Pushing the Limits of the Compressive Memory Introduced in Infini-Attention

Architectural Decisions for Language Modelling with (Small) Transformers

Lauri Kesküll

Supervisor(s): Prof.dr. A. van Deursen, Prof. Maliheh Izadi, Aral De Moor

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 23, 2024

Name of the student: Lauri Kesküll
Final project course: CSE3000 Research Project
Thesis committee: Prof. Thomas Abeel, Prof.dr. A. van Deursen, Prof. Maliheh Izadi, Aral De Moor

Abstract

Transformers are a type of neural network architecture used in natural language processing. They excel in tasks such as translation, text generation, and language modeling by capturing long-range dependencies. Increasing input sequence length enhances performance but at a high computational cost. This study investigates the effectiveness of Infini-attention, a proposed solution to mitigate these costs, and explores its integration strategies. We implemented and trained Infini-attention on the GPT-NEO platform and TinyStories dataset, evaluating on the BabyLM pipeline. Our findings reveal the optimal strategy for integrating Infini-attention.

1 Introduction

Transformers [26] have revolutionized natural language processing, yet they grapple with limitations thanks to fixed-size context windows, restricting their capacity to process lengthy documents and understand dependencies outside these windows. Previously proposed model architectures such as Transformer-XL [7] and BigBird [33] have addressed these issues through sparse attention mechanisms and cacheing of past Key-Value matrices. However, challenges persist in terms of scalability and especially performance [25]. For instance, BigBird struggles with certain tasks that standard attention mechanisms can easily handle, such as finding the most distant point for each point in a sequence [33]. Similarly, Transformer-XL improves efficiency but suffers from a lack of gradient flow into previous segments, impacting its ability to utilize the extended context effectively [7].

In response to these ongoing challenges, Munkhdalai et al. have introduced the concept of Infini-attention [17], an innovative approach for efficient management of infinitely long contexts. Infini-attention integrates a compressive memory [12] into the transformer architecture, allowing for a dynamic and scalable handling of extended contexts with a linear instead of quadratic increase in computational demands. This mechanism enables continuous contextual understanding over extended sequences.

While Infini-attention has demonstrated promising results in fine-tuning scenarios, this study explores whether the architecture becomes even more powerful when integrated at earlier stages of training. Our approach involves evaluating the performance of Infini-attention when integrated during pre-training versus fine-tuning and pinpointing the optimal amount of fine-tuning required for successful integration of Infini-attention. Additionally, we empirically test the feasibility of using shortened local context lengths in conjunction with Infini-attention in order to determine how effectively compressive memory can alleviate the problems that arise with shortened context length.

These experiments yield valuable insights for optimizing transformer configurations, particularly in scenarios where extending the context window is resource-constrained. One example of such a domain is local devices like laptops and smartphones. Having language models (LMs) available locally would aid in the insurance of privacy of consumers and greatly reduce latency. Additionally, improved sample efficiency is critical as we face a scarcity of new data for model training. Using the data we have available to us efficiently is highly important for future advancements.

To align our experiments to use small and sample-efficient models, we utilize the TinyStories dataset [9] and a small language model, GPT-NEO [4], enhanced with rotary embeddings [24]. This experimental setup aims to uncover the full potential of Infini-attention and compressive memory, guiding the development of future transformer models that are both data-efficient and powerful.

Our contributions are as follows:

- Concluded whether Infini-Attention should be incorporated during pre-training or fine-tuning.
- Provided insights into the integration process of Infini-attention by analyzing the convergence behavior of gating parameters during fine-tuning.
- Created a replication package[1] for reproducing our findings, our models[2] published on HuggingFace, and our modified `transformers` library[3] to work with segment-level Infini-Attention.

2 Background

Transformers' self-attention mechanism incurs quadratic costs as sequence length increases. This section examines the origins and implications of this issue and reviews solutions that aim to address these limitations.

2.1 The Quadratic Cost of Self-attention

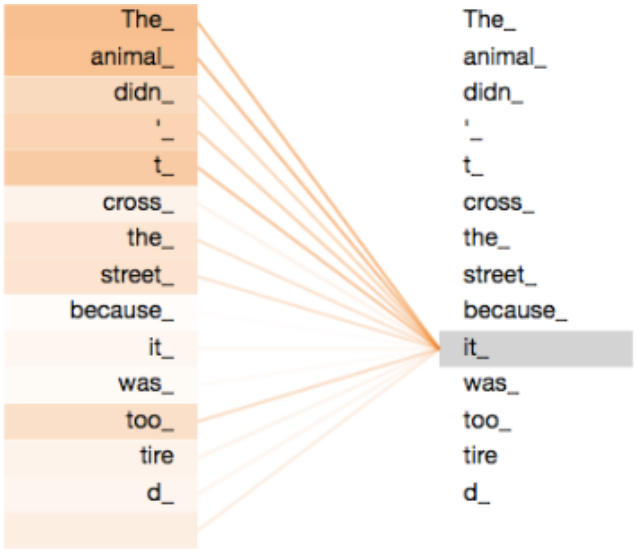

Figure 1: Attention scores of a single token visualized. Taken from Alammar [1]

[1] https://github.com/AISE-TUDelft/tiny-transformers

[2] https://huggingface.co/collections/AISE-TUDelft/brp-tiny-transformers-666c352b3b570f44d7d2a519

[3] https://github.com/laurikskl/transformers

Transformers, since their introduction by Vaswani et al. [26], have become the cornerstone of modern natural language processing (NLP) due to their capability to model long-range dependencies through self-attention mechanisms. However, the fixed-size context window inherent to standard Transformers limits their ability to process lengthy documents and understand dependencies beyond these windows.

