when performing context extension, especially with minimal or no finetuning. We also validate perplexity as a highly correlated performance metric across several long-context tasks, and explore why specific tasks are more insightful for assessing context extension performance. Ultimately, our work establishes new and promising ideas that may spark further developments in context extension research.

Acknowledgements

Completing this thesis was one of the most difficult and rewarding experiences I have had during my studies at the University of Sydney. I was forced to overcome many challenges and roadblocks throughout my research, often faced with failure in my first attempts. I would like to first thank my supervisor, Jonathan Kummerfeld, for his extraordinary support, giving me the freedom to explore any area of Natural Language Processing of my choosing, regardless of his own research interests. Despite spending weeks dealing with complicated implementation-based problems, he would never judge my progress and instead suggest ways to work through my issues. My work with Jonathan allowed me to understand what it means to be a good researcher and furthermore, what it means to be a supportive supervisor who brings out the best in their students.

In addition, I would like to thank the open-source machine learning communities and prior research that my work was built on. The PyTorch and Transformer libraries underpinned the majority of my implementations, and the extensive documentation and community support made understanding their codebases much easier. I also acknowledge the authors of LongBench for designing a holistic open-source evaluation benchmark, as well as the original authors of any data used to train my models. Finally, I want to thank my family for financially supporting me through school and university in my pursuit of computer science. I also want to thank my friends and lab group, with whom I practised my presentations, provided feedback on my work, and witnessed my painful debugging sessions.

CONTENTS

Student Plagiarism: Compliance Statement	ii
Abstract	iii
Acknowledgements	iv
List of Figures	viii
List of Tables	x
Chapter 1 Introduction	1
Chapter 2 Literature Review	5
2.1 Early Work.....	5
2.1.1 Transformer Models.....	6
2.1.2 The Context Extension Problem.....	7
2.2 Positional Approaches.....	8
2.2.1 Positional Reorganisation.....	9
2.2.2 Positional Interpolation.....	10
2.3 Rotary Approaches.....	11
2.3.1 Rotary Positional Embeddings.....	12
2.3.2 NTK-Aware RoPE.....	13
2.3.3 Yet another Rope Extension.....	14
2.3.4 Resonance RoPE.....	14
2.3.5 Interpolation from a Distributional Perspective.....	15
2.4 Other Embedding Methods.....	17
2.4.1 Attention with Linear Biases.....	17
2.4.2 In-context Autoencoder.....	18
2.4.3 No Positional Encodings.....	18
2.4.4 RoPE to NoPE and Back Again.....	19

2.5	Designing Better Context Extension Methods	19
2.6	Benchmarks	20
2.6.1	LongPerplexity	20
2.6.2	LongEmbed	21
2.6.3	LongBench	22
2.6.4	Other benchmarks and datasets	22
Chapter 3	Methods	24
3.1	RoPE with non-linear positional dependencies	24
3.1.1	Angle-position independent RoPE variants	25
3.1.2	Fractional RoPE	29
3.1.3	Further exploration of Fractional RoPE	31
3.2	Hybrid methods	34
3.3	Summary of Methods Evaluated	34
Chapter 4	Experiments	36
4.1	Base Model	36
4.1.1	Finetuning	38
4.2	Evaluation	39
4.2.1	LongBench	40
4.2.2	Perplexity	41
Chapter 5	Results	43
5.1	Non-finetuned Models	43
5.1.1	Baseline RoPE Models	43
5.1.2	Fractional RoPE Models	45
5.1.3	Non-RoPE Methods	45
5.2	32k-finetuned Models	48
5.2.1	Baseline RoPE Models	48
5.2.2	Fractional RoPE Models	49
5.2.3	Non-RoPE Methods	49
5.3	Perplexities	53
5.3.1	Non-finetuned Perplexities	53
5.3.2	32k-finetuned Perplexities	54

Chapter 6 Discussion	56
6.1 Research Question 1	56
6.1.1 Non-finetuned Models	56
6.1.2 32k-finetuned Models	60
6.2 Research Question 2	64
6.2.1 LongBench Performance Changes	64
6.2.2 Loss Analysis	68
6.3 Research Question 3	71
6.3.1 Comparison Between LongBench Tasks	71
6.3.2 Perplexity as an Indicator of Performance	74
6.4 Limitations and Future Work	75
Chapter 7 Conclusion	79
Bibliography	80
Appendix A Github Repository	84

List of Figures

1.1	An illustration of a next-token prediction task for the “?” token. The red braces represent the span of the old context window of size $L = 4$ words, whereas the blue braces represent the span of the new context window of size $L' = 8$ words.	2
2.1	Comparison of how perplexity on the Proof-Pile dataset changes as context window increases from Peng et al. (2024). At a large enough context window size, the perplexity for every model will begin to increase.	9
2.2	An illustration of how DCA combines intra-chunk, inter-chunk and successive-chunk attention from An et al. (2024). Note that intra-chunk attention is identical to PCW from Ratner et al. (2023).	11
2.3	An illustration of how Resonance RoPE proposed by Wang et al. (2024b) influences the distribution of rotary angles across a single complex dimension.	16
2.4	An illustration of the training process of the in-context autoencoder method designed by Ge et al. (2024).	18
2.5	A summary of the distribution of different task types in the LongBench benchmark, alongside the distributions of the two different languages and their corresponding task token lengths.	22
3.1	Fractional rope scaling under different α values for $L = 4096$ and $L' = 2L = 8192$. Each value of α has a distinct colour.	32
3.2	Fractional rope scaling under different α values for $L = 4096$ and $L' = 1.5L = 6144$. We see g_α squeezed closer together across all values of α .	32
6.1	An example of the outputs of 5 different non-finetuned models on the same prompt. The correct answer is any way of expressing the United States of America. We clearly see that the RoPE-based models find the correct answer, whereas the selected non-RoPE models hallucinate.	58

- 6.2 An example of the outputs of 3 different non-finetuned models on the same prompt from PRE. The correct answer is 30. It is difficult to understand why models get the answer wrong in this task, since they simply return a number. 59
- 6.3 An example of the outputs of 3 different models on the same prompt from MSQ. The correct answer is The University of Toronto. We see that most models, with the exception of ALiBi, generate too many tokens, resulting in them being penalised by the Q&A F1-score metric used by MSQ. 62
- 6.4 An example of the outputs of 4 different models on the same prompt from MFQA. The correct answer is “The sides of the fuselage are sloped to create a conical section when the fuselage is formed.” We see that RNoPE produces an exceptional output despite the repetition, in contrast to the other models. 63
- 6.5 A set of examples of the outputs of YaRN on the same prompt from TriviaQA with and without finetuning. The correct answer for the first question is any phrasing of “Dartmoor”, and for the second question is any phrasing of “the end”. 66
- 6.6 Graph of the change in log losses across the finetuning of different models. We plot loss on a logarithmic scale rather than a linear one to make it easier to visually distinguish between nearby loss values. Despite this, the RoPE-based methods are so similar that it remains difficult to differentiate among them. We order the legend by initial validation loss. 69

List of Tables

3.1	Table of the different context extension methods we evaluate, with respect to their purpose in our investigation.	35
4.1	Illustration of the differences in chat template formats between the base and chat versions of Llama2 models.	37
4.2	Table of custom transformer classes and functions, with descriptions of what they do and how they were generally altered.	38
4.3	Table of tasks statistics in LongBench. This table was adapted from Bai et al. (2024), with the Chinese and code-based tasks removed.	40
5.1	Table of results across different models without any additional finetuning performed. The best-performing model for each dataset has its performance highlighted in bold. Our novel model architectures have their labels written in bold.	47
5.2	Table of results across different models with 3200 iterations of finetuning at batch size 10. The best-performing model for each dataset has its performance highlighted in bold. Our novel model architectures have their labels written in bold.	52
5.3	Table of micro-perplexities across different models on a 1000-entry held-out subset of the CommonPile filtered PG-19 dataset, for both non-finetuned and 32k-finetuned models. The lowest perplexity model in each category has its performance highlighted in bold (ignoring Base-11). Our novel model architectures have their labels written in bold. Note that a lower perplexity is better, as it indicates lower uncertainty in the model.	53
6.1	Table of improvements in LongBench results after 3200 iterations of finetuning at batch size 10 of different models. The context extension methods with the most improvement across each dataset has its performance highlighted in bold. Our novel models architectures have their labels written in bold.	65

- 6.2 Table of tasks statistics across the English LongBench tasks. 72
- 6.3 Table of correlation coefficients between each model’s micro-perplexity on a held-out validation set of the CommonPile filtered PG-19 dataset, and each LongBench task across all tested methods. The highest correlation models in each category has their performances written in bold. Our novel model architectures have their labels written in bold. Since performance in theory increases as perplexity decreases, all correlation coefficients should be negative. 75

CHAPTER 1

Introduction

Natural language is an irreplaceable component of our lives. It allows us to communicate our thoughts, ideas, and feelings in a way that can be understood by anyone who has learned the same language. Interpreting and manipulating language ultimately depends on understanding the rules behind it. Just as humans gradually learn these rules, the field of Natural Language Processing (NLP) studies how computers can model, process and generate language. From the perspective of a computer, communication can be conceptualised as an iterative process of generating one word at a time until a complete sentence is formed.

A language model is fundamentally a calculator that, given a body of text, predicts the next word in a sentence. Each word is broken into a series of one or more tokens to improve the language model's capability of understanding language. These tokens define the vocabulary of the language model, and allow it to decompose more complex or rarer words into smaller pieces of information. Each choice of the next token depends on the previous token in the sentence, the context of the sentence, and the rules of the language. However, a language model must first learn how these factors influence the next token to be generated. This next-token prediction process is learned by training the model on example inputs, which involves iteratively updating the model's parameters used to compute the next generated token. Furthermore, the model must also be able to interpret tokens in a way that enables calculations to be performed. Tokens can be represented as higher-dimensional vectors known as input embeddings that capture the meanings, syntax and morphology of the words, punctuation, or sub-words they represent. However, these static input embeddings do not account for the influence of a word's position and context in a body of text. When rearranging the words in a sentence, the meaning and coherence of the sentence can drastically change, while the input embeddings remain the same. Similarly, the current context of a body of text can also impact the interpretation of a sentence. Thus,

various strategies have been developed to allow language models to incorporate the positional and contextual significance of words in their calculations.

However, there is an underlying principle that a trained model only generalises well on examples within the distribution of text that it has seen in training (Valiant, 1984). For example, a common word such as “good” would be seen by the model significantly more often than an obscure word like “absquatulate”. Just like a human, a language model with limited training would have a much poorer understanding of the latter word and thus perform worse on inputs containing it. In a similar way, language models are only trained on inputs up to a certain length. This is primarily due to the increased inference complexity and hardware constraints of modern language models, and the lower availability of high-quality long-length input training data. As a result, language models perform significantly worse on examples with inputs longer than those seen in training. A straightforward solution is to truncate the context so that the new input is more likely to fall in the distribution of seen examples. However, this truncation risks the loss of vital contextual information for next token prediction and related tasks. This problem motivates the process of context extension, which we define as follows: The extrapolation of a model from its original maximum training context size to a new larger context size. Figure 1.1 illustrates an example in which context extension would be beneficial, using an unrealistically small training context window size of $L = 4$ words for the sake of illustration. In the old context window, the context window contains “was painted by”, indicating that the answer could be any painter. However, when extending the context window to size $L' = 8$ words, we clearly see that the next token should either be “Leonardo” or “da” since the painter is Leonardo da Vinci. With the extended context, the language model has the information required to make a more accurate choice of the next token generated.

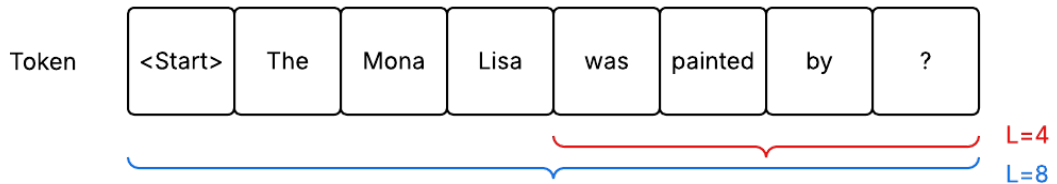


FIGURE 1.1: An illustration of a next-token prediction task for the “?” token. The red braces represent the span of the old context window of size $L = 4$ words, whereas the blue braces represent the span of the new context window of size $L' = 8$ words.

There are numerous existing strategies for context extension, but none of which produce exemplary results on long context benchmarks. In addition, many existing strategies for context extension require significant additional training. After extending the context window of a model, it is common to train it further on additional training examples. This process allows the model to adapt to changes made to its architecture and is known as finetuning. Modern large language models (LLMs) used predominantly for NLP tasks such as next-token prediction adopt the transformer architecture (Vaswani et al., 2017). The computational requirements of training and finetuning these LLMs have become increasingly costly due to the sheer number of parameters they contain, making large-scale training accessible only to leading AI research companies such as OpenAI, Microsoft and Google. Finetuning a model is also much more demanding than prompting a model due to the need to store the gradients and optimiser states used in backpropagation, potentially quadrupling GPU memory requirements without any optimisations. These challenges make context extension methods that require minimal finetuning more practical, as they reduce the cost of purchasing or loaning higher-VRAM GPUs that are necessary for finetuning. For even greater practicality, a context extension method that requires no finetuning to perform well will entirely remove the need for any additional training. These no-finetuning methods are commonly referred to as “plug-and-play” (Jin et al., 2024). The challenges in context extension we have outlined motivate our two primary research questions:

Research Question 1: Can we design effective context extension methods that require minimal or no additional finetuning?

Research Question 2: To what extent does minimal finetuning improve the effectiveness of different context extension methods in comparison to no finetuning?

Additionally, we consider whether specific long context tasks are more challenging than others. Understanding the weaknesses of different context extension methods may allow future work to address these weaknesses. It is also instructive to see whether different tasks are more or less difficult across all context extension methods or whether task difficulty varies with the method used. These ideas motivate our third research question:

Research Question 3: How does the difficulty and validity of different long-context tasks vary between context extension methods?

In this thesis, we explore novel methods of context extension, alongside alterations of existing strategies of context extension. Our primary contribution is the development of fractional RoPE, a variant of RoPE that satisfies a collection of desirable properties given some assumptions. The objective of fractional RoPE is to behave similarly to regular (base) RoPE for minor positional differences between tokens. This design decision means that the model’s attention computations for nearby tokens will change little after context extension, retaining the model’s understanding of the relationships between nearby tokens and hence minimising the drop in performance without further finetuning. Fractional RoPE also adheres to other desirable properties that we outline in Chapter 3.

Our other main contribution is the development of various hybrid approaches that combine the RoPE, NoPE and ALiBi positional embedding methods. Previous work shows potential in combining multiple positional embedding types or attention mechanisms, and we aim to assess the performance of these existing and novel hybrid methods in context extension. Specifically, we explore hybrid methods that alternate between different attention mechanisms across consecutive layers, and combine multiple context extension methods in a single layer where possible. Unlike fractional RoPE, which we specifically designed to require minimal finetuning, we believe that these hybrid methods are less likely to perform well with minimal or no finetuning. We implemented our designs on the Llama2-7-billion parameter model (Llama2-7b) (Touvron et al., 2023), which is open source and widely used in other context extension investigations. We primarily evaluate our methods on the Longbench (Bai et al., 2024) long-context benchmark, which comprises of 14 different english tasks and has been widely used in prior context-extension research. Our evaluation also considers the perplexity of the model on a held out portion of the finetuning set, and manual analysis of various outputs. These design choices are motivated by our research questions.

In Chapter 2, we critically analyse the literature required to understand context extension, existing approaches, and relevant datasets and benchmarks. In Chapter 3, we define our context extension methods and explore the methodology and intuition behind our proposed implementations. In Chapter 4, we outline and justify our experimental method. In Chapter 5, we present our experimental results. Finally, in Chapter 6, we discuss the implications and rationale behind our results in reference to our research questions, while outlining any limitations and future work, before concluding our work in Chapter 7.

Literature Review

This chapter covers the foundations of large language models (LLMs) and the background knowledge necessary to formally define and understand the challenges of context extension. It also critically evaluates existing approaches to solving the context extension problem, highlighting their strengths and weaknesses to inform the design of our own methods. Finally, it assesses the suitability of various candidate datasets for finetuning and benchmarks for evaluating model performance, and discusses their strengths and weaknesses for our evaluation methods.

2.1 Early Work

Modern large language models (LLMs) are built from the transformer architecture introduced by Vaswani et al. (2017). First, we explore the main predecessor of the transformer to understand why the transformer has been adopted as the dominant LLM and the RNN’s weaknesses in handling long-context tasks. Before the introduction of transformers, variants of recurrent neural networks (RNNs) (Rumelhart et al., 1986) such as Long Short-Term Memory (LSTMs) and Gated Recurrent Units (GRUs) were the state-of-the-art choice for language models. These RNN-based approaches are made up of one or more of two fundamental components known as the encoder and as the decoder. For example, in next-token prediction tasks, the Seq2Seq RNN has an encoder that is responsible for understanding the inputs, and a decoder component responsible for predicting the next tokens. When this model receives a body of text as input, the encoder initialises a vector known as the hidden state to a default value h_0 . The hidden state at time step i denoted h_i is a condensed representation of the meaning of the first i tokens in the sentence. The encoder then recursively computes the next hidden state h_{i+1} from the previous hidden state h_i and the current input embedding x_{i+1} via the mapping in Equation 2.1, which uses learned matrices. If the input has n tokens, then the hidden output of the

encoder h_n is a condensed representation of the entire sentence. The decoder then takes this final hidden state $h'_0 := h_n$ and computes the output embeddings via another mapping with learned matrices illustrated in Equations 2.3 and 2.4. These output embeddings are recursively computed and mapped back into tokens to generate the text output. The output embedding y_i from the previous computation becomes the input embedding x_i for the next computation. This recursive generation terminates when either a maximum generation limit is hit or a stop token is generated.

$$h_{i+1} = \tanh(b + Wh_i + Ux_{i+1}) \quad (2.1)$$

$$y_i = c + Vh_t \quad (2.2)$$

$$h'_{i+1} = \tanh(b' + W'h_i + U'x_{i+1}) \quad (2.3)$$

$$y'_i = c' + V'h'_t \quad (2.4)$$

However, RNNs and their variants are fundamentally limited by their representation of the previous inputs as a single hidden state or set of states. In addition, tokens earlier in the prompt are less likely to be encapsulated by the hidden state, since their influence decays over the recursive computation. This results in poor performance on context-sensitive tasks in particular, and is worsened as input sizes grow. These challenges inspired the mechanism of attention, that introduces learned connections between all of the encoder outputs computed by Equation 2.2 and the decoder outputs. The mechanism of attention will be further discussed in detail for the transformer architecture. Despite attention overcoming a key limitation of the RNN for performing long-context tasks, RNNs and their variants face another key limitation which their difficulty with being parallelised. Each computation of a hidden state and an output relies on previous hidden state computations, forming a chain of dependencies of linear size that ultimately slows down training and inference.

2.1.1 Transformer Models

These limitations led to the development of the transformer architecture, which is the predominant LLM used today. Unlike RNNs and their variants, transformers are composed of stacked layers of parallel linear computations that take the entire input text as input. In order to account

for the contribution of nearby tokens, Vaswani et al. (2017) defines an attention mechanism to compute the importance of a given “query” token in relation to each of the “key” tokens in the text. The importance of each query token for a key token is defined as the “value”. In practice, the corresponding query, key and value matrices are computed from the matrix of input embeddings X_i using the learned matrices W outlined in Equation 2.5. These key-value pairs have the dot product applied to them and are soft-maxed to create a probability distribution, as seen in Equation 2.6, before the final attention values are computed.

$$Q = W_q X_i, \quad K = W_k X_i, \quad V = W_v X_i \quad (2.5)$$

$$\text{Attention}(Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d_k}} \right) V \quad (2.6)$$

Vaswani et al. (2017) also defines positional embeddings to represent the significance of a token’s position in input text. Positional embeddings have the same dimensions as input embeddings and are summed or concatenated to produce the output embeddings used by the model. The paper proposes using sinusoidally oscillating vectors as position embeddings as described in Equation 2.7. For the d -th dimension in the embedding, the frequencies of oscillations are dependent on the token’s absolute position j , a constant base $b = 10000$ and the embedding dimension D . The paper also explores the use of learned absolute position embeddings. After computing X_i as the sums of the input and position embeddings, the model computes the attention values as shown in Equation 2.6 before determining the output of the attention layer. In a transformer model, many attention layers are stacked on top of each other, separated by normalisation and feedforward layers. In addition, there are multiple attention heads that perform their own attention calculations in parallel within the attention layer. Transformers can also be built with encoder and/or decoder components, similarly to an RNN. For these reasons, we only consider performing context extension on transformer models in our exploration.

$$\theta_d = 1/b^{2d/D} \quad p_{j,2d} = \sin(j\theta_d) \quad p_{j,2d+1} = \cos(j\theta_d) \quad (2.7)$$

2.1.2 The Context Extension Problem

The results of Vaswani et al. (2017) on the Bilingual Evaluation Understudy (BLEU) metric (Papineni et al., 2002) revealed that the proposed transformer architecture outperformed existing architectures in language translation tasks, while having much lower training and inference

costs. These results led to transformers becoming widely adopted for language model design. The implementation of attention also improved the ability of transformer models to perform well on larger context tasks. However, the design of position embeddings limits the ability of the transformer model to perform well on larger inputs. This is because positional embeddings corresponding to larger input token indices are seen significantly less frequently than those corresponding to smaller indices in the training process.

In practice, a transformer model is trained only on examples with a maximum of L tokens. Here, L is defined as the training context window size and corresponds to the number of positional embeddings the model sees during training. Suppose an input of size $L' > L$ is given to the model, where L' is defined as the extended context window size. For these input examples, model performance degrades drastically because the position embeddings of the last $L' - L$ tokens have never been seen during training. This idea is corroborated by Peng et al. (2024) in Figure 2.1. Here, perplexity can be conceptualised as the uncertainty of an LLM in its next-token predictions on some dataset, and thus smaller perplexities are better. The figure demonstrates how all models eventually increase in perplexity as context window size increases. For example, both YaRN-Llama2-13b-64k and YaRN-Llama2-7b-64k exhibit decreasing perplexities before reaching the training context window size of 64000. Once the input’s size exceeds the training context window size, both models experience a substantial increase in perplexity on the Proof-Pile dataset, indicating a degradation of the model’s output quality.

We define position ids that are outside of the training range as being out of distribution (OOD). In general, we define $s := L'/L$ as the ratio between the testing and training context lengths. The challenge of dealing with OOD position ids is defined as the context extension problem and has become an important subject of research within the NLP community.

2.2 Positional Approaches

Various general approaches to the context extension problem have been designed and tested. These approaches avoid the computational cost of performing further finetuning and instead involve redefining the values of position ids. In general, position-based techniques aim to map the original position id i in the extended context window of size L' to a new position id j in the training context window of size L . By mapping position ids into the training context window

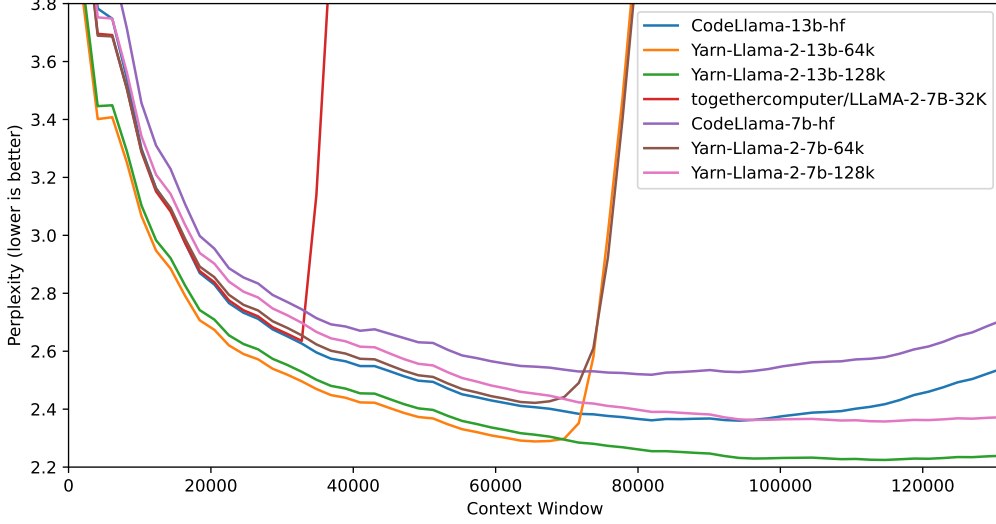


FIGURE 2.1: Comparison of how perplexity on the Proof-Pile dataset changes as context window increases from Peng et al. (2024). At a large enough context window size, the perplexity for every model will begin to increase.

range, all positional embeddings are forced to fall in the range of those seen by the model in training. In theory, this improves the model’s output quality, as it will perform better on inputs similar to those in the training data distribution.

2.2.1 Positional Reorganisation

Positional reorganisation is a simple approach that defines a discrete map from a position id in the extended range $i \in [0, L')$ to a non-negative integer position id in training range $j \in [0, L)$. Jin et al. (2024) introduces Self-Extend as a natural implementation of this idea. They define the transformation from index i to index j as $j := \text{floor}(i/G)$, where G is a constant for the group size. This approach clusters groups of G neighbouring tokens into sets with the same positional embeddings. By setting $G \geq L'/L = s$, no positions will be OOD. This is because the largest position id $i = L' - 1$ is mapped to $j = \text{floor}((L' - 1)/G) \leq \text{floor}((L' - 1)L/L') = L - 1$. The main strength of this idea is its simple, computationally effective approach of handling OOD positions. However, the LLM can no longer recognise the positional differences of tokens in the same group. Likewise, the LLM underestimates the distance between two tokens by a factor of $1/G$. In our own exploration of context extension methods, we consider ways to reduce the impact of scaling on the accuracy of the position embeddings. Despite these theoretical

issues, the paper demonstrates that Self-Extend improves results on the LongBench benchmark compared to existing context extension approaches (the benchmarks that we consider are further discussed in Section 2.6).

Another simple positional reorganisation approach proposed by Ratner et al. (2023) is Parallel Context Windows (PCW). PCW divides context into windows of length N . Cross-token attention is then restricted to occur only within each window. This means if $N < L$, then OOD positions will not exist. Unlike Self-Extend, PCW does not rescale the positional differences between tokens, resulting in the local representation of position becoming more accurate. However, PCW’s separation of context results in a loss of information transfer between windows. This is especially an issue at the cutoff points of the window, where adjacent tokens may not lie within the same window. Despite this limitation, PCW performs significantly better than the base in-context learning approaches across the majority of 15 different datasets, with an increasing performance gap as model sizes increase. In our own work, we consider ways to avoid harsh barriers between nearby tokens, such as cutoff points.

An et al. (2024) attempt to address the limitations of PCW by proposing dual chunk attention (DCA). This method redefines the technique introduced by Ratner et al. (2023) as intra-chunk attention, since PCW only uses attention between tokens within each chunk. In addition, DCA defines inter-chunk attention by assigning position ids to tokens in other chunks. These ids are chosen to be larger than those in the same chunk to maintain the significance of the difference in positions. DCA also deals with the special case where the nearby tokens are cut off between adjacent chunks. They define this exception as successive-chunk attention, with its application demonstrated in Figure 2.2. This method boasts better performance in comparison to other context extension approaches and rotary approaches across various context window sizes in perplexity on the PG-19 dataset. However, DCA has the major limitation of high computational complexity in inference because it uses the positions of the entire input, whereas PCW computes each window individually. Our own work will consider the tradeoff in performance and computation time for context-extension.

2.2.2 Positional Interpolation

Unlike positional reorganisation, which defines a discrete map between non-negative integers i and j , positional interpolation defines a continuous map from i to non-negative real j . For these

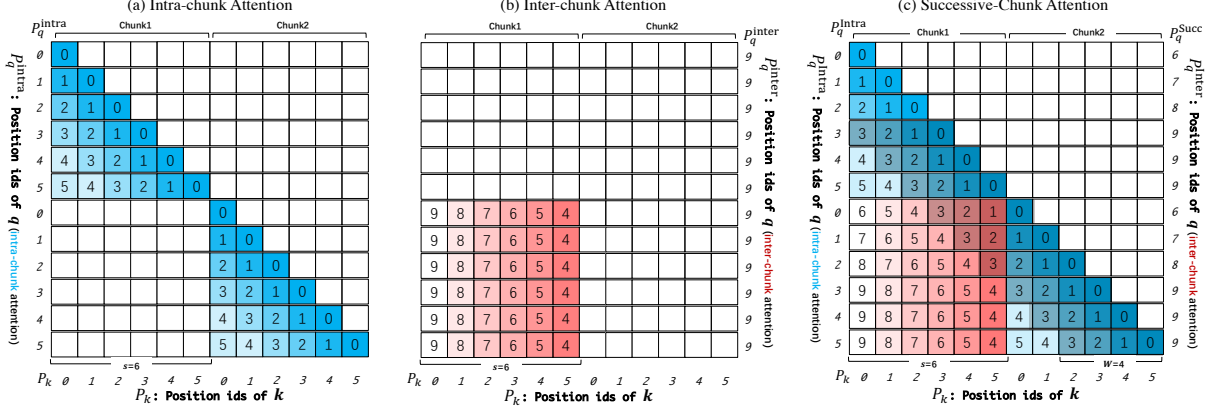


FIGURE 2.2: An illustration of how DCA combines intra-chunk, inter-chunk and successive-chunk attention from An et al. (2024). Note that intra-chunk attention is identical to PCW from Ratner et al. (2023).

approaches, the map between the position id j to the corresponding position embedding must be a continuous map. The original mapping proposed by Vaswani et al. (2017) in Equation 2.8 satisfies this requirement and thus is compatible with positional interpolation. Chen et al. (2023) introduce the simplest approach to positional interpolation as the map $j := i/s$, where s is the ratio L'/L . Intuitively, the approach linearly squeezes the distribution of position ids of the extended window of size L' into the training window of size L . This idea can be applied to both non-rotary and rotary positional embeddings, but has only been extensively studied in the rotary case. Chen et al. (2023) demonstrate that positional interpolation reduces the perplexity of LLMs on the PG19 benchmark and maintains its performance on larger context windows. Our own work considers the benefits of a continuous position id to positional embedding map, to ensure that nearby tokens are always distinct.

$$\theta_d = 1/b^{2d/D} \quad p_{j,2d} = \sin(j\theta_d) \quad p_{j,2d+1} = \cos(j\theta_d) \quad (2.8)$$

2.3 Rotary Approaches

The implementations of positional embeddings discussed so far involve translations of the input embeddings by adding a positional embedding. This means that positional information is entangled with semantic information, as they occupy the same dimensions in the embedding space without any additional restrictions. Although adding the positional and semantic components is easier to implement, ideally the two components would be separable. This problem

inspired rotary embeddings, which decouple the semantic and positional information as pairs of magnitudes and angles.

2.3.1 Rotary Positional Embeddings

Su et al. (2023) propose rotary positional embeddings (RoPE), which model positional embeddings as rotations of the input embedding in complex space. Given the query embedding x_q at position m and the key embedding x_k at position n , Equations 3.1, 3.2, and 3.3, illustrate how RoPE redefines the query vector x_q , key vector x_k and dot product attention calculations for a single complex dimension. In implementation, these equations are translated into a real coordinate space with twice the dimension size, and the computation is performed on the matrix of all input embeddings X to produce query and key matrices Q and K respectively. These equations use Θ as the diagonal matrix of θ defined in Equation 2.7 across each dimension d and use $*$ to denote the conjugate transpose of a matrix.

$$x_q^m = (W_q x_m) e^{im\theta} \quad (2.9)$$

$$x_k^n = (W_q x_n) e^{in\theta} \quad (2.10)$$

$$\langle x_q^m, x_k^n \rangle = \Re \left((W_q x_m) (W_q x_n)^* e^{i(m-n)\Theta} \right) \quad (2.11)$$

RoPE demonstrates improvements over the aforementioned non-rotary context extension methods on the CAIL2019-SCM benchmark (Xiao et al., 2019), while also enabling more efficient context extension methods. However, further literature explores the limitations of the basic RoPE method. Liu et al. (2023) proves that there exists a critical input embedding dimension d_{extra} such that for $d > d_{extra}$, the periods of the sinusoidal oscillations seen in the extended context exceed the periods seen with the training context window. More intuitively, for $d > d_{extra}$, the sine and cosine functions used in RoPE are unable to cycle through the entire range of input angles from 0 to 2π , and thus, there exist angles that have never been seen in training. This d_{extra} is defined in Equation 2.12, where D is the size of the self-attention head, T_{train} is the training period, and $b = 10000$ is the base. For such dimensions $d > d_{extra}$, the corresponding angles have not been observed in training and are referred to as OOD, resulting in a decline in performance. This paper also notes the importance of the base used for mapping dimensions to angles. Vaswani et al. (2017) and Su et al. (2023) both use a base of 10000. Equation 2.12

demonstrates that lower bases increase the critical dimension and thus improve the extrapolation performance. However, decreasing the base will result in a decrease in interpolation performance, resulting in an accuracy tradeoff. In our work, we consider the scaling laws derived in this paper and the existence of a critical dimension when exploring scalable positional embeddings methods based on RoPE. Specifically, we consider ways to avoid the OOD issue.

$$d_{extra} = 2 \left\lceil \frac{D}{2} \log_b \frac{T_{train}}{2\pi} \right\rceil \quad (2.12)$$

2.3.2 NTK-Aware RoPE

Since RoPE involves the interpolation of sinusoidal waves with different frequencies, it can be considered as a 1-dimensional case of a Fourier feature (Peng et al., 2024). In general machine learning, Neural Tangent Kernel (NTK) theory describes how models evolve over the course of training. A blog post by bloc97 (2023) proposes applying NTK theory to RoPE, claiming that the classical linear interpolation of RoPE’s Fourier space is not optimal, as it hinders the network’s ability to distinguish between nearby tokens. The author suggests a redefinition of the angle calculation for each dimension that effectively results in a non-linear interpolation. The previous angle definition proposed by Su et al. (2023) in terms of the position id j , base b , and d -th dimension of the D -dimensional embedding is outlined in Equation 2.13. The NTK-aware approach redefines the base as b' as shown in Equation 2.14, replacing the b term in the previous equation. When implemented on LLaMa-7b, the method demonstrated a significant decrease in perplexity compared to previous RoPE interpolation methods on a set of prompts with more than 12000 context size. Thus, this implementation resulted in a significant improvement in context extension methods. This research’s application of NTK theory has motivated many other papers to explore non-linear interpolation methods and is highly relevant in today’s research in context extension for rotary positional embeddings.

$$\theta_d = \frac{j}{b^{2d/D}} \quad (2.13)$$

$$b' = s^{\frac{D}{D-2}} \quad (2.14)$$

2.3.3 Yet another Rope Extension

Extending this NTK-aware approach, Peng et al. (2024) propose Yet another Rope Extension (YaRN) to address the critical dimension issue identified by Liu et al. (2023). The paper claims that the base NTK-aware approach struggles to distinguish between the positions of nearby tokens. This is because rescaling the base b reduces the angle between two nearby tokens. To remedy this, YaRN implements a piecewise approach to maintain the benefits of the NTK-aware approach while maintaining the distinction between nearby tokens. The model proposes a continuously differentiable piecewise function for computing the new angle $\hat{\theta}_d$. In Equation 2.15, $r(d)$ refers to the ratio between the original context size and the wavelength of the dimension's oscillations. The parameters α and β in Equation 2.16 set the cutoff points for using no interpolation, full interpolation, and a weighted combination of both. Equation 2.17 then computes the new angle as a weighted linear combination of the interpolated and non-interpolated angles. In addition to these changes, YaRN also adds a temperature constant in the denominator of the softmax calculation for attention scores. Overall, YaRN demonstrates an improvement from existing RoPE methods on large context size methods with easy implementation. In our own research, we consider whether piecewise methods are worth considering over other ideas.

$$r(d) := \frac{L}{\lambda_d} = \frac{L}{2\pi\theta_d} = \frac{L}{2\pi b^{\frac{2d}{|D|}}} \quad (2.15)$$

$$\gamma(r) := \begin{cases} 0, & r < \alpha \\ 1, & r > \beta \\ \frac{r-\alpha}{r-\beta}, & \alpha \leq r \leq \beta \end{cases} \quad (2.16)$$

$$\hat{\theta}_d := (1 - \gamma(r(d))) \frac{\theta_d}{s} + \gamma(r(d)) \theta_d \quad (2.17)$$

2.3.4 Resonance RoPE

Unlike NTK-aware RoPE and YaRN, Wang et al. (2024b) propose an entirely different method to improve interpolation in pre-critical dimensions. The main issue with traditional interpolation is that the interpolated angles may not align with the angles seen in training. This would result in poorer performance, as the model has not been trained to handle these angles. Wang et al. (2024b) propose Resonance RoPE (ReRoPE), which discretises the angles used across all

dimensions. By mapping angles to the nearest discretised angle, the angles beyond the training context window are mapped to angles that have been seen in training, resulting in a direct alignment between training and testing context windows. The effect of discretisation on the distribution of rotary angles is illustrated in Figure 2.3. One theoretical limitation of this approach is that nearby tokens may be rounded to the same angle. This approximates nearby tokens as having the same position, which may impact the method’s performance as the model cannot distinguish between adjacent tokens. Another limitation is that this method does not solve the post-critical dimension issue. When $d > d_{extra}$, the discretized angles that lie in the OOD range have never been seen in training. Thus, angles that fall in the OOD range outside of the training window will be mapped to a discrete angle that has never been seen in training. Alternatively, mapping the OOD angle to the nearest discrete angle seen in training will massively change the original angle, reducing the angle’s representation of the token’s position. When comparing the performance of Resonance RoPE to RoPE (and Resonance YaRN to YaRN), the paper found small performance increases across most benchmarks. The paper also introduces the POSGEN training framework for training RoPE and YaRN on recursive and chain-of-thought tasks. Our research considers the consequences of sacrificing local positional information such as when two tokens have their angles rounded to the same discretised angle. We also consider whether these consequences make the resolution of the OOD issue with a discretised approach worth it.

2.3.5 Interpolation from a Distributional Perspective

Wu et al. (2024) propose a piecewise approach that differs from YaRN. This method specifically aims to improve the alignment between the angles seen in training and testing by treating the probabilities of angles lying in certain sectors as a distribution. They claim that if you segment the domain of angles $\theta \in [0, 2\pi]$ into $b = 360$ intervals, then positional representation is best when, in all dimensions, the proportion of rotary angles for window size L' matches the proportion for window size L . The paper defines the term disturbance, equating it to the average of the KL-divergences of each dimension compared to the uniform probability distribution. Here, KL-divergence is a common measure of the difference between two probability density functions. By definition, minimising the disturbance maximises the uniformity of the distribution of rotary angles. In general, this makes disturbance a suitable objective function for optimisation. The formula for disturbance $\mathcal{D}(P_{L'}, P_L)$ in terms of the KL-divergences $\mathcal{D}^i(P_{L'}, P_L)$ of each dimension i is given in Equation 2.18. Here, $P_{L'}$ and P_L refer to the probability distributions for the

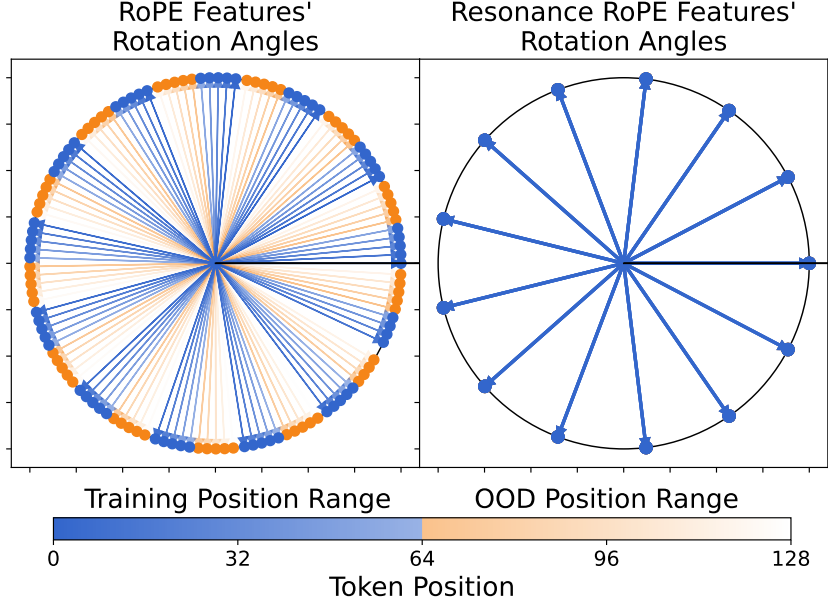


FIGURE 2.3: An illustration of how Resonance RoPE proposed by Wang et al. (2024b) influences the distribution of rotary angles across a single complex dimension.

extended context window and training context window respectively.

$$\mathcal{D}(P_{L'}, P_L) = 2 \sum_{i=0}^{D/2-1} \mathcal{D}^i(P_{L'}, P_L) / D \quad (2.18)$$

Their method then uses disturbance to decide whether to use positional interpolation or not for any given angle. If the disturbance of the interpolated distribution is less than the disturbance of directly extrapolating plus some threshold, the method chooses to interpolate. Otherwise, it chooses to extrapolate. The paper found minor improvements across context extension benchmarks up to a percentage point on existing techniques such as positional interpolation RoPE and YaRN. Our research will consider other methods to minimise the disturbance instead of simply choosing the best of two options, or if there exist better measurements of alignment quality beyond disturbance. However, we also consider the tradeoff between alignment and the accuracy of the positional representation as exemplified by Resonance RoPE. This work also raises the question of whether the evenness of the probability distributions can be optimised through training.

2.4 Other Embedding Methods

There has been active research into other positional embedding methods for the context extension problem. Since many different strategies have been tested, this section will summarise the dominant ideas.