Increasing the context length in Transformers leads to a quadratic scaling in computational and memory requirements. Specifically, the self-attention mechanism, which is the heart of Transformers, has a time complexity of $O(n^2)$ and a space complexity of $O(n^2)$, where n is indicative of sequence length.

Quadratic scaling arises because the self-attention mechanism computes pairwise interactions between all elements in the input sequence (see Figure 1). Consequently, processing very long sequences becomes infeasible, necessitating further innovations in model architecture and optimization techniques. Techniques such as sparse attention [11], memory-efficient attention mechanisms [15, 7, 3], and hierarchical models [18, 32] are actively being explored to mitigate these challenges and push the boundaries of sequence length that Transformers can effectively handle.

2.2 Previously Proposed Solutions

One of the more popular solutions proposed, Transformer-XL [7], aims to address the limitations of fixed-length context in standard Transformer models by incorporating a segment-level recurrence mechanism. By reusing hidden states from previous segments as memory, Transformer-XL can model dependencies spanning multiple segments, thereby enhancing its ability to process and generate long texts.

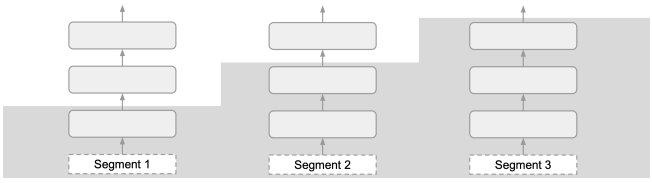

Figure 2: Segment-wise processing of Transformer-XL, which captures long-term dependencies by reusing hidden states from previous segments as memory. Taken from Munkhdalai et al. [17]

However, Transformer-XL also comes with notable challenges, particularly in terms of computational resources. The requirement to cache hidden states from previous segments leads to substantial memory usage, which can be manageable for smaller models but becomes problematic for larger models (e.g., 30B parameters or more). Segment 3 (see Figure 2), for example, needs to maintain the cache from segments 1 and 2 in VRAM, which significantly increases memory demands. This high memory demand makes it difficult to scale Transformer-XL. Additionally, engineering the training pipeline to handle the recurrence mechanism and the extensive memory caching adds complexity. These factors contribute to Transformer-XL being less popular despite its innovative approach.

BigBird [33] is a Transformer model that addresses the problem of quadratic scaling by introducing a sparse attention mechanism. This reduces the computational complexity from quadratic to linear, enabling the model to process significantly longer sequences with the same computational resources.

The core innovation in BigBird is its sparse attention mechanism, which combines three types of attention patterns: global, local, and random. Global attention involves a small set of special tokens that attend to all parts of the sequence, ensuring that important global information is captured. Local attention ensures that each token attends to its neighboring tokens, preserving local context. Random attention allows each token to attend to a set of randomly selected tokens, facilitating the capture of long-range dependencies without the need for full attention across the entire sequence.

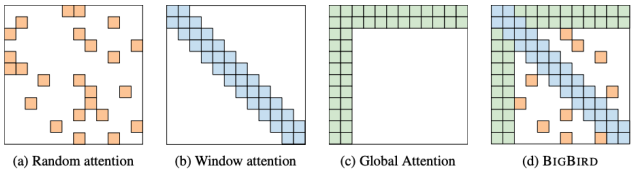

Figure 3: Self-attention mechanism of BigBird. Each colored square is indicative of a pair-wise score calculation between two tokens in a sequence. Taken from Zaheer et al. [33]

However, while BigBird achieves improvements in handling longer contexts, it is not without drawbacks. The introduction of sparse attention patterns requires careful tuning, making it difficult to maintain universally high performance among different tasks. Moreover, the random selection process in the attention mechanism leads to unwanted variability in model performance.

Infini-attention [17] addresses the limitations of traditional Transformers by incorporating a compressive memory into the Transformer architecture. This approach allows the model to handle infinitely long contexts with a linear increase in computational demands, transitioning from $O(n^2)$ time complexity to $O(n)$ time complexity and achieving $O(1)$ space complexity.

Infini-attention combines local self-attention and long-term linear attention [21] within a single Transformer block. It works by storing key-value (KV) pairs from previous segments in a compressive memory [12] instead of discarding them. When processing new sequences, the model uses attention queries to retrieve values from this memory.

The authors of Infini-attention promise that their solution to the context-length problem allows Large Language Models to naturally scale to infinitely long inputs while using bounded memory and computation. However, their approach relies on a linear attention mechanism [22]. Past architectures that have relied on the same mechanism, such as the Performer [5], have been empirically demonstrated to suffer from performance issues in comparison to the vanilla Transformer architecture [25].