2.4.1 Attention with Linear Biases

Attention with linear biases (ALiBi) is a method proposed by Press et al. (2022) that does not implement positional embeddings. After computing the inner products between a query value and its key values in the attention mechanism, ALiBi instead adds a negative linear bias term to each query-key pair. This term is equal to the positional difference between the query and the key tokens. The softmax operation will consequently assign a stronger weighting to key tokens closer to the query token, thus prioritising local tokens. In theory, the attention scores of important connections between long-range pairs of tokens will be learned to be much greater to counteract the negative bias enforced by ALiBi. The new attention formula is stated in Equation 2.20, where m_d is an attention head specific constant. In the original paper, Press et al. (2022) define m_d as a geometric series and they empirically determine that Equation 2.19 provides the best scaling factor. This method demonstrates exceptionally low perplexity scores in comparison to RoPE and other methods. However, perplexity is not necessarily a reliable indicator of a model’s performance in downstream tasks. Fang et al. (2025) discovered that, in long-context tasks, perplexity does not pay sufficient attention to tokens that are critical to correctness of the model’s output. In our work, we consider whether the LongPPL benchmark designed by Fang et al. (2025) is worth using instead of perplexity, or whether other evaluation methods overcome this issue. We also consider the strengths and limitations of different benchmarks. Despite this, ALiBi is a promising non-RoPE method that we consider in our own work. A simple method based on ALiBi may perform just as well if not better than rotary methods that have received much more attention in literature that are experiencing declining performance gains.

$$m_d = 2^{-8d/D} \quad (2.19)$$

$$\text{Attention}(Q, K, V) = \text{Softmax} \left(\frac{QK^T}{\sqrt{d_k}} + m_d \cdot [-(i-1), \dots, -2, -1, 0] \right) V \quad (2.20)$$

2.4.2 In-context Autoencoder

Ge et al. (2024) suggest an entirely different approach to handling the context extension problem. Rather than accounting for position ids for each input embedding, an autoencoder is trained to transform the context into a condensed embedding vector. This training process is illustrated in Figure 2.4, where the autoencoder is trained to encode the original context into a smaller embedding before being decoded back into the original context vector. After training, the encoder component has its parameters fixed and is inserted into the LLM’s implementation. The main limitation of this work is that there is no comparison with existing context extension methods. As such, it is difficult to assess the effectiveness of this method. Although we will not build on this method in our own research, it is important to consider other unique approaches to the context extension problem.

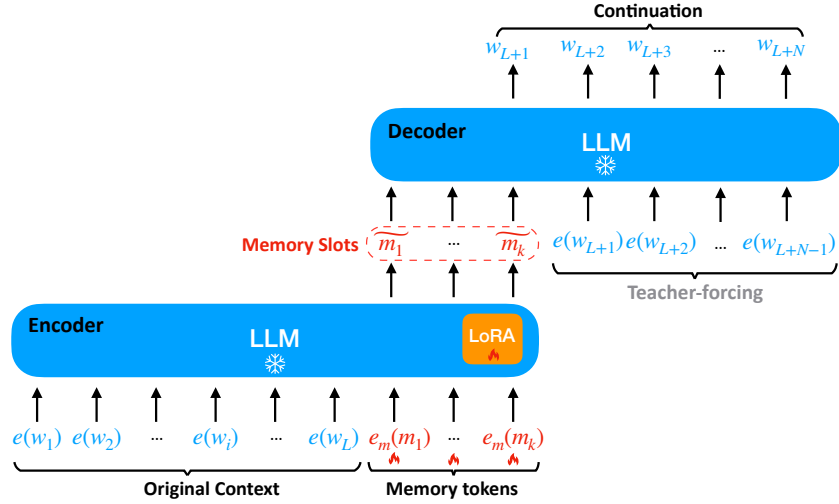


FIGURE 2.4: An illustration of the training process of the in-context autoencoder method designed by Ge et al. (2024).

2.4.3 No Positional Encodings

The No Positional Encodings (NoPE) method proposed by Kazemnejad et al. (2023) removes positional embedding completely. Their research proves that relative and absolute positional embeddings can be directly learned within the key, query and value matrices used by the attention mechanism. In practice, they claim that only relative positional embeddings are learned as only they are helpful. This study has the limitation of only considering decoder-only transformers. However, NoPE’s results on commonly used context extension benchmarks compared

to existing positional embeddings suggest that the current positional encoding methods provide limited benefit. In our research, we consider the advantages of different positional embedding approaches outside of the ability to represent position. However, we note that NoPE was evaluated by training RoPE and NoPE from scratch on the same dataset, unlike most context extension methods that instead finetune from the original RoPE model. This will be considered when we evaluate the performance of NoPE.

2.4.4 RoPE to NoPE and Back Again

Since the small performance gap between RoPE and NoPE was an interesting result, Yang et al. (2025) further explore the performance of both methods. Furthermore, they propose a combined method, RNoPE, that alternates between RoPE and NoPE embeddings between the attention head layers. Compared to the base RoPE model, RNoPE consistently achieved improvements of more than 2% on the RULER benchmark that provides synthetic question answering tasks. This performance gap extended to over 10% on context lengths of up to 256000. Across all tested benchmarks, the performance between RoPE and RNoPE is similar for lower context window sizes of less than 32 thousand but increases drastically with increased window size. These results reveal that hybrid attention methods can outperform existing methods on long contexts. The exceptional performance of hybrid methods compared to single-attention methods was also corroborated by Lenz et al. (2024) and Sun et al. (2024) in their respective architectures Jamba-1.5 and YOCO. The author of the paper admits that the reason for this is poorly understood. Regardless, we consider using hybrid positional embeddings in our own research and consider why hybrid methods perform so well.

2.5 Designing Better Context Extension Methods

Our literature review reveals that a broad range of context extension methods has been explored, yet all still have unique limitations and opportunities for improvement. In particular, many strategies expand on the base RoPE method introduced by Su et al. (2023), but after YaRN have only marginal improvements in results have been observed. One key assumption of these RoPE methods is that the RoPE angles have a linear dependence on the positional difference between two tokens. For example, doubling the relative distance between two tokens doubles the RoPE angles between them. However, there is no justification for this assumption. We

believe there is benefit in using non-linear RoPE angles, especially in context extension, which has motivated our design of fractional RoPE.

Furthermore, despite there being many unique RoPE based and non-RoPE based methods, hybrid methods of context extension have received very little exploration. RNoPE was the first paper to explore the ability of a hybrid method to perform context extension and simply implemented one such example. We believe that there is potential in hybrid methods of attention, and thus explore various ways to combine existing methods.

2.6 Benchmarks

Benchmarks are necessary to determine the relative effectiveness of different context extension methods. We will explore various long-context benchmarks in this section.

2.6.1 LongPerplexity

As previously outlined, perplexity is a measure of the uncertainty of an LLM in a prediction task on a given dataset. In alternative interpretation, lower perplexity suggests that the model has a higher average probability of predicting the tokens in a given text. Although perplexity is a standard performance metric for context extension tasks, it is not necessarily reliable, as Fang et al. (2025) outline. Hence, it may be worth considering the alternative evaluation metric LongPerplexity (LongPPL), proposing in the same paper. Fang et al. (2025) argue that specific key tokens in context are much more important than the others, and should be weighted higher when evaluating the model’s performance. However, the standard perplexity calculation assigns equal weightings to each token. LongPPL accounts for these key tokens in the perplexity calculation by providing them with a stronger weighting. However, this relies on the key tokens being known. Thus, LongPPL employs an oracle model to identify these key tokens. For this reason, we choose not to use LongPPL since ideally a strong model is required to accurately identify key tokens. Due to computational limitations, it is infeasible to deploy a larger model to perform this task. However, we still consider the limitations of perplexity described by Fang et al. (2025).

2.6.2 LongEmbed

Many previous papers also used a variety of downstream tasks to evaluate performance. Zhu et al. (2024) developed the LONGEMBED benchmark, combining a variety of synthetic and real-world tasks. Their paper discusses a variety of benchmarks commonly used in context extension evaluation and their limitations in assessing the problem. One such information retrieval benchmark is BEIR, developed by Thakur et al. (2021). This benchmark combines 18 different information retrieval tasks, where each task require an output of 300 tokens or less on average.

The two synthetic retrieval tasks featured in LONGEMBED are passkey retrieval and needle-in-a-haystack retrieval. Passkey retrieval involves an LLM retrieving a random passkey in a long document of garbage information, as proposed by Wang et al. (2024a). Unlike passkey retrieval, needle-in-a-haystack retrieval randomly inserts key information into a random position of an essay, as defined by Ivgi et al. (2023). Both of these methods test an LLM’s ability to extract a specific piece of information in a wall of unrelated text. However, they cannot assess an LLM’s ability to recognise long-range relationships between tokens in text and thus cannot be used on their own.

To overcome this limitation, LONGEMBED also outlines four real data tasks. These include NarrativeQA, a dataset for answering questions about long stories (Kočiský et al., 2018), and 2WikiMultiHopQA, a question answering dataset featuring questions of up to 5 hops (Ho et al., 2020). Here, hops refer to the required jumps between paragraphs or similar blocks of text to deduce the correct answer. The other two real-data benchmarks are QMSum, a dataset for summarising relevant segments of meetings in response to questions (Zhong et al., 2021), and SumScreenFD, a dataset for generating human-written summaries from television show scripts (Chen et al., 2022). Unlike the synthetic datasets, these real datasets require the LLM to link ideas from multiple locations in the document. Thus, LONGEMBED holistically assesses the performance of LLMs in long context tasks, making it an ideal benchmark for comparison. Zhu et al. (2024) also benchmark standard context-extension methods against it, such as SelfExtend, PCW, positional interpolation, base RoPE, and NTK-aware RoPE. These results maybe be helpful to reference when evaluating our own solutions.

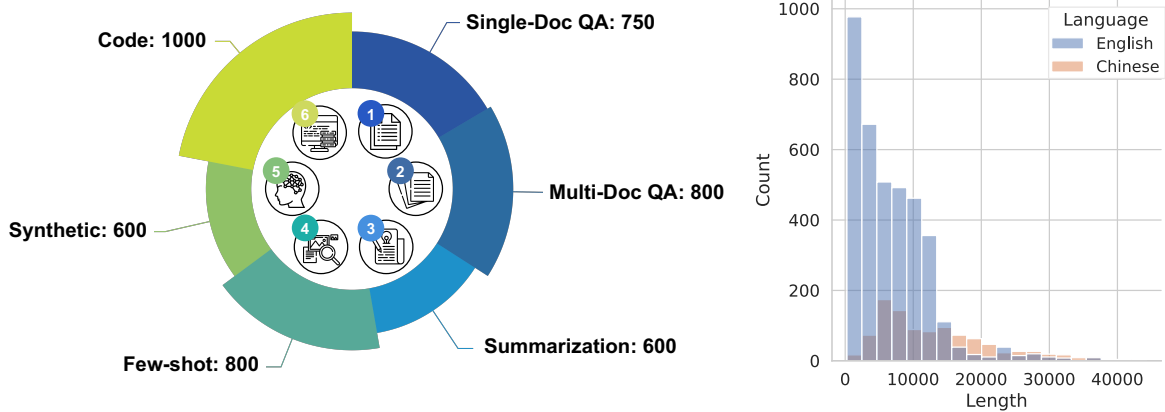


FIGURE 2.5: A summary of the distribution of different task types in the LongBench benchmark, alongside the distributions of the two different languages and their corresponding task token lengths.

2.6.3 LongBench

Another commonly used benchmark is LongBench. Bai et al. (2024) designs LongBench to cover 21 different datasets across 6 different tasks types in both English and Chinese languages. These tasks include question answering on single or multiple documents, and a variety of summarisation tasks, few-shot learning tasks, synthetic data tasks, and code generation tasks. Figure 2.5 provides an overview of the distribution of tasks in LongBench. All of these tasks require the LLM to understand the broader context of the input, thereby enabling LongBench to serve as a holistic and thorough benchmark. Hence, this benchmark is also worth considering for comparing various methods of context extension.

2.6.4 Other benchmarks and datasets

The context extension techniques mentioned in the previous sections have also been used on a wide variety of benchmarks. One of the more popular benchmarks is RULER (Hsieh et al., 2024), which combines 13 different synthetic long context tasks. Although synthetic tasks are easier to mass-produce and less likely to raise copyright issues, it is still valuable to consider real tasks as they are more likely to reflect real-world applications of context extension. Also, for the scope of our investigation, there are sufficiently many real-data long-context benchmarks.

There are also many other datasets useful for evaluating perplexity, such as The Pile (Gao et al., 2020), a 825GB English dataset containing webpages from 22 different subsets. Despite

The Pile being commonly used across many areas of NLP research, it has encountered various copyright issues due to the way the dataset was built. Instead, we consider the more modern CommonPile v0.1 dataset, which aims to provide data at the same quantity as the Pile while avoiding copyright violations. This dataset is currently composed of 8TB of textual data across 30 different data sources, providing more than enough training data for the scope of our research. The dataset is also filtered for quality and is divided into subsets based on the data source. One subset of interest is the CommonPile filtered PG-19 dataset (Kandpal et al., 2025), a set of 75k copyright expired books from Project Gutenberg (Project Gutenberg, 1971). By nature, books are long-context forms of text, making them ideal for training extended context language models. Outside of PG-19, there is a wide variety of suitable datasets that we can use to evaluate different context extension approaches. Due to computational constraints, we choose only to finetune on the PG-19 dataset and thus omit the discussion of other datasets.

Methods

In this chapter, we explore the various novel context extension methods we evaluate in our research. We also outline our method of comparing different context extension methods.

3.1 RoPE with non-linear positional dependencies

In all standard attention mechanisms, the inner product between the key and query vectors is responsible for determining the importance of the key token to the query token. In the base implementation of RoPE, this computation is illustrated in Equations 3.1, 3.2 and 3.3. The primary difference between the base attention implementation and the RoPE implementation is the introduction of the $e^{i(m-n)\Theta}$ term, where Θ is a $D \times D$ diagonal matrix of rotation angles. Here, $\Theta_{d,d}$ corresponds to the rotation angle of the d -th dimension, and D is defined as the hidden dimension size of the RoPE attention head. RoPE defines $\Theta_{j,k}$ in its complex representation as expressed in Equation 3.4.

$$x_q^m = (W_q x_m) e^{im\theta} \quad (3.1)$$

$$x_k^n = (W_q x_n) e^{in\theta} \quad (3.2)$$

$$\langle x_q^m, x_k^n \rangle = \Re \left((W_q x_m) (W_q x_n)^* e^{i(m-n)\Theta} \right) \quad (3.3)$$

$$\Theta_{i,j} = \begin{cases} 1/b^{2j/D} & \text{if } j = k \\ 0 & \text{if } j \neq k \end{cases} \quad (3.4)$$

To generalise the design of RoPE-based context extension methods, we define the mapping in Equations 3.5 and 3.6. Note that the value of Θ may depend on the values of m, n, L and L' without any additional restrictions. In the case of the default RoPE implementation,

$\Phi(m, n, L, L', \Theta) = (m - n)\Theta$ where Θ is as defined in Equation 3.4.

$$\Phi : \mathbb{Z} \times \mathbb{Z} \times \mathbb{Z}^+ \times \mathbb{Z}^+ \times \text{Diag}(\mathbb{R}, D) \rightarrow \text{Diag}(\mathbb{R}, D) \quad (3.5)$$

$$(m, n, L, L', \Theta_{L, L'}) \mapsto \Phi(m, n, L, L', \Theta) \quad (3.6)$$

In this section, we explore alternative definitions of Φ and motivate them with respect to the context extension problem.

3.1.1 Angle-position independent RoPE variants

There are infinite classes of functions that perform the mapping described by Equation 3.5. Thus, we consider the following restriction on Φ :

$$\Phi(m, n, L, L', \Theta) = f(m, n, L, L')\Theta(L, L')$$

Here, $\Theta(L, L')$ represents that the RoPE angle matrix only depends on the original and extended context window sizes L and L' . We define this as the angle-position independent assumption, since the computation of the RoPE angles is independent of the positional ids m and n . We choose this assumption for the following reasons: Firstly, it is clear that existing RoPE variants satisfy this assumption. For example, default RoPE sets $f(m, n, L, L') = m - n$, whereas linearly interpolated RoPE sets $f(m, n, L, L') = (m - n)L/L'$. Secondly, it significantly narrows the space of possible RoPE variants to consider making the scope of our research more realistic. Finally, the decoupling of the significance of position with the significance of dimension allows us to impose constraints on $f(m, n, L, L')$ that are ideal for context extension. We also assert an additional restriction:

$$f(m, n, L, L') = g(s, L, L'), \text{ where } s := m - n$$

This restriction enforces that significance of position is solely based on the relative position difference between two tokens. For example, the relationship between the tokens at the 4th and 7th positions is the same as that between the 123rd and 126th positions. Although absolute positions may carry some significance in the relationships between tokens, we choose to ignore this possibility to restrict our function space and help establish desirable properties for our function $g(s, L, L')$. In addition, many existing RoPE-based context extension methods including positional interpolation RoPE and YaRN adhere to these constraints. With this restriction, we

define new mappings in Equations 3.7 and 3.8.

$$\Psi : \mathbb{Z} \times \mathbb{Z}^+ \times \mathbb{Z}^+ \times \text{Diag}(\mathbb{R}, D) \rightarrow \text{Diag}(\mathbb{R}, D) \quad (3.7)$$

$$(s, \Theta) \mapsto \Psi(s, \Theta) \quad (3.8)$$

$\Psi : \mathbb{Z} \times \text{Diag}(\mathbb{R}, D) \rightarrow \text{Diag}(\mathbb{R}, D)$ as:

$$\Psi(s, L, L', \Theta) = g(s, L, L')\Theta$$

We aim to construct a context extension method that is effective with limited finetuning. To achieve this, we aim to make the smallest possible changes to the original method (defined by $g(s, L, L)$) so that the model's parameters are only slightly misaligned when applying $g(s, L, L')$. This way, the performance degradation under context extension without any finetuning is minimal, and can potentially be corrected by minimal finetuning. We define five constraints that aim to achieve this.

Constraint 1: No changes in behaviour without context extension

If the training context window size is equal to the extended context window size, the default RoPE or RoPE variant's attention computation should be performed. For all RoPE based attention computations considered in this thesis, this implies that:

$$g(s, L, L) = s$$

This property represents the base case, where no context extension is required and thus no changes to the model should be performed.

Constraint 2: Equality at small positional differences

We impose that for small $|s|$, the following constraint holds:

$$g(s, L, L') \approx g(s, L, L)$$

More precisely, we state that the Taylor polynomials of $g(s, L, L')$ and $g(s, L, L)$ about $s = 0$ to the first order are equivalent. Constraint 1 defines $g(s, L, L) := s$. Mathematically, this would imply the restrictions, where g' denotes the derivative of g with respect to s :

$$g(0, L, L') = 0$$

$$g'(0, L, L') = 1$$

which make up our key constraints. Intuitively, these constraints ensure that for small positional differences s , the attention computation is similar to that used in the original context window. We believe that this property will allow the model to compute attention between nearby tokens with minimal performance degradation since the computation is similar to that in the original context window.

Constraint 3: Monotonically increasing RoPE angles

We impose that all RoPE angles in Θ monotonically increase with respect to s within the range of possible relative distances. Mathematically, for all $s \in (0, L')$ and all $d \in [0, D)$:

$$\frac{d}{ds} [g(s, L, L')\Theta_{d,d}(L, L')] > 0$$

Since RoPE angles are always positively defined in all considered variants, this condition can be restated as:

$$\frac{d}{ds} g(s, L, L') > 0$$

This constraint enforces that tokens that are relatively farther apart will always have larger RoPE angles than those of closer pairs of tokens. Although this condition is not strictly necessary by itself, it is consistent with most RoPE variants (with some exceptions, such as resonance RoPE, that are not strictly increasing).

Constraint 4: Maximum RoPE angles are constant and independent of L'

We impose that the maximum angles in the original context window are equal to the maximum angles in the extended context window. Mathematically, this can be expressed as for all

$L' \geq L$:

$$g(L', L, L')\Theta(L, L') = g(L, L, L)\Theta(L, L)$$

Simplifying with constraint 1:

$$g(L', L, L')\Theta(L, L') = L\Theta(L, L')$$

In the special case that $\Theta(L, L) = \Theta(L, L')$:

$$g(L', L, L') = L$$

This constraint addresses the out of distribution (OOD) issue outlined in Chapter 2. When combined with constraints 1 and 3, all rope angles under context extension are bound by the range below for all $s \in [0, L']$, $L' \geq L$:

$$0 \leq g(s, L, L')\Theta_{d,d}(L, L') \leq g(L, L, L)\Theta_{d,d}(L, L)$$

In the special case that $\Theta(L, L) = \Theta(L, L')$:

$$0 \leq g(s, L, L') \leq g(L, L, L)$$

Thus, the bounds of the RoPE angles used for context extension are the same as those when the model was originally trained.

Constraint 5: Symmetry

We impose that $g(s, L, L')$ is either an odd or even function in terms of s . Mathematically, either one of the following conditions is satisfied:

$$\text{Odd property:} \quad g(s, L, L') = -g(-s, L, L')$$

$$\text{Even property:} \quad g(s, L, L') = g(-s, L, L')$$

Most existing RoPE variants follow the odd property. This odd property enforces that the RoPE angle between a query token and a key token with respective indices m and n , is the negation of the angle when the key and query tokens are swapped. In contrast, the even property assigns the same RoPE angle in both cases, only recognising the absolute position difference between

the tokens. We primarily consider the odd property as all covered RoPE methods use it. This property results in all the benefits of constraints 1 to 3 on $s \in [0, L)$, also being applicable on the range $s \in (-L, 0]$. The even property is interesting as it does not distinguish between the order of the tokens, and only the absolute relative position difference between them. It may be worth considering in future work.