Furthermore, Munkhdalai et al. do not push the limits of the compressive memory mechanism satisfactorily nor pro-

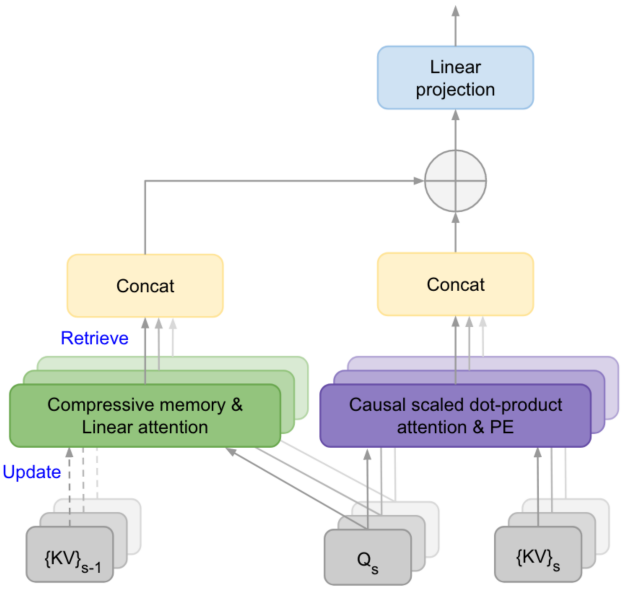

Figure 4: Architecture proposed in Infini-attention. Taken from Munkhdalai et al. [17]

vide detailed instructions on the amount of fine-tuning required for successful integration. The lack of comparison with vanilla Transformers further limits the evaluation of the Infini-attention architecture. These omissions suggest that while promising, the proposed method may face practical challenges in real-world applications and its advantages over existing architectures remain unclear.

3 Approach

In the following paragraphs, we will focus on the compressive memory component of the Infini-attention architecture. First, we define its structure and motivation. Next, we detail the processes of information updating and retrieval. Finally, we address to what extent our finite compressive memory can store an infinite amount of information.

3.1 Structure of the Compressive Memory

The Compressive memory is parameterized with an associative matrix described in Schlag et al. [20]. It is hetero-associative [16], meaning that it allows for retrieval of partial matches, which is important in the context of language modelling with the attention mechanism, as a query and key have a non-binary, scored match. The dimensions of the memory are described below.

$$M \in \mathbb{R}^{d_{\text{key}} \times d_{\text{value}}}$$

3.2 Key-Value Binding in Compressive Memory

Binding [13, 12] is a fundamental concept used for compressive memory as it is crucial for storing and retrieving information efficiently.

Basic Concept

Key-value binding involves associating a key vector $\mathbf{k}$ with a value vector $\mathbf{v}$. This association can be conceptualized as storing data in a memory slot where the key serves as an address and the value as the data. Typically, the outer product of the key and value is used:

$$M = \mathbf{k} \otimes \mathbf{v}$$

Here, M represents the memory matrix, and the outer product $\mathbf{k} \otimes \mathbf{v}$ ensures that each element of the key is associated with each element of the value.

Binding and Retrieval

To bind a key $\mathbf{k}$ with a value $\mathbf{v}$, these vectors are combined such that the original value can be retrieved when the key is provided. If $\mathbf{k}_{\text{query}}$ is the query key, the retrieval operation can be expressed as:

$$\mathbf{v}_{\text{retrieved}} = M\mathbf{k}_{\text{query}}$$

In high-dimensional spaces, the orthogonality of vectors plays a crucial role. Orthogonal vectors ensure that incorrect keys retrieve essentially 'useless' information, while only the correct or nearly correct keys will retrieve the original or close-to-original values. This property is essential for maintaining the integrity of stored information and for efficient retrieval.

Binary Vectors and XOR Operation:

Binary vectors are an excellent way to demonstrate this property, as a similar principle applies using the XOR operation.

Suppose we have two binary vectors:

- Key vector $\mathbf{k} = [1, 0, 1, 0]$
- Value vector $\mathbf{v} = [0, 1, 0, 1]$

To bind $\mathbf{k}$ and $\mathbf{v}$ using XOR:

$$\mathbf{k} \oplus \mathbf{v} = [1, 1, 1, 1]$$

Retrieving the value $\mathbf{v}$ given the key $\mathbf{k}$ involves applying XOR again with the result of the previous XOR operation:

$$\mathbf{v} = \mathbf{k} \oplus (\mathbf{k} \oplus \mathbf{v}) = [0, 1, 0, 1]$$

Taking our memory M that has the binding $\mathbf{k} \oplus \mathbf{v} = [1, 1, 1, 1]$, and applying XOR again with the key that was used to store it, retrieves the original value vector $\mathbf{v} = [0, 1, 0, 1]$.

3.3 Updating the Compressive Memory

We first give a simplified version of the update equation of M for demonstrative purposes, which is an adapted version of the delta rule [30]

$$M = M + \mathbf{k}^T(\mathbf{v} - \mathbf{k} \cdot M) \tag{1}$$

To understand this better, we can 'open up' the matrix M as the outer products of key and value vectors that populate it. Suppose that we have already added the entries $\mathbf{k}_1^T\mathbf{v}_1$ and $\mathbf{k}_2^T\mathbf{v}_2$. We are now adding the entry $\mathbf{k}_3$, $\mathbf{v}_3$, such that $\mathbf{k}_3$ is

not orthogonal to $\mathbf{k}_2$ and $\mathbf{k}_3 = \mathbf{k}_2$. The update proceeds as follows:

$$M = M + \mathbf{k}_3^T(\mathbf{v}_3 - \mathbf{k}_3 M) \tag{2}$$
$$= \mathbf{k}_1^T\mathbf{v}_1 + \mathbf{k}_2^T\mathbf{v}_2 + \mathbf{k}_3^T(\mathbf{v}_3 - \mathbf{v}_2) \tag{3}$$
$$= \mathbf{k}_1^T\mathbf{v}_1 + \mathbf{k}_2^T\mathbf{v}_2 + \mathbf{k}_3^T\mathbf{v}_3 - \mathbf{k}_3^T\mathbf{v}_2 \tag{4}$$
$$= \mathbf{k}_1^T\mathbf{v}_1 + \mathbf{k}_2^T\mathbf{v}_2 + \mathbf{k}_3^T\mathbf{v}_3 - \mathbf{k}_2^T\mathbf{v}_2 \tag{5}$$
$$= \mathbf{k}_1^T\mathbf{v}_1 + \mathbf{k}_2^T\mathbf{v}_2 + 0 \tag{6}$$
$$= \mathbf{k}_1^T\mathbf{v}_1 + \mathbf{k}_2^T\mathbf{v}_2 \tag{7}$$

This can be described as clearing $\mathbf{k}_2 \cdot \mathbf{v}_2$ and keeping $\mathbf{k}_3 \cdot \mathbf{v}_3$, the newest entry. Essentially, the component $\mathbf{k}_2 \cdot \mathbf{v}_2$ is nullified by $\mathbf{k}_3 \cdot (\mathbf{v}_3 - \mathbf{v}_2)$ since $\mathbf{k}_3 = \mathbf{k}_2$ and they are not orthogonal, ensuring that $\mathbf{v}_3 - \mathbf{v}_2$ almost entirely cancels out $\mathbf{k}_2 \cdot \mathbf{v}_2$.

Infini-attention's update rule

A modern twist to the delta update rule was first presented in Schlag et al. [21], which the Infini-attention version of the update rule borrows from. The update to the memory state is given by:

$$M_s \leftarrow M_{s-1} + (\sigma(K))^T \left(V - \frac{\sigma(K)M_{s-1}}{\sigma(K)z_{s-1}}\right) \tag{8}$$

This operation adjusts the memory by integrating new key-value pairs while ensuring that duplicate values are not stored. The update rule Infini-attention uses has an added normalization term for numerical stability and an element-wise activation function ELU+1 [6], which is represented by σ in the above equation. The use of ELU+1 is notably not motivated or explained by Munkhdalai et al. [17]. Previous research [14, 21] suggests that ELU+1 is desirable in similar contexts due to its non-zero gradients on negative inputs and its ability to preserve dimensions. ELU+1 was likely chosen for these properties, particularly its dimension preservation, which is beneficial for memory operations. Nonetheless, a detailed comparison with other activation functions would provide a clearer understanding of its specific advantages in this context.

3.4 Retrieving From the Compressive Memory

The retrieved memory content A_{mem} is computed as follows:

$$A_{mem} = \frac{\sigma(Q)M_{s-1}}{\sigma(Q)z_{s-1}} \tag{9}$$

This equation utilizes the query specific to the local context to fetch relevant values from memory. Here σ and z_{s-1} are the same activation function and normalization term that are used by the update rule. According to Schlag et al. [20], the associative memory mechanism allows for the retrieval of patterns that differ from the input pattern. This indicates that the model can handle overlapping keys storing different values, enabling strong performance even when the compressive memory begins to store non-orthogonal keys.

3.5 How Infinite is this Finite space?

From an analytical perspective, if we have a key dimension of n, then the compressive memory should become full after n unique keys have been added to it. This would give a capacity of 128 values in the case of having keys in 128 dimensions. This, however, would be in stark contrast to the impressive results displayed in Munkhdalai et al. [17], where a model with a key dimension of 1024 and a segment length of 2048 was comfortably scaled to succeed at book summarization tasks which had lengths of up to 500K tokens. Analytical methods would lead us to believe that the compressive memory would allow for a roughly two-fold increase in context length, since we can expect some values to repeat in a story. This would lead us to believe that in such a situation Infini-attention would allow us to extend our model to work with context lengths of up to around 4092, which is dwarfed by the 500k token summarization tasks it was found to be capable of.

This phenomenon can be explained by the principles established by Pentti Kanerva, who demonstrated that nearly orthogonal vectors are sufficient for effective memory retrieval [12]. As illustrated in Figure 5, the distances between most vectors in high-dimensional spaces such as our compressive memory cluster tightly around 0.5, indicating that they are nearly orthogonal and highly dissimilar. Consequently, even when the memory contains keys that are not perfectly orthogonal, the model can still perform well by leveraging the nearly-orthogonal properties of high-dimensional vectors.

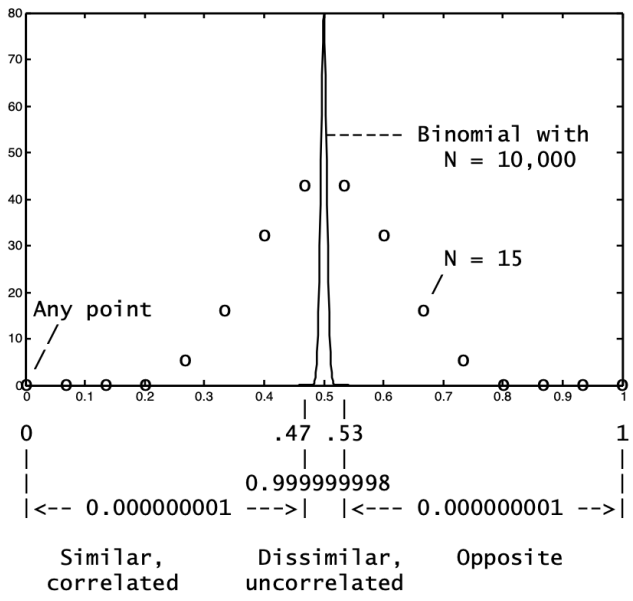

Figure 5: Distribution of distances in 10,000-dimensional spaces. In high-dimensional space (N=10,000), most vectors cluster around a distance of 0.5, indicating near orthogonality and high dissimilarity. Taken from Stanford Online [23]