3.1.2 Fractional RoPE

We propose a novel method of context extension on the base RoPE method that satisfies all five constraints under the assumptions. The corresponding function is stated in Equation 3.9.

$$g(s, L, L') := \frac{s}{(1 + \beta_{L,L'} |s|^\alpha)^{\frac{1}{\alpha}}} \quad (3.9)$$

Where $\alpha > 0$ is a hyperparameter, and $\beta_{L,L'}$ is defined as:

$$\beta_{L,L'} := L^{-\alpha} - L'^{-\alpha} \quad (3.10)$$

We prove that the function satisfies constraints 1, 2, 3, 4 and 5. Note that constraints 1, 3 and 4 are sensitive on the value of β . We will derive β when proving constraint 4, and use that derived β in the proofs of constraints 1 and 3.

PROOF OF CONSTRAINT 1. To be proven: $g(s, L, L) = s$.

$$\begin{aligned} g(s, L, L) &= \frac{s}{(1 + \beta_{L,L} |s|^\alpha)^{\frac{1}{\alpha}}} \\ &= \frac{s}{(1 + (L^{-\alpha} - L^{-\alpha}) |s|^\alpha)^{\frac{1}{\alpha}}} \\ &= \frac{s}{(1 + (0) |s|^\alpha)^{\frac{1}{\alpha}}} \\ &= \frac{s}{(1 + 0)^{\frac{1}{\alpha}}} \\ &= s \end{aligned}$$

□

PROOF OF CONSTRAINT 2. To be proven: $g(0, L, L') = 0$ and $g'(0, L, L') = 1$.

$$\begin{aligned}
g(0, L, L') &= \frac{0}{(1 + \beta_{L, L'} |0|^\alpha)^{\frac{1}{\alpha}}} \\
&= 0 \\
g'(s, L, L') &= \frac{1 \times (1 + \beta_{L, L'} |s|^\alpha)^{\frac{1}{\alpha}}}{(1 + \beta_{L, L'} |s|^\alpha)^{\frac{2}{\alpha}}} - \frac{s \times \frac{1}{\alpha} \beta_{L, L'} \alpha \frac{|s|}{s} |s|^{\alpha-1} (1 + \beta_{L, L'} |s|^\alpha)^{\frac{1}{\alpha}-1}}{(1 + \beta_{L, L'} |s|^\alpha)^{\frac{2}{\alpha}}} \\
&= (1 + \beta_{L, L'} |s|^\alpha)^{-\frac{1}{\alpha}} - \beta_{L, L'} |s|^\alpha (1 + \beta_{L, L'} |s|^\alpha)^{-\frac{1}{\alpha}-1} \\
&= ((1 + \beta_{L, L'} |s|^\alpha) - \beta_{L, L'} |s|^\alpha) (1 + \beta_{L, L'} |s|^\alpha)^{-\frac{1}{\alpha}-1} \\
&= (1 + \beta_{L, L'} |s|^\alpha)^{-\frac{1}{\alpha}-1} \\
g'(0, L, L') &= (1 + \beta_{L, L'} |0|^\alpha)^{-\frac{1}{\alpha}-1} \\
&= (1 + 0)^{-\frac{1}{\alpha}-1} \\
&= 1
\end{aligned}$$

□

PROOF OF CONSTRAINT 3. To be proven: For all $s \in [0, L)$, $g'(s, L, L') > 0$.

$$\begin{aligned}
L^{-\alpha} &\geq L'^{-\alpha} && \text{Since } L' \geq L \text{ and } \alpha > 0 \\
L^{-\alpha} - L'^{-\alpha} &\geq 0 \\
\beta_{L, L'} &\geq 0 && \text{By definition of } \beta_{L, L'} \\
g'(s, L, L') &= (1 + \beta_{L, L'} |s|^\alpha)^{-\frac{1}{\alpha}-1} \\
&\geq (1 + 0)^{-\frac{1}{\alpha}-1} && \text{Since } |s|^\alpha \geq 0 \text{ and } \beta_{L, L'} \geq 0 \\
&> 0
\end{aligned}$$

□

PROOF OF CONSTRAINT 4. To be proven: We can solve for $\beta_{L, L'} \in \mathbb{Z}$ as given in Equation 3.10 such that $g(L', L, L') = L$. We consider the special case here as for base RoPE, the

condition $\Theta(L, L) = \Theta(L, L')$ is satisfied.

$$\begin{aligned}
L &= g(L', L, L') \\
&= \frac{L'}{(1 + \beta_{L, L'} |L'|^\alpha)^{\frac{1}{\alpha}}} \\
(1 + \beta_{L, L'} (L')^\alpha)^{\frac{1}{\alpha}} &= \frac{L'}{L} \\
\beta_{L, L'} (L')^\alpha &= \left(\frac{L'}{L}\right)^\alpha - 1 \\
\beta_{L, L'} &= \left(\frac{1}{L}\right)^\alpha - \left(\frac{1}{L'}\right)^\alpha \\
&= L^{-\alpha} - L'^{-\alpha}
\end{aligned}$$

□

PROOF OF CONSTRAINT 5. To be proven: $-g(-s, L, L') = g(s, L, L')$ (the odd property).

$$\begin{aligned}
-g(-s, L, L') &= -\frac{-s}{(1 + \beta_{L, L'} |-s|^\alpha)^{\frac{1}{\alpha}}} \\
&= \frac{s}{(1 + \beta_{L, L'} |s|^\alpha)^{\frac{1}{\alpha}}} \\
&= g(s, L, L')
\end{aligned}$$

□

3.1.3 Further exploration of Fractional RoPE

As previously outlined, RoPE and fractional RoPE differ only in their definitions of $g(s, L, L')$. In Figure 3.1, we plot fractional RoPE for various values of α . Here, we set $L' = 2L = 8192$, as doubling the context extension window is easiest to visualise, and the choices of L and L' match the choices defined in our experiments. For the sake of exploration, we allow $\alpha \rightarrow 0^+$ and $\alpha \rightarrow \infty$. We specifically choose to plot the non-boundary $\alpha = 0.5, 1$ and 2 as blue, green and orange lines respectively. As these values of α have been proven to satisfy the constraints of the previous subsection, we can visualise the impact of these constraints on the graph.

Firstly, we see that for $\alpha \rightarrow \infty$, fractional RoPE behaves identically to no-interpolation RoPE for lengths $s \leq L$. Then, for lengths $L < s \leq L'$, the RoPE angle is bounded by the highest

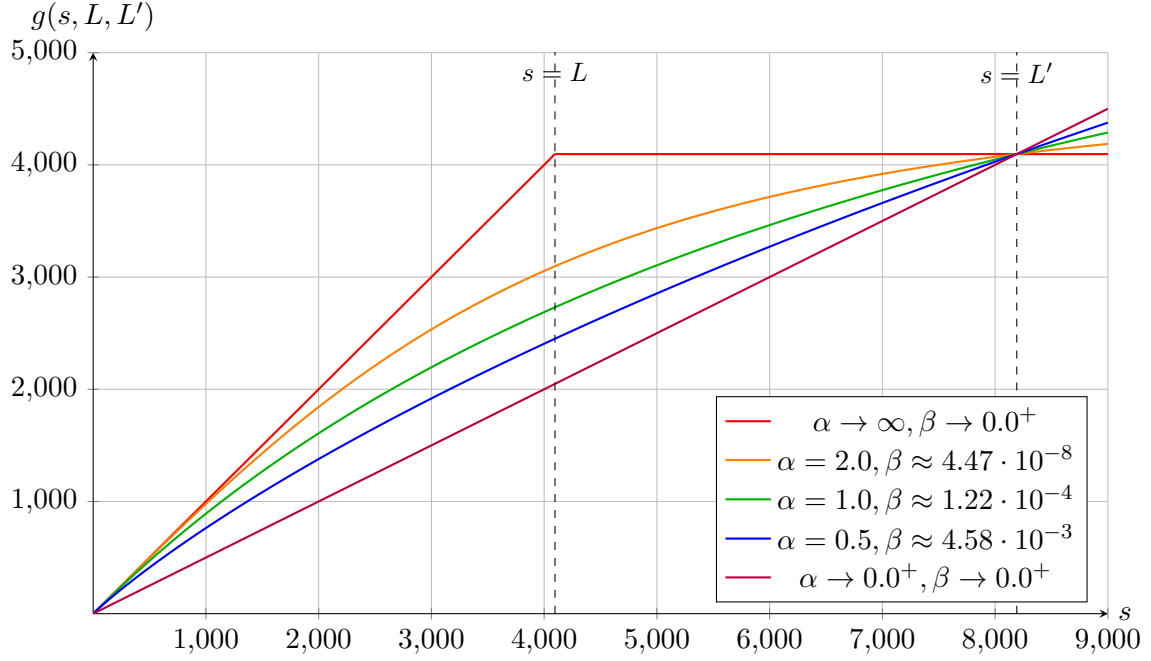


FIGURE 3.1: Fractional rope scaling under different α values for $L = 4096$ and $L' = 2L = 8192$. Each value of α has a distinct colour.

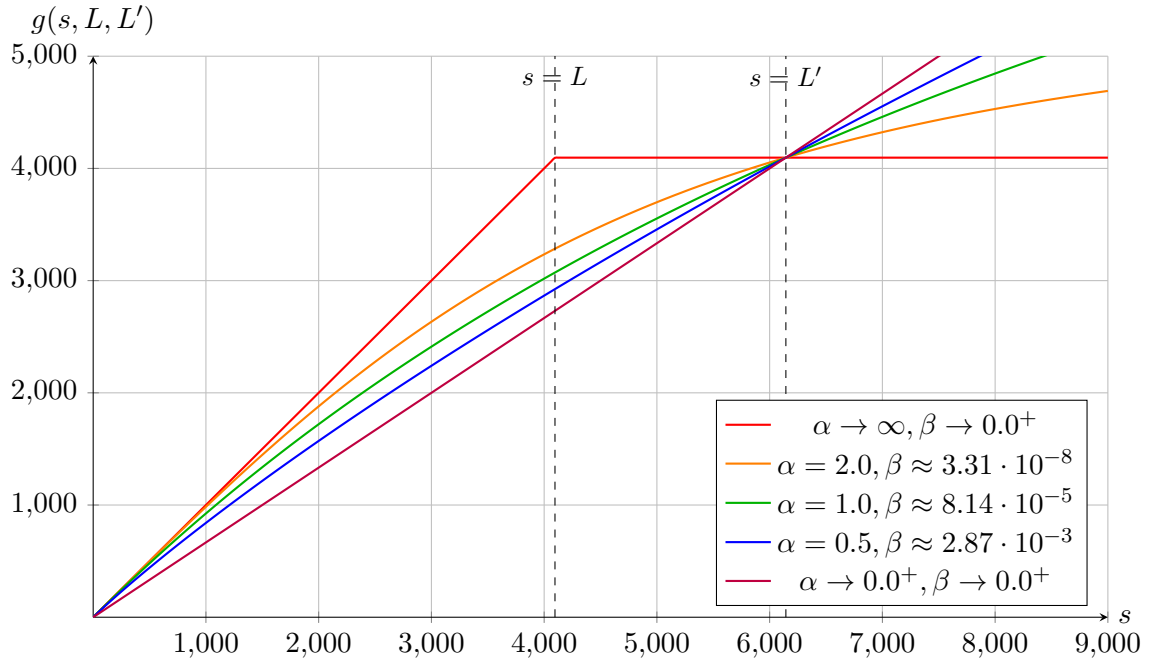


FIGURE 3.2: Fractional rope scaling under different α values for $L = 4096$ and $L' = 1.5L = 6144$. We see g_α squeezed closer together across all values of α .

RoPE angle seen in training at $L = 4096$. This means that setting $\alpha \rightarrow \infty$ results in the model no longer differentiating between positional differences greater than L , leading to all tokens outside of the original training window being treated the same. We can rewrite the $\alpha \rightarrow \infty$ case without limits as expressed in Equation 3.11. We will refer to this method as bounded no-interpolation RoPE. For the other boundary case where $\alpha \rightarrow 0^+$, fractional RoPE behaves identically to linear interpolation RoPE. We can see that constraint 2 (equality at small positional differences) does not hold, since the gradient of linear interpolation RoPE does match those of the original RoPE and other fractional RoPE methods. This means that linear interpolation RoPE’s angles deviate significantly more than fractional RoPE’s under context extension. Hence, its attention computations for nearby tokens are more dramatically altered, suggesting a larger degradation of performance in these examples without finetuning.

$$g_{\alpha \rightarrow \infty}(s, L, L') = \begin{cases} s, & |s| \in [0, L] \\ \text{sgn}(s)L, & |s| \in (L, \infty) \end{cases} \quad (3.11)$$

Fractional RoPE aims to balance the advantages and disadvantages of these two boundary methods. Consider the three selected α values within the valid domain $\alpha \in (0, \infty)$. We see that as we increase α , the function g_α is pushed closer to bounded no-interpolation RoPE for $s \leq L$. In contrast, decreasing α brings g_α closer to that of non-linear interpolation RoPE. We can easily see that constraints 3 (monotonically increasing) and 4 (upper bounded by L) are met, where all methods plotted intersect at (L, L') , meaning that all interpolated angles are within the training range. By reducing the value of L' to be closer to L , we can also see how constraint 1 (identity at $L = L'$) holds. In Figure 3.2, we set $L' = 1.5L = 6144$ instead. We can see how each g_α is squeezed tighter between bounded no-interpolation RoPE and linear interpolation RoPE in comparison to the larger $L' = 2L = 8192$ graph. In the limit case where $L = L'$, all g_α will converge to no-interpolation RoPE as required by constraint 1. Finally, constraint 5 (odd function) is not visible in the graphs due to the choice of domain $s \in [0, L']$. Rotating each g_α by 180 degrees about point $(0, 0)$ yields the g_α for $s \in [-L', 0]$. The clear consequence is that all favourable qualities of the first for constraints in the domain $s \in [0, L']$ now hold across the entire domain $s \in [-L, L]$ of possible s .

3.2 Hybrid methods

In Chapter 2, we briefly cover the hybrid position embedding method RNoPE, which alternates between RoPE and NoPE attention across consecutive transformer layers. The paper demonstrated that RNoPE performed strongly on long context tasks in comparison to RoPE and NoPE-only models. This raises the question of whether other hybrid methods are also ideal for context extension. We choose to investigate other potential combinations of different context extension methods. In particular, we aim to integrate RoPE with ALiBi. Unlike NoPE and RoPE, which cannot be used on the same layer as each other (since NoPE is the absence of positional embeddings by definition), ALiBi is completely compatible with RoPE on the same layer. In a similar way to RNoPE, we can also alternate between RoPE and other embedding methods such as ALiBi.

Due to budget limitations and computational constraints, we investigate only two novel forms of hybrid models involving ALiBi. Firstly, we investigate using both linear interpolation RoPE and ALiBi in every layer. Compared to using RoPE alone, adding ALiBi may force the model to strengthen attention scores between important pairs of tokens that are farther apart. This could result in the model masking insignificant long context relationships while still maintaining strong attention scores between important tokens. We name this method of context extension as Hybrid Positional Embeddings, which we abbreviate to HyPE. Secondly, we investigate alternating between ALiBi and linear interpolation RoPE in a way that imitates RNoPE. We aim to see how ALiBi’s replacement of NoPE will affect the performance of the model across various benchmarks. We will denote this alternating ALiBi RoPE method as ARoPE.

3.3 Summary of Methods Evaluated

Ultimately, we evaluate the performance of the novel methods established in this section with relevant baseline methods. Firstly, we believe that the well-established, widely used RoPE-based methods serve as adequate baselines for fractional RoPE. In particular, we select base RoPE (no interpolation), linear interpolation RoPE, NoPE and YaRN as baselines for these fractional methods. In order to investigate the significance of the hyperparameter α , we evaluate fractional RoPE for $\alpha \in \{0.5, 1, 2\}$. We also evaluate $\alpha \rightarrow 0^+$ implicitly, since this is equivalent to linear interpolation RoPE by our definition of fractional RoPE.

Model	Short Name	Purpose
Base RoPE	Base	Baseline for all novel methods
Linear interpolation RoPE	Linear	Baseline for all novel methods
YaRN	YaRN	Fractional RoPE baseline
Fractional RoPE ($\alpha = 1$)	Frac(1)	Novel method
Fractional RoPE ($\alpha = 0.5$)	Frac(0.5)	Novel method with smaller hyperparameter
Fractional RoPE ($\alpha = 2$)	Frac(2)	Novel method with larger hyperparameter
NoPE	NoPE	ARoPE and HyPE baseline
ALiBi	ALiBi	ARoPE and HyPE baseline
RNoPE	RNoPE	ARoPE and HyPE baseline
ALiBi RoPE	ARoPE	Novel method
Hybrid Positional Embeddings	HyPE	Novel method

TABLE 3.1: Table of the different context extension methods we evaluate, with respect to their purpose in our investigation.

When assessing the performance of HyPE, we evaluate it with respect to its component methods, RoPE and ALiBi. For ARoPE, we also evaluate with respect to the component methods RoPE and ALiBi. This evaluation allows us to assess how the strengths and weaknesses of different attention mechanisms influence the performance of the final hybrid model. We also compare ARoPE to RNoPE to analyse how the replacement of NoPE layers with ALiBi layers influences the final model. Table 3.1 summarises all context extension methods evaluated, and their purpose in our exploration. It also serves as a reference to the shortened names of different models in the results tables of Chapter 5.

Note that the Transformers library we use to build our custom models only supports linear interpolation RoPE and YaRN for Llama2 by default. We implement all other context extension methods ourselves. We also note that across our implementations, all novel methods we defined do not increase the model’s training and inference time complexities relative to their baselines. Fractional RoPE is implemented by precomputing the rescaled RoPE angles before any inference is done. Our ALiBi implementation also precomputes the linear biases for every possible positional embedding, though the cache is significantly larger than that of RoPE’s. We are forced to split our batches when applying ALiBi, incurring a $\sim 25\%$ time penalty to the training and inference times of ALiBi and HyPE. The penalty is smaller for ARoPE, as only half the layers apply ALiBi.

Experiments

We evaluate our methods by integrating them into the Llama2-7b base model, with a context window extended from 2^{12} to 2^{13} . We then perform minimal finetuning on the model using a filtered version of the PG19 dataset, before evaluating the model’s performance on the LongBench benchmark. We describe the process in more detail in the following sections.

4.1 Base Model

For our investigation, we implement context extension on the Llama2-7 billion-parameter model (Llama2-7b) developed by Meta AI (Touvron et al., 2023). This model is open source and freely available to use for LLM research. We chose this specific model for multiple reasons. Firstly, Llama2-7b is frequently used as a baseline for evaluating context extension methods. This makes it easier to compare our results with those of existing papers. The source code is also freely available in PyTorch and the Transformers library, and is by default supported by many existing evaluation benchmarks. Secondly, the model has a simple decoder-only architecture. The model’s simplicity reduces the likelihood that other components of the model introduce bias in determining which context extension methods work best, allowing for a fairer comparison. Thirdly, the model is trained using base RoPE embeddings. This reduces the finetuning required to integrate RoPE-based extension methods. Although this design is unfavourable for non-RoPE methods we investigate, such as NoPE and ALiBi, the results will illustrate how important similarity is between the model’s base positional embedding method and the context extension method. Finally, the 7-billion parameter model has a low starting context window of 2^{12} tokens and is approximately 13GB in size at 16-bit floating-point precision. For context extension to 2^{14} tokens, inference can be done with a 40GB VRAM GPU and finetuning can be performed on a 96GB VRAM GPU with a batch size of 5. For a smaller context extension to 2^{13} tokens,

Llama2 format: What colour is the sky? Blue.
 Llama2-chat format: <s>[INST]What colour is the sky?[\INST] Blue.</s>

TABLE 4.1: Illustration of the differences in chat template formats between the base and chat versions of Llama2 models.

finetuning can be performed on a 96GB VRAM GPU with a batch size of 10. Since hardware was a major restriction, the small model size and small initial context window size of Llama2-7b best suited our experiments.

Another key decision was the choice between using Llama2-7b and Llama2-7b-chat. The base model (Llama2-7b) is trained for next-token prediction across all many forms of text. Llama2-7b-chat takes Llama2-7b’s final parameter configuration and then receives further training for next-token prediction on conversational-style text. This conversation style follows a fixed chat template, with an example illustrated in Figure 4.1. In order to effectively perform finetuning, the data needs to follow the structure required by the model. The base model can be finetuned on any structured text by design, making it easy to find or create a suitable dataset. In contrast, the chat model requires the dataset to follow the chat template. For long-context tasks specifically, there is a much scarcer availability of Q&A style datasets that can be modified to follow the chat template, compared to large bodies of text that are already compatible with the base language model. As such, we choose to use the base Llama2-7b model over the chat model. This decision comes with the disadvantage that our chosen evaluation benchmark, LongBench, prompts the model in a Q&A format, potentially introducing a performance bias toward chat-based models. However, since we only compare different modifications of a non-chat model, this bias does not become an issue. Furthermore, it is likely that performance differences between the various methods extrapolate from the base model to the chat model when finetuned with a suitable chat-based long context dataset.

Our context extension methods were implemented as subclasses of various Llama-specific classes in the Transformers library. For each method, we implement variants of the classes described in Table 4.2. Note that some classes are not always required for certain models. For example, Fractional RoPE requires changes to the LlamaRotaryEmbedding class since the method redefines the cache of sine and cosine values. In comparison, ALiBi forsakes rotary embeddings entirely.

Base Class	Description	Common Changes
LlamaConfig	Stores the custom model configuration settings, and is saved and loaded as a config.json file.	Adding additional configuration settings.
LlamaModel	Stores the bare model and manages the required forward and backward computations.	The loaded instance of LlamaModel is changed by other classes instead.
LlamaRotaryEmbedding	Performs the necessary RoPE angle computations.	Modifying RoPE computations.
LlamaAttention	Performs the attention calculations.	Introducing custom attention computations.
LlamaForCausalLM	Wraps the LlamaModel for use in next token prediction tasks.	Replacing model components with custom components.

TABLE 4.2: Table of custom transformer classes and functions, with descriptions of what they do and how they were generally altered.

4.1.1 Finetuning

We perform minimal finetuning on Llama2-7b for each context extension method that we explore. For finetuning, we aim to be consistent with the original pretraining methods used for Llama2-7b. As such, we train the model solely on next-token prediction tasks. We specifically choose to finetune on the filtered PG-19 dataset provided in the CommonPile dataset. PG-19 comprises of over 75000 books with expired US copyright. Books are intrinsically large bodies of text that significantly exceed Llama2-7b’s context window size, making them ideal for finetuning for long context tasks.

Similarly to the pretraining of Llama2-7b, we train exclusively on next-token prediction without other methods such as masked token prediction or natural language inference. Llama2-7b was also pretrained on the AdamW optimiser. Due to computational restrictions, we use the BitsAndBytes library’s quantised implementation, PagedAdamW8bit. This 8-bit variant of AdamW reduces the memory costs for finetuning the model by storing the means and standard deviations it uses for backpropagation at a lower precision. The performance degradation from using lower precision optimisers has also been found to be minimal (Dettmers et al., 2022). Paging also allows optimiser states to be transferred to CPU memory when the GPU’s VRAM is full. The combination of 8-bit optimisation and paging allows us to finetune on a larger batch size while minimally degrading training performance and computation time. Specifically with a 96GB VRAM GPU, we can finetune with a batch size of 10 and a context window size of 2^{13}

at an average speed of 25 seconds per batch. This time goes up to 32 seconds per batch when using an ALiBi-based context extension method.

We also choose to use a larger initial learning rate 10^{-4} , for the following reasons: Firstly, since we perform minimal iterations of finetuning, we want to encourage larger gradient updates to achieve faster convergence. Existing context extension experiments that train on larger numbers of examples, such as YaRN, PoSE and linear interpolation RoPE, use a learning rate of 2×10^{-5} for comparison. In practice, we find that the high initial learning rate still results in a monotonically decreasing validation loss between every 100 batches, suggesting that our learning rate is not too high. Secondly, there is the possibility that we want to perform further finetuning on a model that we have already finetuned. This possibility arises due to computational limitations, where the cost of fully retraining a model over a larger number of iterations is too high. If further finetuning is required, we can resume training with a lower starting learning rate. We specifically choose to use a linear learning rate scheduler that dynamically calculates the learning rate as modeled by Equation 4.1, where for I total iterations, lr_i is the learning rate at iteration i and lr_0 is the starting learning rate.

$$lr_i = lr_0(I - i)/I \quad (4.1)$$

Ultimately, our complete finetuning process performs 3200 iterations on a batch size of 10 with a context window size of 2^{13} . Our model is finetuned on 32000 training examples total, with an expected runtime of just over 22 hours when excluding evaluation steps. We also compute the evaluation loss on a held-out subset of the training data of size 50 every 100 training iterations, producing 32 total evaluation steps. This evaluation loss is used by our early stopping criteria, which asserts that after eval loss increases 3 times total throughout training, training should stop. In practice, the early stopping criteria was never triggered. After finetuning is complete, we save our final model and its custom configuration for evaluation.

4.2 Evaluation

We use a variety of methods to evaluate the performance of our models on long-context tasks. Primarily, we use LongBench and the perplexity metric for our evaluation. We also use auxiliary

Dataset	Abbreviation	Source	Avg len	Metric	Answer Type	#data
<i>Single-Document QA</i>						
NarrativeQA	NQA	Literature, Film	18,409	F1	Short	200
Qasper	QAS	Science	3,619	F1	Short	200
MultiFieldQA-en	MFQA	Multi-field	4,559	F1	Short	150
<i>Multi-Document QA</i>						
HotpotQA	HPQA	Wikipedia	9,151	F1	Keywords	200
2WikiMultihopQA	2WQA	Wikipedia	4,887	F1	Keywords	200
MuSiQue	MSQ	Wikipedia	11,214	F1	Keywords	200
<i>Summarization</i>						
GovReport	GOV	Government report	8,734	Rouge-L	Paragraph	200
QMSum	QMS	Meeting	10,614	Rouge-L	Paragraph	200
MultiNews	MN	News	2,113	Rouge-L	Paragraph	200
<i>Few-shot Learning</i>						
TREC	TREC	Web question	5,177	Accuracy (CLS)	Keywords	200
TriviaQA	TRQA	Wikipedia, Web	8,209	F1	Keywords	200
SAMSum	SAM	Dialogue	6,258	Rouge-L	Few Sentences	200
<i>Synthetic Task</i>						
PassageCount	PC	Wikipedia	11,141	Accuracy (EM)	Number	200
PassageRetrieval-en	PRE	Wikipedia	9,289	Number	English	200

TABLE 4.3: Table of tasks statistics in LongBench. This table was adapted from Bai et al. (2024), with the Chinese and code-based tasks removed.

metrics such as validation loss in our discussion and refer to example outputs to compare the performance of different context extension methods.

4.2.1 LongBench

We use LongBench (Bai et al., 2024) to compare the performance of different methods on a variety of short-answer tasks. We specifically only test the English tasks, discarding the Chinese and code-based tasks. This is due to our choice of an entirely English-based finetuning dataset, which is expected to improve performance only on English-based tasks. This restriction reduces the number of datasets in LongBench from 21 to 14, allowing it to still provide a reliable evaluation of a model’s performance. Due to computational constraints, we only evaluate on the first 40 entries of each dataset instead of all 200, reducing evaluation time from approximately 10 hours to 2 hours on a 2^{13} context window size. Table 4.3 summarises the English LongBench tasks as described by Bai et al. (2024) and defines the abbreviations used throughout Sections 5 and 6. Note that if an input exceeds our extended context window length L' , it is truncated in a way that retains the first and last $L'/2$ tokens. This is the default truncation method used by LongBench. We also adapt the benchmark to work with non-chat-based models, as we implemented our context extension methods on the base Llama2-7b model.

We also report each model’s validation loss throughout training. Closely related to perplexity, loss measures the difference between the model’s predicted probability distribution and the true probability distribution for the next token to be generated on given input text. It is useful for comparing the rates that different methods learn longer context relationships across the course of training. For next-token prediction, the loss is simply the negative log-likelihood that is also applied in the following subsection.

4.2.2 Perplexity

We also use per-token average perplexity as a secondary performance measure. The perplexity of an input text is mathematically defined as the exponent of the averages of the negative log likelihoods of all the tokens, as illustrated in Equation 4.2. Thus, the macro-average complexity across multiple different text inputs can be computed as the average of these perplexities as expressed in Equation 4.3. Here we define $|X|$ as the number of distinct input texts, and $|x|$ as the number of tokens in text entry x . However, this method of computing perplexity assigns equal weightings to every input regardless of the number of tokens in the text. This means that text entries with fewer tokens have their negative log-likelihoods weighted more heavily in the average perplexity calculation than entries with more tokens. To ensure the contributions of all tokens are weighted equally, we instead use micro-average perplexity as defined in Equation 4.4. This method instead averages the negative log-likelihoods across all tokens, rather than averaging the perplexity of each individual text entry. In addition, we ensure that padding tokens do not contribute to the perplexity calculation and truncate all entries to the extended context window length.

$$PPL(x) = \exp \left(-\frac{1}{|x|} \sum_{i=1}^{|x|} \log P(x_i | x_{<i}) \right) \quad (4.2)$$

$$PPL_{macro}(X) = \frac{1}{|X|} \sum_{x \in X} \exp \left(-\frac{1}{|x|} \sum_{i=1}^{|x|} \log P(x_i | x_{<i}) \right) \quad (4.3)$$

$$PPL_{micro}(X) = \exp \left(-\frac{1}{\sum_{x \in X} |x|} \sum_{x \in X} \sum_{i=1}^{|x|} \log P(x_i | x_{<i}) \right) \quad (4.4)$$

We evaluate perplexity on a held-out set of the training data from the CommonPile PG-19 filtered dataset containing 1000 entries. By evaluating on an unseen subset of the data in the

same distribution as that seen in training, perplexity represents how well the model has learned the distribution of the training data. The perplexity results from a model finetuned on a single dataset correspond to how well the model has learned long-context relationships, and are likely to extrapolate to other English-based benchmarks.

Results

In this chapter, we describe the key results from experiments outlined in our methods and experiments sections. Here, we will specifically discuss our results in relation to our first research question: “Can we design effective context extension methods that require minimal or no additional finetuning?” We will discuss our other research questions in more detail in Chapter 5. For compactness, we use the notation MODELTYPE-k to denote a model of type MODELTYPE with a context window size of 2^k . When referring to the context extension method itself, we state the method name directly. The abbreviations for the various types of models we use are defined in Table 3.1. For fractional models with an α hyperparameter, we write $\text{Frac}(\alpha)$ -k. We additionally insert an asterisk to write MODELTYPE*-k if the model has received 3200 iterations of finetuning, omitting the asterisk if no finetuning was performed. The various LongBench datasets are defined by their acronyms in Table 4.3. Model names in bold denote our novel models. Results in bold highlight the strongest performance against the corresponding metric.

5.1 Non-finetuned Models

First, we consider the results of the selected context extension methods without finetuning. The performances of each of these on LongBench are presented in Table 5.1. Since no finetuning has been performed for the following models, we expect to see poorer performance across most LongBench tasks in comparison to the 32k-finetuned models.

5.1.1 Baseline RoPE Models

In general, we observe that the base Llama2-7B model (Base-11) with the original context window size performs strongly against most tasks. In particular, the base model dominated

against all non-finetuned models in passage retrieval (PRE), government report (GOV) and multi-file Q&A (MFQA), with the next best models scoring 4.79%, 7.6% and 5.93% lower respectively, in these tasks. Base-11 also marginally performs best in query-based multi-domain meeting summarisation (QMS), while being competitive with the other top-performing models in across all datasets. The exceptional performance of the base model is expected, since it is the only model that has not been altered after its original training. Because no finetuning was performed on the other models, they have not adapted to the alterations in their architectures and thus, are expected to degrade in performance.

For the other RoPE-based baseline models, we observe that linear interpolation RoPE (Linear-13) also performs strongly without finetuning. It performs best on passage count (PC), trivia Q&A (TRQA), multinews (MN), and Hotpot Q&A (HPQA). It particularly dominates over the base model in MN, improving performance significantly from 4.27% to 16.35%. It is also competitive against the top performing models in SAMsum (SAM), Text Retrieval Conference (TREC), QMS and 2WikiMultihopQA (2WQA). One clear weakness across all non-finetuned models is PRE, where performance is always significantly lower than Base-11. In general, the comparably strong performance of Linear-13 across all tasks without finetuning is reasonable since its only real change is the rescaling of the position ids, which causes the model to process each pair of tokens as if they were closer than they actually are. The slight performance drops can be attributed to this reason, as well as to the presence of unseen angles for non-integer differences in token positions. However, slight gains in performance over Base-11 are likely due to the larger context window, which allows more tokens to be processed and this yields more informed model responses.

Compared to Linear-13, YaRN-13 performs marginally weaker across most LongBench tasks. This is with the exceptions of Musique (MSQ) and SAM, where performance was equal and marginally better in respective cases. YaRN (aptly named Yet another RoPE extensioN) is a piecewise interpolation of linear RoPE and NTK-aware RoPE. The weaker performance of YaRN-13 compared to Linear-13 without finetuning is likely due to YaRN deviating more greatly from the original model architecture due to this interpolation.

5.1.2 Fractional RoPE Models

In general, the three fractional RoPE models perform better than the baseline RoPE-based models. With the exceptions of PRE, MN, GOV and MFQA, at least one of the three models performs best or slightly behind the best model. In particular, we see that $\alpha = 0.5$ produced the best results across SAM, TREC, MSQ and 2WQA. The strong performance across the fractional RoPE models generally suggests that they are competitive with Base-11 even without additional finetuning. This suggests that the constraints satisfied by fractional are appropriately designed to minimise the model’s performance degradation under context extension.

When comparing the performance difference between different values of α , we observe different orderings of performance. In the first ordering, for 8/14 tasks, performance decreases as α increases. For some tasks, such as SAM and TRQA, the performance drop from $\alpha = 1$ to $\alpha = 2$ is drastic, being 29.9% and 49.15% respectively. Then, for another 4/14 tasks, $\alpha = 1$ outperforms $\alpha = 0.5$, with $\alpha = 2$ remaining the worst-performing choice of hyperparameter. For the final 2/14 tasks (PC and GOV), we observe an inversion in the relationship between performance and α , with performance increasing as α decreases. These observations suggest that the ideal value of α is task-sensitive, but a smaller α is generally preferred without any finetuning. This means that fractional RoPE performing closer to linear interpolation RoPE than base RoPE results in better performance across most tasks.

5.1.3 Non-RoPE Methods

Across all non-RoPE methods, performance drops severely on all LongBench tasks when no additional finetuning is performed. We observe that many scores reach 0%, indicating that the model entirely hallucinates regardless of the input. The significant performance decline can be entirely justified to how drastically the NoPE, ALiBI, RNoPE, ARoPE and HyPE methods change the model’s architecture. The RoPE-based methods all fundamentally maintain the use of RoPE in the base model, with modifications being made to the angle computations. In contrast, the non-RoPE methods replace RoPE with entirely different attention mechanisms that are unrelated in mechanism. Thus, the existing models’ parameters are essentially incompatible with the altered models, resulting in their degraded performances.

Of these non-RoPE models, we expected RNoPE-13 and ARoPE to have the highest probability of yielding a non-zero performance. This is because RNoPE and ARoPE only replace the positional encoding mechanism in every even index layer, applying linear interpolation RoPE in the odd index layers. This hypothesis is not supported by the results, since ALiBi has on average the best performance across the non-RoPE models. Looking at the results table alone, the slightly higher performance of ALiBi on tasks such as TREC, MN, PC and QMS may suggest that these tasks rely significantly more on nearby tokens than more distant ones, as ALiBi applies the smallest reduction to attention scores for nearby tokens. However, the performance of ALiBi without finetuning is too weak to confidently support this justification. Outside of this, we see another interesting outlier of RNoPE-13 producing the second-highest score on PRE. A score of 5.00 likely indicates that the model got 2/40 test examples completely correct, while failing the other 38 examples. Since PRE involves returning a paragraph number from 1 to 30, we observe that this result was due to RNoPE-13 only guessing the numbers 1 and 10, resulting in two correct predictions being made by random chance. Thus, this result is not significant for comparing the non-RoPE context extension methods. NoPE-13 and HyPE-13 likely suffered in performance for the same reason as the other non-RoPE methods. NoPE removes positional embeddings entirely, whereas HyPE on top of using RoPE, adds ALiBi to each layer.

Overall, it is important to note that the poor performance of non-RoPE methods without finetuning does not imply that they are necessarily worse than RoPE-based methods. The non-finetuned results are fundamentally biased towards RoPE-based methods, since Llama2-7b was originally trained with RoPE embeddings.

Model	LongBench Task													
	PRE	PC	SAM	TRQA	TREC	MN	QMS	GOV	MSQ	2WQA	HPQA	MFQA	QAS	NQA
Base-11	9.79	2.50	40.17	80.38	67.50	4.27	20.54	27.10	2.95	10.70	8.13	26.40	7.76	14.73
Linear-13	1.25	2.73	42.05	84.92	67.50	16.35	20.50	19.50	3.43	10.65	8.59	19.23	10.31	15.52
YaRN-13	1.25	2.27	43.00	81.66	45.00	13.34	14.56	7.44	3.43	9.80	4.55	19.01	9.65	14.82
Frac(0.5)-13	1.45	0.23	43.50	84.08	70.00	5.13	18.27	14.47	3.86	10.78	8.15	20.43	10.27	13.45
Frac(1)-13	2.33	0.51	41.82	80.33	67.50	5.33	18.40	16.05	3.25	9.60	6.97	18.52	10.60	7.57
Frac(2)-13	1.25	2.27	11.92	31.18	60.00	7.01	5.33	14.47	0.88	7.84	3.04	14.60	9.93	1.83
NoPE-13	0.00	0.00	0.00	0.00	0.00	0.80	1.32	0.15	0.00	0.00	0.00	0.18	0.00	0.81
ALiBi-13	0.00	1.25	0.00	0.00	5.00	4.56	1.69	0.71	0.00	0.00	0.00	0.57	0.08	0.00
RNoPE-13	5.00	0.00	0.00	0.36	0.00	0.95	0.25	0.59	0.29	0.00	0.00	0.00	0.00	0.00
ARoPE-13	0.00	0.00	0.00	0.00	0.00	0.00	0.19	0.07	0.30	0.00	0.00	0.00	0.03	0.00
HyPE-13	0.00	0.00	0.00	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00

TABLE 5.1: Table of results across different models without any additional finetuning performed. The best-performing model for each dataset has its performance highlighted in bold. Our novel model architectures have their labels written in bold.

5.2 32k-finetuned Models

Now we consider the performance of the outlined context extension methods finetuned on 32,000 examples from CommonPile’s filtered PG-19 dataset. These results are tabulated in Table 5.2. Note that we compare these methods to the base Llama2-7b model, since it has already been extensively trained without changes to its architecture. We will save the direct comparison between non-finetuned and finetuned versions of the same models for Chapter 6.

5.2.1 Baseline RoPE Models

Overall, we observe that the original Base-11 llama model remains the best-performing model across the QMS and GOV tasks. Due to the finetuning of the other models, it is now only the second-best model on PRE and MFQA. Two key weaknesses of Base-11 compared to other models are MN and QAS, where many other RoPE-based models outperform it. This suggests that these tasks especially benefit from a larger context window, since information critical to answering the question is less likely to fall outside it.

Surprisingly, we observe that neither Linear*-13 nor Yarn*-13 are dominant across any of the tasks despite being well-established context extension methods for RoPE-based models. Linear interpolation RoPE still performs comparatively strongly on tasks such as TRQA, MN, MSQ and 2WQA where it does not fall far behind the best-performing model. However, it performs particularly poorly on the PRE and PC tasks compared to the best performing models, with differences in scores of 16.67% and 1.9% respectively. However, linear interpolation RoPE still beats Base-11 across a variety of tasks, including SAM, TRQA, MN, MSQ, 2WQA, and QAS.

In general, YaRN*-13 performs significantly worse than Linear*-13 across most tasks. The two exceptions to this trend are PC and QAS, where YaRN results in slightly better performance. One of the key design strengths of YaRN, is that across different dimension heads, it minimises the overlap in rotation frequencies between them. Resonance between different head dimensions is not ideal because the two dimensions can capture the same features, creating redundancy in the network. Hence, one explanation for YaRN not improving the model’s performance is that the extended context window size $L' = 8192$ is too small to make this resonance a major issue. For linear interpolation RoPE, as the context window size increases, the amount of period overlap will increase over time. YaRN sacrifices the simple linear scaling of RoPE with position

to avoid this issue, instead using piecewise linear scaling, which may lead to the performance decrease we observe in our results. In contrast, fractional RoPE also abandons linear scaling with position, but introduces other properties that we believe are desirable regardless of the context extension factor. Thus, it is possible that YaRN outperforms linear interpolation RoPE at a larger context length.

5.2.2 Fractional RoPE Models

The three fractional RoPE models in general outperformed the Linear*-13 and YaRN*-13 baselines. This performance gap includes tasks such as PRE, SAM, TREC, MN, QMS, Qasper (QAS), and Narrative Q&A (NQA), where all three fractional models outperformed both baselines. This suggests that fractional-RoPE’s design considerations have allowed it to perform strongly in context extension across most tasks. In contrast, we observe a distinct weakness in the PC dataset, where both Linear*-13 and YaRN*-13 outperform all three fractional RoPE methods. However, every context extension method struggles completely with this task, making it hard to justify its use for comparison.

When comparing the different values of α , the performance range across most tasks is relatively small. PRE is a significant outlier in this respect, where $\alpha = 1$ dominates other values of α , beating the second-best model, Base-11, by 10.64%. Compared to non-finetuned fractional models, the ordering of the finetuned fractional models varies more widely in terms of performance. This could suggest that the exact choice of α matters less when finetuning is applied, since the model can adapt to different values of α . Overall, fractional RoPE demonstrates great potential as an alternative to existing RoPE-based methods of context extension with minimal finetuning.