To summarize, the compressive memory component of Infini-attention leverages the properties of high-dimensional spaces to effectively store and retrieve vast amounts of infor-

mation. Despite the theoretical limits imposed by key dimensions, Kanerva's work shows that nearly orthogonal vectors in high-dimensional spaces allow for the practical storage of an almost infinite number of key-value bindings. This capability significantly enhances the model's performance and storage efficiency in tasks requiring long-term dependencies.

4 Experimental Setup

4.1 Research Questions

Our study is guided by the following research questions:

RQ1: Does incorporating Infini-attention from the beginning of pre-training yield better performance compared to adding it in fine-tuning?

Independent Variable: Stage of incorporating Infini-attention (pre-training vs. fine-tuning)

Dependent Variable: Model performance on the BabyLM pipeline, described in section 4.4

RQ2: How well can the compressive memory compensate for shortened context lengths?

Independent Variable: Segment length

Dependent Variable: Model performance on the BabyLM pipeline, described in section 4.4

RQ3: How long should the fine-tuning process last for the gating parameters (betas) to converge?

Independent Variable: Duration of fine-tuning

Dependent Variable: Convergence of the gating parameters (betas), measured by stability and performance on the BabyLM pipeline, described in section 4.4

4.2 Datasets

We exclusively use the TinyStories [9] dataset, which is a collection of short, simple stories generated by GPT-3.5 and GPT-4. The longest samples have a length of 512 tokens, while the median length is 216 tokens. Although there are some stories longer than 1000 tokens, they constitute less than 1% of the dataset and have been trimmed to a maximum length of 512 tokens. The stories use words understandable to 3-4 year olds. By limiting the breadth of our dataset in this manner, it can be used to train and evaluate small language models (LMs), with the sole goal of learning to understand grammar. We use this dataset to leverage its simplicity and manageability, ensuring that our models remain coherent, interpretable, and effective even with limited parameters and a constrained training budget.

4.3 Models

For our baseline, we employ the GPT-NEO model [4], which is derived from the original transformer architecture proposed by Vaswani et al. [26]. In our experiments, we utilize a modified version of GPT-NEO that incorporates rotary embeddings [24] to provide positional encodings to the input sequence. Additionally, we replace the standard self-attention mechanism with Infini-attention [17], aiming to enhance the model's capacity to manage extended contexts efficiently. An important point to note is that due to the triangular (causal) mask in causal language models, each input sequence to a model that trains on segments of length 64 'sees' $\frac{512}{64}$ fewer total training samples. With a context length of 512, the model processes tokens incrementally from $p = (t_n|t_0, ..., t_n)$ for $n = [0, ..., 512]$, where t is segment length. This makes each training sample effectively 512 training samples.

4.4 Evaluation Settings and Metrics

We evaluate on the evaluation pipeline intended for use in the BabyLM 2023 challenge [29]. The pipeline consists of BLiMP [28] and superGLUE [27].

BLiMP (Benchmark of Linguistic Minimal Pairs) tests models on their ability to distinguish between grammatically acceptable and unacceptable sentences across various linguistic phenomena. It includes pairs of minimally different sentences in categories such as anaphoric agreement, argument structure, and subject-verb agreement.

SuperGLUE (General Language Understanding Evaluation) is a benchmark consisting of several language understanding tasks. It evaluates a model's ability to understand and reason about text through tasks involving single-sentence classification, similarity and paraphrase detection, and natural language inference.

4.5 Configuration and Implementation Details

In lieu of a replication package from the authors of Infini-attention, we have undertaken the challenge of implementing several components from scratch. This process includes significant modifications to the HuggingFace transformers library [31] to support Infini-attention for input prompts exceeding the model's local context length. As traditional positional embeddings are limited to the local context length they were trained on, we integrated rotary embeddings inspired by GPT-NEOX [2] to meet our need for positional embeddings capable of handling infinitely long contexts.

Additionally, we have developed a custom attention mechanism aligned with the specifications detailed by Munkhdalai et al. [17]. Although we drew inspiration from various sources, we have tailored our implementation to fit the unique requirements of our study, ensuring robustness and fidelity to the original concepts.

Evaluation and training were conducted using an NVIDIA GTX 3080 and NVIDIA V100 with 4 CPUs and 24GB of memory on the DelftBlue cluster [8]. cards, leveraging the PyTorch library [19].

5 Results

This section presents the outcomes of our experiments, organized by research questions.

5.1 Pretraining vs fine-tuning

The performance differences between pre-training with Infini-Attention and pre-training followed by fine-tuning with Infini-Attention were analyzed. The results are presented in Table 1.

Models that were fully pre-trained with Infini-Attention outperformed those that were only fine-tuned with this augmentation. Notably, the fully pre-trained Infini-Attention GPT-Neo with a segment length (SL) of 256 achieved the highest score on BLiMP with 0.618 and also performed well on

Table 1: BLiMP and SuperGLUE scores for vanilla GPT-Neo, GPT-Neo models that were fully pre-trained with Infini-attention enabled and GPT-Neo models that were first pre-trained without augmentations and then fine-tuned with Infini-Attention enabled. SL stands for the segment length that the model was trained on.

Name	BLiMP	SuperGLUE
GPT-Neo ($SL = 32$)	0.604	0.478
GPT-Neo ($SL = 64$)	0.595	0.512
GPT-Neo ($SL = 128$)	0.587	0.478
GPT-Neo ($SL = 256$)	0.611	0.540
GPT-Neo ($SL = 512$)	0.561	0.537
Infini-attention GPT-Neo Fully Pre-trained ($SL = 64$)	0.580	0.474
Infini-attention GPT-Neo Fully Pre-trained ($SL = 256$)	**0.618**	0.533
Infini-attention GPT-Neo Fine-tuned ($SL = 32$)	0.572	0.476
Infini-attention GPT-Neo Fine-tuned ($SL = 64$)	0.558	0.470
Infini-attention GPT-Neo Fine-tuned ($SL = 128$)	0.574	0.528
Infini-attention GPT-Neo Fine-tuned ($SL = 256$)	0.566	**0.549**

SuperGLUE with a score of 0.533. However, it is noteworthy that the fine-tuned model with a segment length of 256 achieved the best SuperGLUE score of 0.549, while other fine-tuned models showed relatively lower performance. Despite this, fully pre-training using Infini-attention appears to yield better results across different metrics compared to just fine-tuning.

5.2 How much fine-tuning is required

The gating parameters, referred to as betas, are critical for the Infini-attention architecture. These parameters govern the extent to which each attention head incorporates information from the compressive memory, thereby enhancing the model's ability to manage long-term dependencies. When transitioning a vanilla transformer to one equipped with Infini-attention, the betas are the only new parameters introduced that necessitate fine-tuning.

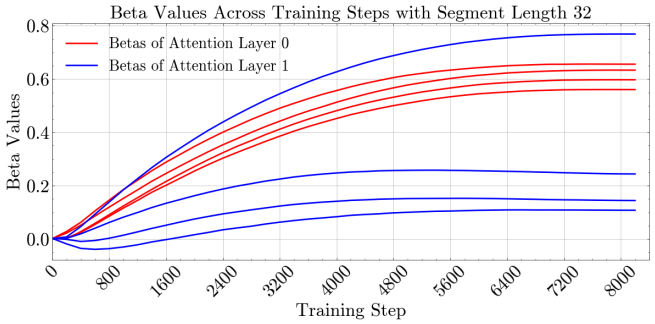

Figure 6: Beta values during a single epoch of fine-tuning with Infini-attention enabled and a segment length of 32.

Figures 6 and 7 illustrate the convergence of gating parameters during the fine-tuning process for segment lengths of 32 and 256, respectively. In Figure 6, the gating parameters exhibit relatively high values, whereas in Figure 7, they trend towards lower values. Notably, Figure 8 presents the best balance between high and low beta values among all the segment lengths observed.

We also note that the beta values of the first self-attention layer tend to cluster around slightly positive values near 0.1. However, an exception is observed in the model trained with

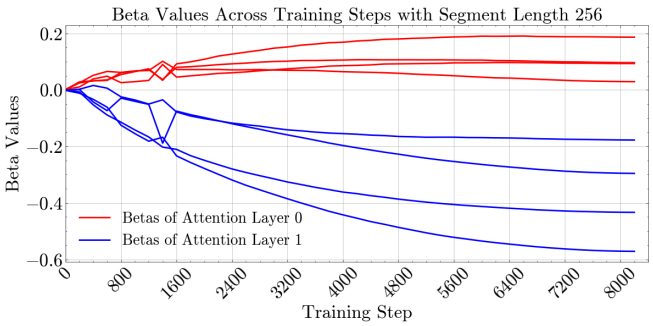

Figure 7: Beta values during a single epoch of fine-tuning with Infini-attention enabled and a segment length of 256.

a segment length of 32, where the beta values are significantly higher, clustering around 0.6.

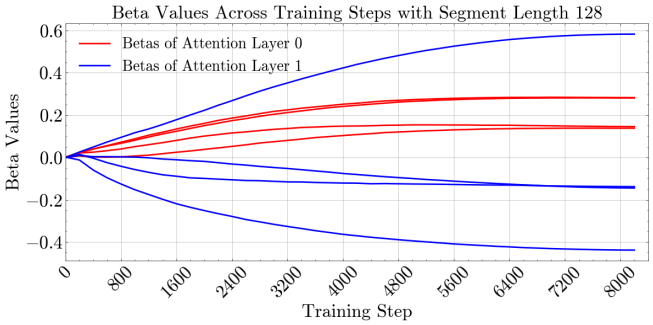

Figure 8: Beta values during a single epoch of fine-tuning with Infini-attention enabled and a segment length of 128.