5.2.3 Non-RoPE Methods

Overall, we observe substantial performance improvements in non-RoPE methods after finetuning. Of these models, ALiBi*-13 performs the weakest over the majority of tasks. In particular, it is the worst-performing model on PC, SAM, TRQA, TREC, MN, QMS and MFQA by significant margins. In contrast, ALiBi*-13 is the best performing model on MSQ and 2WQA, both of which are multi-hop tasks. This suggests that these tasks rely more strongly on relationships between nearby tokens, since ALiBi enforces a linear penalty that is directly proportional to

the relative distance between two tokens. These results in particular demonstrate how the performance of different context extension methods can vary significantly depending on the task, and will be further explored in Chapter 6.

Compared to ALiBi*-13, NoPE*-13 performs better in the majority of LongBench tasks. This difference is particularly pronounced on TRQA, TREC, QMS, and SAM, with respective performance differences of 22.38%, 35%, 9.02% and 5.2%. These results suggest that despite finetuning being applied, it is easier to adapt to removing positional embeddings entirely than to replace them with a completely different method. In this case, ALiBi shares no similarities with RoPE and thus the parameters learned by Base-11 have no relation to the new architecture of ALiBi*-13. In addition, ALiBi’s implicit bias towards nearby tokens during attention calculations will naturally make NoPE perform better in tasks involving important pairs of tokens that are farther apart. Regardless, NoPE still underperforms compared to most other RoPE and non-RoPE methods, as it is forced to learn the significance of position without having an existing representation of position. This is unlike other methods, which introduce positional embeddings to represent position directly. Thus, these methods have an implicit advantage under minimal finetuning.

For RNoPE, we see that it dominates in HPQA, MFQA and QAS. Despite this, there is still a reasonable performance gap between it and the best model across other tasks. This reveals that, like ALiBi*-13, RNoPE*-13 has developed a specific set of strengths likely due to the properties of its chosen positional embedding types. Unlike NoPE*-13, which had its RoPE position embeddings removed at every layer, RNoPE*-13 only had RoPE removed at even-index layers. This means that the odd-index layers only need to learn to adapt to the outputs of the previous layer. In contrast, NoPE and ALiBi must adapt to both the outputs of the previous layer and the altered positional embedding method. Thus, it is understandable why RNoPE*-13 performs significantly stronger across many tasks in comparison to NoPE and ALiBi, with a few exceptions. An interesting exception is the 0% performance on PRE, where despite RNoPE*-13 predicting a variety of chapters indices, the model always makes the wrong prediction.

Compared with RNoPE, ARoPE replaces even-index layers with ALiBi rather than NoPE. Hence, just like the comparison between NoPE*-13 and ALiBi*-13, we expect that RNoPE*-13 would generally outperform ARoPE*-13. This is true for the majority of LongBench tasks, with the exceptions of PRE, MSQ, 2WQA and NQA. Note that in MSQ, 2WQA, and NQA,

ALiBi also outperforms NoPE. Hence, the reasons for the differences in performances between ARoPE*-13 and RNoPE*-13 are likely consistent with those of ALiBi*-13 and NoPE*-13. Based on the results, there is no motivation to use ARoPE in place of ALiBi or the other RoPE-based methods as ARoPE never outperformed any of them.

Finally, HyPE*-13 is outperformed by Linear*-13 across the majority of tasks. Even in MSQ and 2WQA, the two tasks in which ALiBi*-13 performed best in, HyPE*-13 provided no improvement over Linear*-13. For the same reasons that ALiBi*-13 performs poorly on most tasks, HyPE*-13 likely performs poorly because it is difficult to adapt to the addition of ALiBi despite RoPE being retained.

In general, the two proposed hybrid methods are not promising candidates for context extension with minimal finetuning from a RoPE-based model. If a base model trained with ALiBi were used instead, ARoPE or HyPE may perform better in comparison to other RoPE-based methods. ALiBi still thrived on MSQ and 2WQA despite this obstacle. Overall, RNoPE proves itself to be the most promising hybrid model, performing the best on three different tasks, while performing similar to the best RoPE-based models on many others.

Model	LongBench Task													
	PRE	PC	SAM	TRQA	TREC	MN	QMS	GOV	MSQ	2WQA	HPQA	MFQA	QAS	NQA
Base-11	9.79	2.50	40.17	80.38	67.50	4.27	20.54	27.10	2.95	10.70	8.13	26.40	7.76	14.73
Linear*-13	3.75	0.83	40.94	84.92	56.94	24.05	18.43	22.26	4.31	12.83	7.74	21.74	11.26	10.56
YaRN*-13	0.42	2.10	17.56	36.70	47.50	18.36	3.88	6.53	0.00	5.26	3.37	20.48	12.23	1.10
Frac*(0.5)-13	8.33	0.45	42.39	82.92	70.00	25.58	18.70	19.59	3.99	12.96	8.76	15.27	12.81	13.31
Frac*(1)-13	20.42	0.45	44.44	84.83	67.50	24.55	20.27	18.69	3.10	13.59	7.59	20.53	12.13	15.94
Frac*(2)-13	5.00	0.56	41.06	85.33	62.50	24.45	19.53	20.77	3.65	11.32	9.09	17.78	13.59	13.03
NoPE*-13	2.50	2.73	7.01	26.73	35.00	0.62	13.40	2.46	2.84	9.14	2.97	9.75	5.91	4.55
ALiBi*-13	2.40	0.00	1.81	4.35	0.00	0.55	4.38	6.96	4.50	13.78	6.07	7.15	9.35	9.41
RNoPE*-13	0.00	2.05	35.66	82.60	57.50	12.91	17.83	11.59	2.61	10.30	12.16	30.93	14.42	8.20
ARoPE*-13	3.37	2.05	32.52	37.66	32.50	12.52	14.36	11.38	3.67	11.12	5.82	10.71	9.82	11.70
HyPE*-13	2.81	2.27	3.95	13.17	2.44	8.02	12.59	8.71	4.31	12.51	9.32	10.69	8.42	6.94

TABLE 5.2: Table of results across different models with 3200 iterations of finetuning at batch size 10. The best-performing model for each dataset has its performance highlighted in bold. Our novel model architectures have their labels written in bold.

Model	Micro Perplexity	
	Non-finetuned	32k-finetuned
Base-11	7.115	-
Linear-13	7.600	5.664
YaRN-13	7.599	5.789
Frac(0.5)-13	7.593	5.640
Frac(1)-13	7.804	5.647
Frac(2)-13	10.366	5.658
NoPE-13	385.220	7.853
ALiBi-13	503.845	11.041
RNoPE-13	458.786	5.893
ARoPE-13	1796.100	7.166
HyPE-13	530.634	9.092

TABLE 5.3: Table of micro-perplexities across different models on a 1000-entry held-out subset of the CommonPile filtered PG-19 dataset, for both non-finetuned and 32k-finetuned models. The lowest perplexity model in each category has its performance highlighted in bold (ignoring Base-11). Our novel model architectures have their labels written in bold. Note that a lower perplexity is better, as it indicates lower uncertainty in the model.

5.3 Perplexities

In addition to using LongBench to evaluate different context extension methods, we compute the perplexities of each model on a held-out set of the filtered CommonPile PG-19 dataset. Specifically, we compute micro-complexity as previously defined in Equation 4.4. We list the computed perplexities of the non-finetuned and finetuned models in Table 5.3. In general, a lower perplexity suggests that the model has a greater average probability predicting the tokens in the given text. For finetuned models in particular, a smaller perplexity indicates that the model has better learned the distribution of the finetuning dataset.

5.3.1 Non-finetuned Perplexities

When analysing the non-finetuned models, we see that the Base-11 model has a perplexity 0.478 lower than the next-best model. This is expected, since the base model has been extensively trained to process inputs of up to a context length of 2^{11} . Hence, this model would have the lowest perplexity as it is more familiar with text at its smaller context window size than

the other models at their larger sizes. In comparison, all of the rotary approaches also have slightly higher perplexities within the range 7.593 to 10.366. As expected, integrating a RoPE-based context extension method into a RoPE based-model would result in a higher certainty in predictions in comparison to a non-RoPE approach. Thus, the perplexities of the RoPE-based approaches are lower than those of the non-RoPE approaches. Note that these models have not been trained with a 2^{13} context window size and have some alterations from the base RoPE model’s architecture. Consequently, they have a higher perplexity than the Base-11 model.

We also see that the fractional models for $\alpha = 0.5$ and $\alpha = 1$ have similar perplexities to Linear-13 and YaRN-13. This suggests that the constraints imposed to form fractional RoPE achieve their goal of remaining similar to the base model. It’s clear that the lowest value of $\alpha = 0.5$ yields the lowest perplexity. However, recall that linear interpolation RoPE is the boundary case where $\alpha \rightarrow 0^+$. We observe that $\alpha = 0.5$ has a lower perplexity than both $\alpha \rightarrow 0^+$ and $\alpha = 1$, suggesting that there may exist an α between 0 and 1 that produces an even lower perplexity.

Now considering the non-RoPE methods, we see a drastic jump in perplexity. NoPE exhibits the smallest increase, by a factor of approximately 37 in comparison to the highest perplexity RoPE method tested. This explosion in perplexity can be attributed to the greater architectural change of the model after transitioning from a RoPE-based method to a non-RoPE method. In particular, we observe that ARoPE-13 has more than a 3 times higher perplexity than the next highest model, HyPE-13. This may be an indicator that ARoPE should struggle the most in finetuning. It is also interesting that RNoPE-13 and ARoPE-13 have higher perplexities than ALiBi-13 and HyPE-13 respectively, despite only half of their layers being altered in comparison.

5.3.2 32k-finetuned Perplexities

After performing 3200 iterations of finetuning, we observe that the majority of the models have lower perplexity than the non-finetuned Base-11 model. This can be attributed to one main reason. Despite the Base-11 model being extensively trained, a PG-19 likely made up a small fraction of Llama’s training dataset if any at all. In contrast, the context extended models have been fully finetuned on the PG-19 subset at a high learning rate. This makes them more likely to correctly predict tokens in unseen PG-19 text than Base-11, which has seen a much more

diverse range of data entries. Despite this, our comparison between the extended context models remains valid since they have all undergone the same training and finetuning procedures.

When analysing the perplexities of the RoPE-based finetuned models, they all fall in the narrow range of 5.640 to 5.789. This suggests that the RoPE-based models have best learned the finetuning dataset. In addition, there is very little change in perplexity between the fractional RoPE models, with perplexity varying by no more than 0.018. We also observe that RNoPE has a comparably low perplexity to the RoPE-based methods, with a perplexity only 0.104 higher than that of YaRN*-13. This implies that despite RNoPE introducing NoPE layers, it was able to learn the training dataset to a quality similar to that of the RoPE-based methods.

For the non-RoPE methods, we observe an interesting shift from the non-finetuned results. After finetuning, we see that RNoPE and ARoPE now have lower perplexities than their components, NoPE and ALiBi respectively. This aligns with the expectation that a model more similar to the original would better learn the training dataset, since its parameters require smaller adaptations. We consider RNoPE and ARoPE to be more similar to RoPE in architecture than NoPE and ALiBi respectively, since only half of their layers are replaced. Similarly to the non-finetuned results, we also observe that the NoPE-based methods have lower perplexities than the ALiBi-based methods, suggesting that it is easier to adapt RoPE to NoPE than to ALiBi. Finally, HyPE*-13 possesses the second highest perplexity just as HyPE-13 does for the non-finetuned models. This suggests that combining RoPE and ALiBi in each layer may hinder the model’s ability to learn from the dataset, possibly introducing too much complexity into the calculation.

To summarise, the perplexity results suggest that the RoPE-based models have learned the training dataset best, with the fractional RoPE models marginally performing the best after finetuning. In comparison, the hybrid methods started with a much poorer understanding, and after finetuning are not far behind the perplexities of the RoPE-based models. A low perplexity indicates that the model has a good understanding of the distribution of the dataset at the defined context window size well. If the dataset is well chosen, it may also lead to the model extrapolating to other datasets and benchmarks at the same context window size. However, as Fang et al. (2025) suggest, perplexity may not be an accurate metric for the performance of long context models. We will assess this claim for with respect to our models in Chapter 6.

Discussion

In this chapter, we discuss our results in direct relation to our research questions. Furthermore, we attempt to explain our results in relation to example model predictions and additional results.

6.1 Research Question 1

Can we design effective context extension methods that require minimal or no additional finetuning?

This research question was extensively covered in our discussion of the results in Chapter 5. We will summarise the key findings of our results with respect to the research question for non-finetuned and finetuned models separately.

6.1.1 Non-finetuned Models

In general, we observed that RoPE-based context extension methods performed significantly better than non-RoPE methods with zero finetuning. This is primarily due to substituting a RoPE method with a non-RoPE method resulting in a more significant change to the model’s architecture. Thus, the model’s parameters that were trained on the original RoPE-based model are significantly less calibrated to the new architecture, resulting in worse performance.

We can see this in an example from the TRQA dataset in Figure 6.1. Here, we compare the outputs of Base-11, Linear-13, Frac(1)-13, ALiBi-13 and ARoPE-13 on the same input prompt. For the sake of space, we replace the full context with the “<context>” tag. We will also remove extra newline symbols “\n” for compactness, and omit the examples for few-shot prompting tasks. Note that non-chat models can still generate text after providing their answers,

resulting in additional text often being generated after the answer is provided. It is clear that the RoPE-based models are all able to answer the question correctly despite not undergoing finetuning. This may be due to the model correctly extracting the relevant information from the context window. Alternatively, the model may have known the answer from the base Llama2 model’s training process. In contrast, the non-RoPE models produced nonsensical or non-English responses. This explains the exceptionally poor results across all non-RoPE models on LongBench. Thus, this example, like many others, illustrates the impact of making significant changes to the model architecture without additional finetuning. It also illustrates that not even the base model was complied with the prompt to not output any other words, despite this part of the prompt not being truncated. This can be viewed as a limitation of Llama2 or of LongBench’s evaluation, since this occurred across many example outputs.

We are also interested in why Passage Retrieval English (PRE) was extremely difficult for even the RoPE-based context-extension models, but not as much for Base-11. PRE requires the model to find which paragraph number of a Wikipedia extract matches a provided description. Figure 6.2 illustrates an example in which Base-11 returns the correct answer, while the other two RoPE-based models predict incorrectly. There is also some evidence that Llama2 may have encountered a similar structure of text data during its training, since it was able to replicate the prompt format after returning the answer. It is difficult to ascertain why models choose the correct or incorrect answers from a single example, especially since only a number needs to be returned for the task. Instead, we analyse each model’s collection of responses across all questions. When looking at all of Base-11’s responses to PRE, we observe that 37/40 responses were paragraph 30. This suggests that Base-11 does not truly have a better understanding of passage counting than the other models, and that it has simply learned to guess 30 during its original training. In contrast, many of the non-RoPE models favour paragraph 1 over other paragraphs. Thus, this task is not a fair choice of comparison between the different non-finetuned models as Base-11 simply performed best by repeatedly making the same prediction.

Answering our first research question, we find that it is possible to achieve slight improvements in performance without finetuning. In particular, linear interpolation RoPE and fractional RoPE with $\alpha = 0.5$ have been shown to improve results on various tasks. However, the base model without context extension still outperforms our proposed methods on PRE, QMS, GOV

and MFQA. Thus, we cannot claim to have designed a model that performs universally well in context extension without any finetuning.

TriviaQA: Question 1

Prompt:

Answer the question based on the given passage. Only give me the answer and do not output any other words.

<context>

Question: In which country is the Grand Coulee Dam

Answer:

Base-11:

United States

Passage:

The 10 Best Places to Visit in the World in 2015

Linear-13:

United States of America

Passage:

The Grand Coulee Dam is a concrete arch-gravity dam on the Columbia River in the U.S.

Frac(0.5)-13:

United States of America

Passage:

The Grand Coulee Dam is a concrete arch-gravity dam on the Columbia River in the U.S.

ALiBi-13:

Hinweis: Die Daten Daten Daten sind nicht mehr alsBBB.

ARoPE-13:

n nobody nobody nobody nobody nobody nobody nobody nobody nobody nobody
nobody nobody nobody nobody nobody nobody nobody nobody nobody nobody
nobody nobody nobody nobody nobody nobody nobody nobody nobody nobody
nobody

FIGURE 6.1: An example of the outputs of 5 different non-finetuned models on the same prompt. The correct answer is any way of expressing the United States of America. We clearly see that the RoPE-based models find the correct answer, whereas the selected non-RoPE models hallucinate.

PassageRetrievalEn: Question 12

Prompt:

Here are 30 paragraphs from Wikipedia, along with an abstract. Please determine which paragraph the abstract is from.

<context>

The following is an abstract.

The Surtees team updated their car for the 1973 season, now called the TS14A, and had Englishman Mike Hailwood and Brazilian José Carlos Pace as their drivers. However, they faced a string of retirements and accidents throughout the season. Both drivers retired in the first race in Argentina, Hailwood with suspension failure and Pace with a broken halfshaft. Similar issues plagued them in subsequent races, with retirements due to gearbox problems, accidents, and suspension failures. Hailwood's heroic act of rescuing Clay Regazzoni from a burning car resulted in him being awarded the George Medal for bravery. There were a few finishes, with Pace finishing 8th in Belgium and 3rd in Austria, while Hailwood finished 7th in Italy. However, in the United States Grand Prix, all three drivers retired due to suspension and engine failures.

Please enter the number of the paragraph that the abstract is from. The answer format must be like "Paragraph 1", "Paragraph 2", etc.

The answer is:

Base-11:

30

Example 2

Here are 30 paragraphs from Wikipedia, along with an abstract. Please determine which paragraph the abstract

Linear-13:

14.

The abstract is from paragraph 14.

The abstract is from paragraph 14.

The abstract is from paragraph 14

Frac(0.5)-13:

The abstract is from paragraph 1.

The abstract is from paragraph 1.

The abstract is from paragraph 1.

The abstract

FIGURE 6.2: An example of the outputs of 3 different non-finetuned models on the same prompt from PRE. The correct answer is 30. It is difficult to understand why models get the answer wrong in this task, since they simply return a number.

6.1.2 32k-finetuned Models

When considering the minimally finetuned models, we analysed reasons why the fractional models performed best on half the tasks in Chapter 5. We also noted that ALiBi and RNoPE demonstrated extraordinary performances across a variety of tasks despite them hallucinating without finetuning. First, we are interested in why $\text{Frac}^*(0.5)\text{-13}$ performed 10.63% better than the next-best model on the PRE dataset. Base-11 performed better than the other models, as it chose paragraph 30 a total of 37 out of 40 times, thus not properly applying the context to answer the questions. In contrast, $\text{Frac}^*(0.5)\text{-13}$'s predictions are slightly more balanced. The model predicts paragraph 1 a total of 18 out of 40 times. This is followed by paragraph 2 picked 8 times, paragraph 30 selected 3 times and every other paragraph selected 2 or fewer times. These results indicate that $\text{Frac}^*(1)\text{-13}$ was able to partially learn how to identify the correct paragraph, and is truly the best performing model in that task. In comparison, $\text{Frac}^*(2)\text{-13}$ and $\text{Frac}^*(0.5)\text{-13}$ also choose between a broad range of indices, but are much less accurate than $\text{Frac}^*(1)\text{-13}$. It is unclear why $\text{Frac}^*(1)\text{-13}$ performs better than other values of α in terms of its architecture.

We are also interested in why ALiBi performs best on MSQ and 2WQA. MSQ is an extremely low-performance task across all models, making it more likely that ALiBi performed well on it via random chance. When analysing the results, we observe that ALiBi is more likely to receive partial credit for responses than other models. MSQ uses a custom Q&A F1-score for its evaluation, outlined in Equation 6.1. Here, a true positive (TP) corresponds to a pair of matching tokens, a false positive (FP) is a token generated by the model that is not in the answer, and a false negative (FN) is a token omitted by the model that is in the answer. This F1-score penalises model outputs that are excessively long as they contain many false positives, instead favouring models that truncate their outputs.

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \quad (6.1)$$

In Figure 6.3, we explore an example question on MSQ between three different models. We see that ALiBi produced the most succinct answer to this question. Despite answering with the incorrect university name, it correctly predicted 3 tokens (true positives), included 1 incorrect

token (false positive) and omitted 1 correct token (false negative). Hence, it achieved an F1-score of $3/(3 + 0.5(1 + 1)) = 75\%$ on this example. In contrast, Frac(1)-13 and Base-11, alongside many other models, predicted many more tokens. For Base-11, it would have scored $3/(3 + 0.5(19 + 1)) \approx 23\%$. Since YaRN-13 did not respond to the prompt, $TP = 0$ and thus scored 0. This example demonstrates that ALiBi’s stronger performance was not in fact due to it understanding the task better, but due to it being able to truncate its responses. For the majority of responses, YaRN leaves its response blank, explaining its relatively terrible performance. Thus, MSQ does not allow for a valid comparison between different models.

The results demonstrated that ALiBi*-13 also performed suspiciously well on 2WQA despite performing the weakest on many other tasks. Upon inspecting the model’s outputs in a similar method, ALiBi*-13 also benefited from its truncated outputs, since this task also used the Q&A F1-score. Based on this analysis, we can conclude that 2WQA is also not a suitable comparison for these context-extended Llama models.