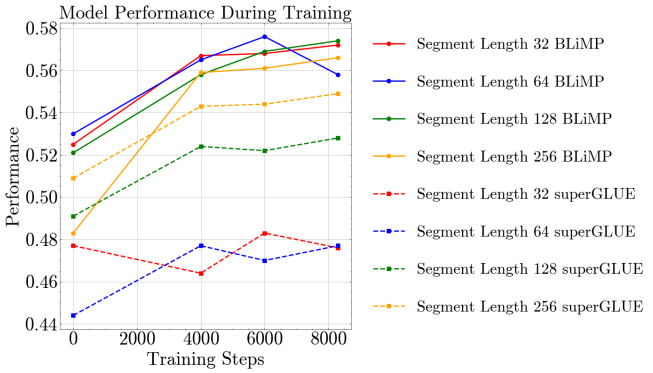

Figure 9: BLiMP and superGLUE scores of Infini-attention enabled GPT-Neo models throughout the fine-tuning process that are trained with differing segment lengths

Figure 9 shows the model performance during training with different segment lengths for BLiMP and superGLUE tasks. The performance values for both metrics are plotted for segment lengths of 32, 64, 128, and 256.

5.3 Decreasing the context length

The results show (Table 1) that Infini-Attention enabled GPT-Neo models which have been trained on shortened segment

lengths are able to maintain performance on the BLiMP benchmark but do suffer from decreased performance on the tasks posed in superGLUE.

6 Discussion

6.1 Implications

Impact of Pre-training and Fine-tuning
The key difference between fully pre-training using Infini-attention and doing pre-training followed by fine-tuning with Infini-attention appears to stem from the choice of segment length used during training. We observed that models tend to perform better if they are consistently trained on the same segment length throughout the entire training process. This holds true regardless of whether Infini-attention is enabled or not. The models that were fine-tuned using Infini-attention started from a common checkpoint, which had been trained on segment lengths of 512 tokens. We suspect that fully pre-trained models managed to beat out the fine-tuned models because they were trained on the same segment length throughout the process. While it may be tempting to vary segment lengths due to budget constraints, our recommendation for using Infini-attention is to maintain a consistent segment length during both pre-training and fine-tuning phases.

Although we hypothesized that the model might benefit from having more time to acclimate to Infini-attention, our findings tend to indicate otherwise. It is difficult to state this for certain without additional experiments, in which we maintain a consistent segment length throughout vanilla pre-training and Infini-attention enabled fine-tuning. However, it is important to note that training with Infini-attention enabled is more computationally expensive due to the extra matrix multiplication steps involved in the forward pass during update and retrieval operations with compressive memory. Therefore, pre-training with Infini-attention-enabled is likely unnecessary.

Efficiency of Fine-tuning: Rapid Convergence of Betas
The behavior of beta values in the Infini-Attention architecture indicates the model's reliance on compressive memory versus local context. High beta values show a dependence on compressive memory, while low values emphasize local context.

Figures 6, 8, and 7 show that beta values converge around step 4000 for all models. Evaluations confirm that performance improvements beyond this point are marginal, as shown in Figure 9. This suggests that fine-tuning beyond step 4000 (0.5 epochs) offers diminishing returns, marking it as a practical convergence point. Despite variations in segment length, the gating parameters stabilize around similar values, ensuring a balanced use of local context and compressive memory.

The higher beta values exhibited by the model with a short segment length of 32 tokens (Figure 6) indicate a greater reliance on compressive memory due to the limited local context. In contrast, the model with a 256-token segment length has lower beta values (Figure 7), relying more on the sufficient local context available. The beta values found in the 128 token segment length model however seem to be the most balanced in their attention towards local context and compressive memory (Figure 8). This may be a sign that Infini-attention is most comfortably able to extend the maximal processable sequence length of a model by a factor of 4, as the longest sequence we trained on had a length of 512.

In summary, shorter segment lengths necessitate higher beta values, while longer segment lengths favor local context. The convergence around step 4000 indicates that 0.5 epochs worth of fine-tuning is required to fully utilize Infini-Attention.

Effects of Context Length on Performance
The results show (Table 1) that shorter context lengths in Infini-attention-enabled GPT-NEO models lead to decreased performance, especially on the superGLUE benchmark. Tasks in superGLUE often involve long sequences that need detailed contextual understanding.

As previously discussed, Infini-attention uses a compressive memory to store past key-value (KV) pairs. While this enhances the model's capacity for long-term dependencies, it may introduce challenges for induction heads [10], which rely on precise patterns in the input sequence to predict token completions. Compressive memory may obscure these patterns, as the values retrieved from it are 'fuzzy', but this calls for further experiments.

6.2 Threats to the Validity

Internal Validity
One potential threat to the validity of our findings is the size of the model used in our experiments. The model may be too small to fully leverage the benefits provided by Infini-Attention. Larger models might better utilize the enhanced capacity for long-term dependencies provided by Infini-Attention, potentially leading to more significant performance improvements.

External Validity
The quality of the TinyStories dataset also poses a threat to the validity of our findings. Although this dataset is specifically designed for grammatical evaluation, it falls short in terms of contextual understanding, which is a critical component of the superGLUE evaluation metric. For instance, consider the following excerpt from one of the stories:

"She felt it was too boring to just write about trees and flowers. Suddenly, an idea came to her. She decided to write about her waist."

This snippet illustrates the lack of contextually coherent narrative that can be found in some samples. Such issues in the dataset can lead to misleading evaluations, as the data might not adequately represent the complexities and contextual dependencies required for tasks like those in superGLUE. Therefore, the dataset's limitations must be considered when interpreting our results, and future studies should ensure that evaluation datasets are contextually rich and coherent.