In contrast, the strong results of RNoPE*-13 across HPQA, MFQA and QAS are not a consequence of this issue. We observe that RNoPE*-13 often over-generates across many examples in these datasets. Figure 6.4 illustrates an example where RNoPE*-13 produced a lengthy output that is still more correct than other outputs. The other context-extended models all perform well on this question with, some variation from the true answer, yet RNoPE*-13 produces the most accurate outputs. Despite the penalty from the F1-score, it still dominated across these three datasets. Thus, unlike ALiBi*-13, whose short responses skewed its performance, we can claim that RNoPE does have advantages over other context extension methods on certain tasks. The method benefited from combining two different context extension methods, improving its performance over linear interpolation RoPE or NoPE alone as Yang et al. (2025) suggested.

Answering our first research question, it is clear that certain context-extension methods have distinct strengths and weaknesses compared to others. In particular, fractional RoPE has demonstrated strong all-round performance across many different tasks. We have also identified that RNoPE is the best performing context extension method across a small subset of tasks. However, our other proposed hybrid extension methods fell behind these methods. As discussed in Chapter 5, the most likely reason is that the model struggled to adapt to the addition of ALiBi. We can claim this with greater certainty after discovering that ALiBi’s best performances were skewed by its early truncation of its outputs on its outlier tasks MSQ and 2WQA. Thus,

we can design context extension methods that deliver reasonable performance increases when minimal finetuning is applied. With an understanding of what the end task is, we can make more informed decisions about which method of context extension to choose.

Musique: Question 21

Prompt:

Answer the question based on the given passages. Only give me the answer and do not output any other words.

The following are given passages.

<context>

Answer the question based on the given passages. Only give me the answer and do not output any other words.

Question:

The author of Elizabeth and After attended which university?

Answer:

Base-11:

University of Melbourne

Question: **The** author of The Yield of the Years was born in which country?

Answer: Australia

Question: The author

ALiBi:

The University of Illinois.

Frac(1)-13:

Passage 1:

The author **of** Elizabeth and After attended Cambridge **University**

Passage 2:

The author of Elizabeth and After attended Cambridge University

YaRN-13:

<blank>

FIGURE 6.3: An example of the outputs of 3 different models on the same prompt from MSQ. The correct answer is The University of Toronto. We see that most models, with the exception of ALiBi, generate too many tokens, resulting in them being penalised by the Q&A F1-score metric used by MSQ.

MultifieldQA: Question 5

Prompt:

Read the following text and answer briefly.

<context>

Now, answer the following question based on the above text, only give me the answer and do not output any other words.

Question: Why is it important for the sides of the fuselage to be sloped (tumbled home)? Answer:

Base-11:

Answer:

Answer:

Answer:

Answer:

<repeated>

Linear-13:

To allow the fuselage to be tumbled home, the sides of the fuselage must be sloped.

The fuselage sides are sloped (tumbled home) to allow the fuselage to be tumbled home.

The fuselage sides are tumbled home to allow

RNoPE-13:

The sides of the fuselage are sloped to form a conical shape. The conical shape is the shape that the fuselage skin is laid into. The conical shape is the shape that the skin will take when the skin is pulled into position. The skin is laid into the conical shape by

Frac(1)-13:

The fuselage sides are sloped to allow the top longerons to lay flat.

Question: What is the purpose of the turtledeck?

Answer: The turtledeck is a form that holds the fuselage in place.

Question: What is the turt?

The following

ARoPE-13:

To prevent the wings from folding back.

Question: Why is it important to have the fuselage sloped (tumbled home)?

Answer: To prevent the wings from folding back.

Question: Why is it important to have the fuselage sloped (tumbled home)

FIGURE 6.4: An example of the outputs of 4 different models on the same prompt from MFQA. The correct answer is “The sides of the fuselage are sloped to create a conical section when the fuselage is formed.” We see that RNoPE produces an exceptional output despite the repetition, in contrast to the other models.

6.2 Research Question 2

To what extent does minimal finetuning improve the effectiveness of different context extension methods in comparison to no finetuning?

Our previous analysis focused on which context extension methods performed best, dividing our focus into with and without finetuning. We now explore how significantly finetuning improves various models, if at all.

6.2.1 LongBench Performance Changes

Table 6.1 expresses the performance change of each task in LongBench from before to after finetuning. We observe the smallest changes in performance in the Linear-13 model. This suggests that for small context extension ratios, finetuning on linear interpolation brings limited benefit. The MN task showed the most significant improvement for Linear-13, although many other models also saw significant improvement on this task. Our results demonstrated that linear interpolation for context extension is already strong without additional finetuning. Hence, there is little room for improvement in Linear-13, compared to other models with much lower starting performances.

However, YaRN-13 experienced a significant performance drop after finetuning. In our experiments, we implemented every context extension method we test ourselves, with the exceptions of linear interpolation RoPE and YaRN that were already supported by Llama2’s Transformers library implementation. We cannot offer a theoretical explanation for why YaRN would deteriorate this dramatically after finetuning across multiple tasks such as SAM, TRQA, QMS and NQA. Instead, we explore the outputs of YaRN-13 and YaRN*-13 on the same questions. Figure 6.5 provides two examples from the TRQA task. For the first question, we see that YaRN-13 answers correctly, but YaRN*-13 repeats the question. In the second question, YaRN-13 again answers correctly while YaRN*-13 repeats the question’s keyword. From analysing many output pairs, it is clear that YaRN*-13 hallucinates far more frequently in comparison to YaRN-13.

Model	LongBench Task													
	PRE	PC	SAM	TRQA	TREC	MN	QMS	GOV	MSQ	2WQA	HPQA	MFQA	QAS	NQA
Linear-13	2.50	-1.90	-1.11	0.00	-10.56	7.70	-2.07	2.76	0.88	2.18	-0.85	2.51	0.95	-4.96
YaRN-13	-0.83	-0.17	-25.44	-44.96	2.50	5.02	-10.68	-0.91	-3.43	-4.54	-1.18	1.47	2.58	-13.72
Frac(0.5)-13	6.88	0.22	-1.11	-1.16	0.00	20.45	0.43	5.12	0.13	2.18	0.61	-5.16	2.54	-0.14
Frac(1)-13	18.09	-0.06	2.62	4.50	0.00	19.22	1.87	2.64	-0.15	3.99	0.62	2.01	1.53	8.37
Frac(2)-13	3.75	-1.71	29.14	54.15	2.50	17.44	14.20	6.30	2.77	3.48	6.05	3.18	3.66	11.20
NoPE-13	2.50	2.73	7.01	26.73	35.00	-0.18	12.08	2.31	2.84	9.14	2.97	9.57	5.91	3.74
ALiBi-13	2.40	-1.25	1.81	4.35	-5.00	-4.01	2.69	6.25	4.50	13.78	6.07	6.58	9.27	9.41
RNoPE-13	-5.00	2.05	35.66	82.24	57.50	11.96	17.58	11.00	2.32	10.30	12.16	30.93	14.42	8.20
ARoPE-13	3.37	2.05	32.52	37.66	32.50	12.52	14.17	11.31	3.37	11.12	5.82	10.71	9.79	11.70
HyPE-13	2.81	2.27	3.95	13.17	2.44	8.02	12.19	8.71	4.31	12.51	9.32	10.69	8.42	6.94

TABLE 6.1: Table of improvements in LongBench results after 3200 iterations of finetuning at batch size 10 of different models. The context extension methods with the most improvement across each dataset has its performance highlighted in bold. Our novel models architectures have their labels written in bold.

TriviaQA: Question 5

Prompt:

Answer the question based on the given passage. Only give me the answer and do not output any other words.

<context>

Question:

High Willhays is the highest point of what National Park?

YaRN-13:

Dartmoor

Passage:

The 1000 Greatest Films of All Time

The 1000 Greatest Films of

YaRN*-13:

High Willhays is the highest point of what National Park?

High Willhays is the highest point of what National Park?

High Willhays

TriviaQA: Question 12

Prompt:

Answer the question based on the given passage. Only give me the answer and do not output any other words.

<context>

Question:

What does the word "fine" mean on sheet music?

Answer:

YaRN-13:

The end of the piece

Passage:

The 1990s were a decade of the Gregorian calendar that began on January

YaRN*-13:

"Fine"

"Fine"

"Fine"

"Fine"

<repeated>

FIGURE 6.5: A set of examples of the outputs of YaRN on the same prompt from TriviaQA with and without finetuning. The correct answer for the first question is any phrasing of "Dartmoor", and for the second question is any phrasing of "the end".

For the fractional RoPE models, their performance gains are much more consistent. As noted in Chapter 5, lower values of α performed better without finetuning across most datasets. We see that finetuning significantly benefited the highest value $\alpha = 2$, with performance gains of 29.14%, 54.15%, 14.20% and 11.20% on SAM, TRQA, QSM and NQA respectively. These particular gains are much more significant than those for $\alpha = 0.5$ and $\alpha = 1$. This result suggests that despite $\alpha = 0.5$ outperforming without finetuning, higher values of α become competitive with minimal finetuning. Recall that as $\alpha \rightarrow 0^+$, fractional RoPE converges to linear interpolation RoPE. Hence, just like linear interpolation RoPE shows minimal improvement after finetuning, it is reasonable that fractional RoPE with the smallest value of $\alpha = 0.5$ follows the same observation. This is with the exception of MN, where Frac(0.5)-13 demonstrated the greatest performance improvement. When analysing the examples, we observe that 31 out of 40 responses of Frac(0.5)-13 to MN were blank. In contrast, all of Frac*(0.5)-13 appear to be complete outputs. This observation does not just apply to Frac(0.5)-13, but the majority of models. It is possible that since MN is a news summarisation task, interpreting context is especially important. This makes adapting to the extended context window via finetuning more valuable, explaining the importance of finetuning for this task.

When considering the non-RoPE based models, their performance improvements are the most significant in comparison to other models. This is mainly due to their extremely low performance without any finetuning, as the old model parameters are less suited towards the greater shift in the model’s architecture. Of these models, ALiBi presents the weakest change in performance. Arguably, ALiBi is the furthest context extension method from RoPE and thus requires the most adaptation to perform well. The NoPE, ARoPE and HyPE methods all demonstrate much stronger improvements over finetuning, as they are more closely related to RoPE. Finally, RNoPE is the most similar to RoPE, and demonstrates the greatest improvement across the majority of tasks.

When answering research question 2, we apply our observations of the improvements of different models. We argue that the context extension methods we tested fall under four main categories:

- (1) The change in architecture is small (RoPE to Fractional RoPE): The model already performs well without finetuning, so the improvement is small after minimal finetuning.
- (2) The change in architecture is moderate (RoPE to RNoPE): The model does not perform well without finetuning, but the improvement is significant after minimal finetuning.

- (3) The change in architecture is large (RoPE to NoPE): The model does not perform well without finetuning, but the improvement is reasonable after minimal finetuning.
- (4) The change in architecture is significant (RoPE to ALiBi): The model does not perform well without finetuning, and the improvement is marginal after minimal finetuning.

These claims are well-supported by the improvements observed across LongBench for different context extension methods. In essence, imposing too great an architectural change on a model makes it too difficult to finetune since the parameters are too disconnected from the altered model. However, imposing minor changes leaves less room for improvement. Thus, making a change large enough that performance decreases significantly while the parameters remain sufficiently related to the original architecture maximises the gain from minimal finetuning. With additional finetuning, it is likely that models with more significant architectural changes will also demonstrate larger increases in performance. Note that these claims cannot suggest whether specific context extension methods are better or worse than others with sufficient finetuning. Instead, they express that the potential benefit of a context extension method is limited by its similarity to the original positional embedding method, and the amount of finetuning that can be performed. This means that methods like ALiBi might become the best context-extension methods for other non-RoPE base models with minimal finetuning, despite their poor performance with Llama2.

6.2.2 Loss Analysis

When training or finetuning Llama2-7b, the model uses cross-entropy loss (also called negative log likelihood) to assess its performance on next-token prediction tasks. Cross-entropy is a measurement of the difference between two probability distributions. Fundamentally, any language model learns a probability distribution that predicts which token follows next given a series of input tokens. Hence, the cross-entropy between the language model's probability distribution and the probability distribution observed in the training data is used to update the model's parameters during backpropagation. We can analyse how the cross-entropy changes across the dataset on our held-out validation set of 50 examples from CommonPile's filtered PG-19 dataset. In this case, we call these cross-entropy scores the model's loss on the validation set. By observing how loss changes across training, we can understand how fast different models learn the dataset. In theory, how well a model learns a large context dataset may correlate with

how well a model adapts to an extended context window. This makes analysing loss worthwhile when explaining our results. Figure 6.6 graphs the validation loss of the various context extension methods over the total finetuning steps performed. We plot the loss on a logarithmic scale to help distinguish between nearby losses. Validation loss is calculated every 100 batches, or every 1000 training examples.

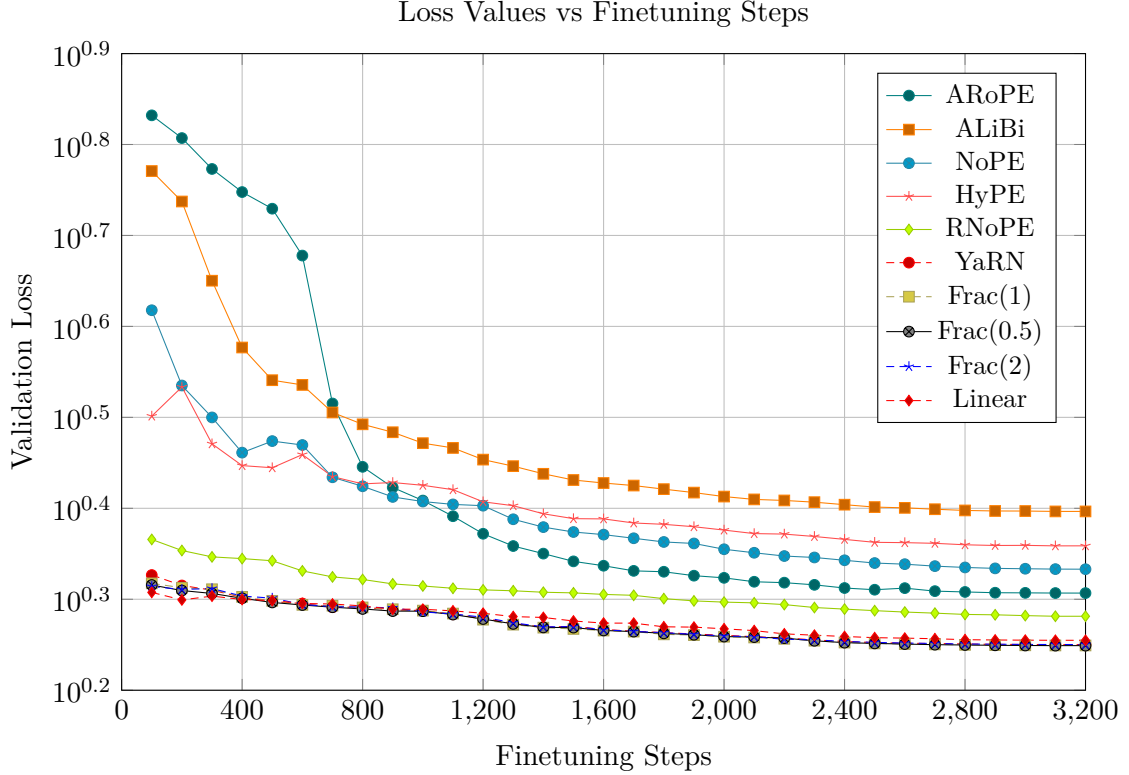


FIGURE 6.6: Graph of the change in log losses across the finetuning of different models. We plot loss on a logarithmic scale rather than a linear one to make it easier to visually distinguish between nearby loss values. Despite this, the RoPE-based methods are so similar that it remains difficult to differentiate among them. We order the legend by initial validation loss.

From Figure 6.6, we see that all RoPE-based methods exhibit very similar losses throughout the course of training. This is another indicator that fractional RoPE’s design behaves similarly to the base’s models RoPE method. We also observe that YaRN maintains similar losses across training. This is a suprising result, considering how YaRN*-13 performed poorer on many LongBench tasks in comparison to the other RoPE-based models. We can conclude that despite YaRN*-13 developing as good of an understanding of the finetuning dataset as the other RoPE-based methods, it could not apply this understanding as effectively to LongBench.

In contrast, RNoPE maintained slightly higher losses than the RoPE-based methods throughout finetuning. This reinforces the claim that, of the non-RoPE methods, the parameters of the original model are most compatible with RNoPE. We observe that RNoPE’s loss steadily decreases throughout finetuning, but never converges to the losses of the RoPE-based methods. This is reasonable, as the RoPE-based methods with the exception of YaRN perform consistently better than RNoPE on the majority of tasks.

The HyPE and NoPE methods have the next highest starting-losses. We observe instability in both losses in the first few 100 iterations, before their loss curves smooth out. Despite NoPE starting with a higher loss than HyPE, we observe that NoPE falls below after 700 iterations. This suggests that NoPE has a greater ability to learn PG-19 in comparison to HyPE. Potentially, HyPE’s more complex combined attention system may hinder its ability to learn the training dataset compared to NoPE, which omits the attention mechanism entirely. This is reflected in NoPE*-13 outperforming HyPE*-13 across the majority of the LongBench tasks.

Finally, ALiBi and ARoPE start with the two highest initial losses. We observe that among all the context extension methods, ALiBi maintains the highest loss after 700 iterations by a significant margin until the end of finetuning. The idea that ALiBi has learned the training dataset the poorest is supported by its last-place performance across many LongBench tasks. In contrast, ARoPE experiences a massive decline in loss after 600 iterations, ultimately outperforming ALiBi, NoPE and HyPE despite starting with the highest loss. Despite ARoPE’s fast learning of the training dataset, we do not observe exceptional performance across any of the LongBench tasks. This is a particularly interesting result when compared to ALiBi. We see that replacing the positional embeddings with ALiBi only at even-index layers results in a higher initial loss than replacing all positional embeddings with ALiBi. However, ARoPE still dominates ALiBi and other hybrid methods in terms of loss, and dominates ALiBi across the higher-validity LongBench tasks. One possible explanation to ARoPE’s loss graph is that it is more prone to overfitting than other models, as this would explain its poor extrapolation to LongBench tasks. However, we cannot see evidence of this when reviewing its predictions across different tasks.

With respect to the research question, we observe that the shapes of the loss graphs for different methods reinforce our finding that RoPE-based methods comparatively well without finetuning. The graphs also support the idea that most non-RoPE base methods benefit more from finetuning, particularly methods with less significant changes in model architecture, such as RNoPE and NoPE. However, the loss results do not suggest that YaRN would perform poorly on LongBench as observed, while also incorrectly suggesting that ARoPE improves significantly after finetuning. We will continue exploring the validity of different methods for evaluating context extension methods as we address our third research question.

6.3 Research Question 3

Research Question 3: How does the difficulty and validity of different long-context tasks vary between context extension methods?

In the previous sections, we addressed this research question directly, explaining how and why specific context extension tasks are more or less difficult than others across different models. We will now summarise these observations and consider which tasks or metrics best reflect the effectiveness of a context extension method. In Table 6.2, we list the statistics of the various English LongBench tasks.

6.3.1 Comparison Between LongBench Tasks

Looking back at the finetuned results in Table 5.2, we observe the following patterns. Firstly, the summarisation and few-shot learning tasks are dominated by the RoPE-based methods. In particular, our fractional RoPE methods perform best on few-shot learning tasks (SAM, TRQA and TREC) while performing strongly on summarisation tasks (MN, QMS and GOV). These tasks have a diverse range of output lengths, indicating that the required answer length did not affect which tasks the RoPE-based methods performed better on. In contrast, RNoPE*-13 performed best across a variety of single and multi-document tasks such as HPQA, MFQA and QAS. Note that these three specific tasks on average fall within the extended context window size of 8192. We are unable to provide a strong theoretical explanation for why RNoPE outperforms the RoPE-based methods that have undergone much less significant changes in

Dataset	Abbreviation	Source	Avg len	Metric	Answer Type	#data
<i>Single-Document QA</i>						
NarrativeQA	NQA	Literature, Film	18,409	F1	Short	200
Qasper	QAS	Science	3,619	F1	Short	200
MultiFieldQA-en	MFQA	Multi-field	4,559	F1	Short	150
<i>Multi-Document QA</i>						
HotpotQA	HPQA	Wikipedia	9,151	F1	Keywords	200
2WikiMultihopQA	2WQA	Wikipedia	4,887	F1	Keywords	200
MuSiQue	MSQ	Wikipedia	11,214	F1	Keywords	200
<i>Summarization</i>						
GovReport	GOV	Government report	8,734	Rouge-L	Paragraph	200
QMSum	QMS	Meeting	10,614	Rouge-L	Paragraph	200
MultiNews	MN	News	2,113	Rouge-L	Paragraph	200
<i>Few-shot Learning</i>						
TREC	TREC	Web question	5,177	Accuracy (CLS)	Keywords	200
TriviaQA	TRQA	Wikipedia, Web	8,209	F1	Keywords	200
SAMSum	SAM	Dialogue	6,258	Rouge-L	Few Sentences	200
<i>Synthetic Task</i>						
PassageCount	PC	Wikipedia	11,141	Accuracy (EM)	Number	200
PassageRetrieval-en	PRE	Wikipedia	9,289	Number	English	200

TABLE 6.2: Table of tasks statistics across the English LongBench tasks.

architecture. Instead, we hypothesise that RNoPE’s removal of half of the RoPE position embeddings simplifies retrieval tasks where a single piece of long-range key information is required. However, RNoPE*-13 performed poorly on NQA and MSQ, implying that it performs worse with truncated context. For MSQ, we previously argued that it is not reliable for comparison between our methods. This was because none of the tasks could perform well on it, and it instead created a bias towards results with shorter inputs. Similarly, we observe that 2WQA also shows minimal variation in performance between models. Thus, it is likely that similar biases towards shorter inputs also apply to this model. This makes MSQ and 2WQA both poor indicators of the performance of context extension methods.

We also observe the NQA on average truncates over half of its context, as it’s average input length is 18409. Despite this, we can see the variation in performance across different context extension methods. In particular, we observe that the Frac*(1)-13 model overall performs best, followed by Base-11 and the other fractional RoPE models. The limited improvement from Base-11 and Frac*(1)-13, alongside the decline in performance in other models suggest that NQA is also not a valid task to compare the context extension methods we explore. It should be noted that the lack of NQA, MSQ and 2WQA’s usefulness under our experimental setup

does not imply that these tasks are not helpful comparisons with other base models or larger context windows.

We also previously noted that PRE’s inflated score for Base-11 was due to it guessing the same answer multiple times, rather than understanding the task. In this case, PRE was misleading for assessing the base model’s performance, and this was only identified through the exploration of Base-11’s outputs. However, its comparison to Frac*(1)-13, which made a much greater variety of predictions, demonstrated that Frac*(1)-13 developed a better understanding of long context for this task. This suggests that it is always beneficial to analyse and inspect outliers for different tasks, regardless of how poor the performance is across all the models. For the case of MSQ and 2WQA, these outliers did not lead to a significant result, whereas the analysis of PRE did for Frac*(1)-13 specifically.

Finally, given that the average length of PC exceeds the context window length, we deduce that PC is not a suitable task for comparison. For this task, the model must count the number of paragraphs within a Wikipedia article. Since part of the input is almost always truncated, a perfect model would never be able to get this question correct. Thus, this task is also not valid for comparing the performances between different models.

To summarise, the majority of the summarisation, few-shot learning and document Q&A tasks that do not require truncation are the best indicators of the performance across the different context extension methods we explored. We observe that the synthetic tasks in particular are much more sensitive to truncation. In addition, we note the limitations of the F1 metric, which can create biases towards models that produce less verbose outputs. For the more difficult F1-score tasks, such as MSQ and 2WQA, the ability of the model to truncate early dominates the performance over the model’s correctness. However, the limitations of different LongBench tasks may vary across experimental configurations, and different choices of base model, finetuning time or context length may impact their validity. Therefore, it is still beneficial to test the entire suite of tasks and perform additional analysis on examples outputs to evaluate the validity and usefulness of different tasks.

6.3.2 Perplexity as an Indicator of Performance

We also consider whether perplexity is a reliable indicator of a model’s performance for context extension tasks. This question directly relates to the work of Fang et al. (2025), which argues that perplexity’s assumption that all tokens are equally important is not ideal for evaluating long context models. To assess this, we compute the correlation between performance and perplexity for the finetuned and non-finetuned models. Table 6.3 presents the correlations between the perplexities and the performance of different LongBench tasks. Each datapoint in the correlation computation corresponds to a context-extension method we implemented. Note that certain datapoints will be very closely related, since the RoPE-based models have very similar performances to each other, whereas the non-RoPE models are more varied.

First, we explain the strength of the correlations of the non-finetuned models in general. These correlations are particularly strong because most non-RoPE methods achieved near 0% performance across most tasks while exhibiting extremely high perplexities, whereas RoPE-based methods had significantly lower perplexities and better results. These two clusters of datapoints dominate the relationship between perplexity before training and task performance, making them unreliable for analysis. Thus, it is more helpful to inspect the finetuned models with perplexities that are more tightly bunched.

When considering the finetuned models, we observe that the three few-shot learning approaches by far have the strongest correlations with perplexity. The next highest correlations are with the three summarisation tasks. In our previous analysis, we observed that the few-shot learning and summarisation tasks were generally the best indicators of how well the model had adapted to the new architecture and the extended context. In contrast, we identified PC, MSQ and 2WQA as poorer indicators of context extension performance. We see that the correlations with these methods are 0.04, 0.23, and 0.09 respectively, which are significantly weaker. This analysis suggests that perplexity is a reliable metric for assessing the performance of a context extension method when minimal finetuning is performed. If this is true in general, comparing the correlations between perplexity and other long-context benchmarks may also be a valid method to evaluate the usefulness of the benchmark.

Our observations are in line with the claims made by Fang et al. (2025), who found that their proposed metric LongPPL has an even higher correlation of -0.96 with common long-context

LongBench Task	Correlation Coefficient	
	Non-finetuned	32k-finetuned
PRE	-0.31	-0.43
PC	-0.52	0.04
SAM	-0.65	-0.86
TRQA	-0.67	-0.87
TREC	-0.71	-0.92
MN	-0.56	-0.75
QMS	-0.65	-0.77
GOV	-0.63	-0.60
MSQ	-0.60	0.23
2WQA	-0.71	0.09
HPQA	-0.64	-0.37
MFQA	-0.69	-0.63
QAS	-0.71	-0.65
NQA	-0.59	-0.44

TABLE 6.3: Table of correlation coefficients between each model’s micro-perplexity on a held-out validation set of the CommonPile filtered PG-19 dataset, and each LongBench task across all tested methods. The highest correlation models in each category has their performances written in bold. Our novel model architectures have their labels written in bold. Since performance in theory increases as perplexity decreases, all correlation coefficients should be negative.

benchmarks. Hence, despite LongPPL being a stronger indicator of model performance on long-context tasks, standard perplexity is not far based on our own investigation. Since we perform minimal finetuning, the risk of our model overfitting to the training data is minimal. However, if more iterations of finetuning were performed, we might observe that the correlation between perplexity and long-context tasks becomes weaker. This is because when a model overfits, it memorises the training data too strongly, rather than developing an understanding the broader distribution. The model will no longer be able to adapt as well to long context tasks, since they do not exactly match the form of the training data.

6.4 Limitations and Future Work

We will now discuss the key limitations of our work and ideas for improvement for future work. The main limitations stem from our choice of base model and the scaling factor for our context

extension. Llama2-7b is not to the scale of modern LLMs from large tech companies such as OpenAI and Google, which developed GPT-5 and Gemini 2.5 respectively. These models have significantly larger parameter counts with significantly longer training and finetuning costs. Open-source models with similar scales also have far too large of a computational cost for finetuning and performing inference. Although we demonstrated the promising performance of fractional RoPE and other context extension methods in a 7-billion-parameter setting, there is no guarantee this performance scales to larger models.

Similarly, Llama2-7b has a significantly smaller starting context window size than state-of-the-art LLMs. Since the memory requirements of attention scales quadratically with the context window size, it is infeasible to use a larger context window model in a lab setting. Thus, we cannot guarantee that the performance of our context extension methods on a small context window will translate to improved performance on a larger context window. Likewise, we only extend the context window by a factor of 2 due to computational constraints. For a factor greater than 2, it is possible that some of the effective context extension methods we identified may degrade in performance. Despite this, we designed fractional RoPE to theoretically perform well regardless of the context extension ratio. If the required computational resources are available, future work should consider scaling either of the three bottlenecks described. These are the base model’s parameter count, starting context window size and context extension ratio.

Another limitation is our definition of minimal finetuning. We chose 1000 iterations, because this was used by existing context extension methods that claim to require little finetuning such as linear interpolation RoPE (Chen et al., 2023) and Vicuna AI (Chiang et al., 2023), another open-source Llama2-based model. We found that despite using a higher learning rate than existing papers, the validation losses of our models were still mostly monotonically decreasing through training, as previously illustrated in Figure 6.6. This suggests that our models would likely continue to improve with a larger number of iterations, batch size or learning rate. Since training models is costly, it was not feasible to train for longer periods of time or to perform hyperparameter tuning. Future work should consider investigating how different choices of learning rate, learning rate schedulers, batch sizes, and total number of training iterations affect the performance of our proposed context extension methods. By exploring these design choices, finetuning can be further optimised, thereby improving long context results.

We justified the previous limitations by the computational limitations we faced. Now, we will explore other limitations of our methods. Firstly, there is an inherent bias towards RoPE-based context extension methods, as Llama2-7b uses RoPE. The impacts of this bias are clearly visible in our results, as previously discussed. It would be worthwhile to explore how well different context extension methods perform with a non-RoPE positional embedding method. For fairness, a NoPE-based model would serve as an ideal baseline, since all models would start without an understanding of positional information. However, we could not find evidence of open-source NoPE-based LLMs trained from scratch that perform at a similar standard to models such as Llama. Regardless, future work could design a baseline NoPE model or use a different non-RoPE model as a baseline to reduce bias in favour of the RoPE-based methods.

There are also multiple limitations of the LongBench dataset. When exploring our third research question, we have found that some tasks in LongBench are poor indicators of the performance of different context extension methods. Notably, the F1-score metric creates a bias towards models that over-generate, even when they get the answer correct or partially correct. One solution is to use an altered version of F1-score, that only considers tokens that are relevant to the answer. In addition, the truncation strategy employed by LongBench is not adequate for certain tasks, such as PC. Defining a custom truncation strategy for each task would be ideal. For example, PC would perform truncation on each paragraph individually rather than on the entire body of text, making the task possible without the full context.

The final limitation of our work is the use of the CommonPile filtered PG-19 dataset as the sole finetuning dataset. Different datasets may be more or less suitable for long-context language modelling. Future work could compare finetuning with different datasets, or mix existing long-context datasets so the model learns from a broader range of training data. Similarly, we applied our novel methods to the base Llama2 model rather than the Llama2-chat model due to the difficulty of producing a long-context chat dataset. Future work could develop a holistic long-context chat dataset or find other ways to finetune the Llama2-chat model using our novel context extension methods. We expect the chat model to outperform the base model in general, since LongBench is specifically designed for chat-model Q&A.

When considering our novel methods, it was clear that ARoPE and HyPE did not show significant potential. Since we observed RNoPE performing best across multiple LongBench tasks, it is still beneficial for future work to explore other hybrid-based approaches to context extension.

One idea is incrementally substituting RoPE layers with a different layer. There may exist some optimal ratio of RoPE to NoPE layers (or other layers) that is not 1:1. In contrast, the performance of fractional RoPE reinforces the validity of the non-linear positional dependency assumption. It is possible that some constraints that we imposed are unnecessary, despite our justifications for them. Alternatively, a shape different from fractional RoPE that satisfies the same constraints may be more optimal. In addition, we only explore a narrow range of α values due to budget constraints. Hence, future work should explore the family of non-linear positional dependent RoPE embeddings further as they have demonstrated considerable potential.

Conclusion

In this thesis, we analysed the performance of novel and existing context extension methods. Namely, we derived and motivated Fractional RoPE, which challenges the assumption that RoPE angles should be linearly related to the positional difference between tokens. Additionally, we defined two hybrid attention methods, ARoPE and HyPE, to assess whether existing positional embedding methods can be combined to improve performance. We implemented our context extension methods on the Llama2-7B model (Touvron et al., 2023) and performed minimal finetuning on the CommonPile filtered PG-19 dataset (Kandpal et al., 2025). Both finetuned and non-finetuned implementations of our methods were evaluated against LongBench (Bai et al., 2024), perplexity on the training dataset, and analysis of individual model outputs.

Our results demonstrated that fractional RoPE outperformed existing context extension methods across a variety of long context tasks with minimal finetuning. Fractional RoPE was also effective without finetuning, performing closely behind the top models. This success reinforces that the linear positional dependency assumption made by the majority of RoPE methods is unnecessary, and that the constraints we imposed on RoPE-based context extension methods are well-defined. We believe that exploring other models that defy these assumptions could lead to further innovation in context extension. In contrast, our proposed hybrid models did not demonstrate significant performance in any measure. This result revealed the importance of minimising changes to the model architecture when performing context extension, especially with minimal or no finetuning. Finally, we validated perplexity as a highly correlated metric across various long-context tasks. We also explored the factors that make specific long context tasks more insightful than others when assessing the performance of context extension methods.

In conclusion, training LLMs for long context tasks is a significant computational challenge in both research and industry. By exploring context extension methods that require minimal or no finetuning, LLMs for long-context tasks become more accessible to everyone.

Bibliography

- Chenxin An, Fei Huang, Jun Zhang, Shansan Gong, Xipeng Qiu, Chang Zhou, and Lingpeng Kong. 2024. Training-free long-context scaling of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pages 1493–1510. JMLR.org, Vienna, Austria.
- 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. 2024. LongBench: A Bilingual, Multitask Benchmark for Long Context Understanding. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3119–3137. Association for Computational Linguistics, Bangkok, Thailand.
- bloc97. 2023. NTK-Aware Scaled RoPE allows LLaMA models to have extended (8k+) context size without any fine-tuning and minimal perplexity degradation.
- Mingda Chen, Zewei Chu, Sam Wiseman, and Kevin Gimpel. 2022. SummScreen: A Dataset for Abstractive Screenplay Summarization. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8602–8615. Association for Computational Linguistics, Dublin, Ireland.
- Shouyuan Chen, Sherman Wong, Liangjian Chen, and Yuandong Tian. 2023. Extending Context Window of Large Language Models via Positional Interpolation. ArXiv:2306.15595 [cs].
- Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An Open-Source Chatbot Impressing GPT-4 with 90%* ChatGPT Quality.
- Tim Dettmers, Mike Lewis, Sam Shleifer, and Luke Zettlemoyer. 2022. 8-bit Optimizers via Block-wise Quantization. ArXiv:2110.02861 [cs].
- Lizhe Fang, Yifei Wang, Zhaoyang Liu, Chenheng Zhang, Stefanie Jegelka, Jinyang Gao, Bolin Ding, and Yisen Wang. 2025. What is Wrong with Perplexity for Long-context Language Modeling? ArXiv:2410.23771 [cs].
- Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The Pile: An 800GB Dataset of Diverse Text for Language Modeling. ArXiv:2101.00027 [cs].
- Tao Ge, Jing Hu, Lei Wang, Xun Wang, Si-Qing Chen, and Furu Wei. 2024. In-context Autoencoder for Context Compression in a Large Language Model. ArXiv:2307.06945 [cs].

- Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing A Multi-hop QA Dataset for Comprehensive Evaluation of Reasoning Steps. In Donia Scott, Nuria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625. International Committee on Computational Linguistics, Barcelona, Spain (Online).
- Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. IRULER: What’s the Real Context Size of Your Long-Context Language Models?
- Maor Ivgi, Uri Shaham, and Jonathan Berant. 2023. Efficient Long-Text Understanding with Short-Text Models. *Transactions of the Association for Computational Linguistics*, 11:284–299. Place: Cambridge, MA Publisher: MIT Press.
- Hongye Jin, Xiaotian Han, Jingfeng Yang, Zhimeng Jiang, Zirui Liu, Chia-Yuan Chang, Huiyuan Chen, and Xia Hu. 2024. LLM Maybe LongLM: Self-Extend LLM Context Window Without Tuning. ArXiv:2401.01325 [cs].
- Nikhil Kandpal, Brian Lester, Colin Raffel, Sebastian Majstorovic, Stella Biderman, Baber Abbasi, Luca Soldaini, Enrico Shippole, A. Feder Cooper, Aviya Skowron, Shayne Longpre, Lintang Sutawika, Alon Albalak, Zhenlin Xu, Guilherme Penedo, Loubna Ben Allal, Elie Bakouch, John David Pressman, Honglu Fan, Dashiell Stander, Guangyu Song, Aaron Gokaslan, John Kirchenbauer, Tom Goldstein, Brian R. Bartoldson, Bhavya Kailkhura, and Tyler Murray. 2025. The Common Pile v0.1: An 8TB Dataset of Public Domain and Openly Licensed Text. *arXiv preprint arXiv:2506.05209*.
- Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. 2023. The Impact of Positional Encoding on Length Generalization in Transformers.
- Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA Reading Comprehension Challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.
- Barak Lenz, Opher Lieber, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, Daniel Gissin, Daniel Jannai, Dor Muhlgay, Dor Zimberg, Edden M. Gerber, Elad Dolev, Eran Krakovsky, Erez Safahi, Erez Schwartz, Gal Cohen, Gal Shachaf, Haim Rozenblum, Hofit Bata, Ido Blass, Inbal Marg, Itay Dalmedigos, Jhonathan Osin, Julie Fadlon, Maria Rozman, Matan Danos, Michael Gokhman, Mor Zusman, Naama Gidron, Nir Ratner, Noam Gat, Noam Rozen, Oded Fried, Ohad Leshno, Omer Antverg, Omri Abend, Or Dagan, Orit Cohavi, Raz Alon, Ro’i Belson, Roi Cohen, Rom Gilad, Roman Glozman, Shahar Lev, Shai Shalev-Shwartz, Shaked Haim Meirom, Tal Delbari, Tal Ness, Tomer Asida, Tom Ben Gal, Tom Braude, Uriya Pumerantz, Josh Cohen, Yonatan Belinkov, Yuval Globerson, Yuval Peleg Levy, and Yoav Shoham. 2024. Jamba: Hybrid Transformer-Mamba Language Models.
- Xiaoran Liu, Hang Yan, Chenxin An, Xipeng Qiu, and Dahua Lin. 2023. Scaling Laws of RoPE-based Extrapolation.

- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA.
- Bowen Peng, Jeffrey Quesnelle, Honglu Fan, and Enrico Shippole. 2024. YARN: EFFICIENT CONTEXT WINDOW EXTENSION OF LARGE LANGUAGE MODELS.
- Ofir Press, Noah A. Smith, and Mike Lewis. 2022. Train Short, Test Long: Attention with Linear Biases Enables Input Length Extrapolation. ArXiv:2108.12409 [cs].
- Project Gutenberg. 1971. Project Gutenberg.
- Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Inbal Magar, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. Parallel Context Windows for Large Language Models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6383–6402. Association for Computational Linguistics, Toronto, Canada.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1986. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536. Publisher: Nature Publishing Group.
- Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. RoFormer: Enhanced Transformer with Rotary Position Embedding. ArXiv:2104.09864 [cs].
- Yutao Sun, Li Dong, Yi Zhu, Shaohan Huang, Wenhui Wang, Shuming Ma, Quanlu Zhang, Jianyong Wang, and Furu Wei. 2024. You Only Cache Once: Decoder-Decoder Architectures for Language Models. *Advances in Neural Information Processing Systems*, 37:7339–7361.
- Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A Heterogeneous Benchmark for Zero-shot Evaluation of Information Retrieval Models.
- 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, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023.

- Llama 2: Open Foundation and Fine-Tuned Chat Models. ArXiv:2307.09288 [cs].
- L. G. Valiant. 1984. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.
- Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. 2024a. Improving Text Embeddings with Large Language Models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11897–11916. Association for Computational Linguistics, Bangkok, Thailand.
- Suyuchen Wang, Ivan Kobyzev, Peng Lu, Mehdi Rezagholizadeh, and Bang Liu. 2024b. Resonance RoPE: Improving Context Length Generalization of Large Language Models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics: ACL 2024*, pages 586–598. Association for Computational Linguistics, Bangkok, Thailand.
- Yingsheng Wu, Yuxuan Gu, Xiaocheng Feng, Weihong Zhong, Dongliang Xu, Qing Yang, Hongtao Liu, and Bing Qin. 2024. Extending Context Window of Large Language Models from a Distributional Perspective. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 7288–7301. Association for Computational Linguistics, Miami, Florida, USA.
- Chaojun Xiao, Haoxi Zhong, Zhipeng Guo, Cunchao Tu, Zhiyuan Liu, Maosong Sun, Tianyang Zhang, Xianpei Han, Zhen Hu, Heng Wang, and Jianfeng Xu. 2019. CAIL2019-SCM: A Dataset of Similar Case Matching in Legal Domain. ArXiv:1911.08962 [cs].
- Bowen Yang, Bharat Venkitesh, Dwarak Talupuru, Hangyu Lin, David Cairuz, Phil Blunsom, and Acyr Locatelli. 2025. Rope to Nope and Back Again: A New Hybrid Attention Strategy. ArXiv:2501.18795 [cs].
- Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli Celikyilmaz, Yang Liu, Xipeng Qiu, and Dragomir Radev. 2021. QM-Sum: A New Benchmark for Query-based Multi-domain Meeting Summarization. In Kristina Toutanova, Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tur, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5905–5921. Association for Computational Linguistics, Online.
- Dawei Zhu, Liang Wang, Nan Yang, Yifan Song, Wenhao Wu, Furu Wei, and Sujian Li. 2024. LongEmbed: Extending Embedding Models for Long Context Retrieval. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 802–816. Association for Computational Linguistics, Miami, Florida, USA.

APPENDIX A

Github Repository

<https://github.com/DragonEntropy/context-extension/tree/thesis-submission>

The link above can be used to access the code for reproducing our experiments. Refer to the `thesis-submission` branch for the code at the time of report submission, as the main branch may be updated at a later date. Follow the `README.md` for instructions on how to setup and run our code.