Construct Validity Another critical threat to the validity of our results is the suitability of the evaluation metric.

The evaluation metrics used, particularly BLiMP, may not be adequately designed to push the limits of a model's context length capabilities. These metrics might not fully capture the improvements in contextual understanding and long-term dependency management that Infini-attention aims to provide. To fit the BabyLM challenge, we had to test tuning down the local context length of Infini-Attention, which is undesirable in terms of performance with such short segment lengths to begin with (512). Consequently, the observed performance differences might not reflect the true potential of Infini-Attention, highlighting the need for more tailored evaluation metrics that can better assess performance on longer sequences.

6.3 Future Work

As we have found that using Infini-Attention during pre-training is not necessary and that a small amount of additional fine-tuning is sufficient, future research should focus on models that utilize a checkpoint that has been pre-trained on the same segment length intended to be used after fine-tuning with Infini-attention enabled. This approach could reveal more significant performance improvements that were not fully captured with the smaller models used in the current study.

Additionally, current evaluation metrics may not be fully capable of pushing the limits of a model's context length capabilities. Future studies should employ or develop evaluation metrics specifically designed to measure how well models handle longer context lengths. Such metrics would better assess the improvements in contextual understanding and long-term dependency management that Infini-Attention aims to provide.

We also hypothesize that Infini-attention and compressive memory show promise for 'open-book' questions, where a question or task is posed before presenting the model with the input sequence required to complete the given task. This contrasts with the 'closed-book' questions typical of benchmarks like superGLUE, where the model is first given an input sequence and then required to answer based on what it has seen. Future research should explore the application of Infini-attention in scenarios that simulate 'open-book' conditions, as this could fully leverage the model's capability to manage long-term dependencies as it would be able to 'know' beforehand what kind of signal it should save from the input sequence to complete the task it was given.

7 Conclusion

The primary goal of this research was to evaluate the effectiveness of Infini-Attention in transformer models, specifically investigating whether its integration during pre-training or fine-tuning yields better performance. We also explored how well compressive memory can compensate for shortened context lengths in transformer models.

Our findings indicate that models trained with consistent segment lengths throughout the entire training process outperformed those with varied segment lengths. This suggests that maintaining consistent segment lengths during both pre-training and fine-tuning phases is crucial for optimal performance when using Infini-Attention.

Additionally, we have found that beta values converge around step 4000 for all models in our experiments, and evaluations confirm that performance improvements beyond this point are marginal. This suggests that fine-tuning beyond step 4000 (0.5 epochs) offers diminishing returns, marking it as a practical convergence point. Despite variations in segment length, the gating parameters stabilize around similar values, ensuring a balanced use of local context and compressive memory.

The results also showed that shorter context lengths lead to decreased performance, which Infini-attention did not seem to properly compensate for. This happened particularly on tasks requiring long-term dependencies, as seen in the superGLUE benchmark. This highlights the importance of adequate context length for maintaining high performance in models utilizing Infini-Attention.

Future research should focus on evaluating models that utilize a baseline model that has been pre-trained on the same segment length intended to be used after fine-tuning with Infini-attention enabled. Additionally, more tailored evaluation metrics that better assess performance on longer sequences should be developed to capture the true potential of Infini-Attention.

8 Responsible Research

In this study, we prioritized transparency and reproducibility by openly sharing our data, methodologies, and findings. The necessary modifications to the HuggingFace Transformers library for processing long text inputs in segments are available in a forked version on Lauri Kesküll's GitHub[4]. Furthermore, the altered GPT-NEO model, utilizing Infini-Attention for its attention mechanism and incorporating rotary embeddings from GPT-NEOX, is also accessible on this platform.

We strictly adhered to scientific integrity principles, avoiding any data manipulation, fabrication, or plagiarism. All reported results stem from our experimental setup and are accurately presented without any alterations. The data used in this study is properly cited, and all references to previous work are duly acknowledged.

Ethical considerations were thoroughly addressed throughout the study. There were no conflicts of interest, and all procedures complied with guidelines for ethical research conduct. Our experiments were designed and conducted with a commitment to honesty, rigor, transparency, and respect for intellectual property.

We integrated principles and practices from chapters 2 and 3 of the Netherlands Code of Conduct for Research Integrity. This included maintaining detailed records of our research process, openly sharing our findings, and fostering an environment of integrity and respect within our research team. We remain committed to upholding the highest standards of ethical research practice.

References

[1] Jay Alammar. The illustrated transformer. https://jalammar.github.io/illustrated-transformer/, 2018. Blog post.

[4] https://github.com/laurikskl/transformers

[2] Alex Andonian, Quentin Anthony, Stella Biderman, Sid Black, Preetham Gali, Leo Gao, Eric Hallahan, Josh Levy-Kramer, Connor Leahy, Lucas Nestler, Kip Parker, Michael Pieler, Jason Phang, Shivanshu Purohit, Hailey Schoelkopf, Dashiell Stander, Tri Songz, Curt Tigges, Benjamin Thérien, Phil Wang, and Samuel Weinbach. Gpt-neox: Large scale autoregressive language modeling in pytorch, August 2021. URL https://www.github.com/eleutherai/gpt-neox. If you use this software, please cite it using these metadata.

[3] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer, 2020.

[4] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021. URL https://doi.org/10.5281/zenodo.5297715. If you use this software, please cite it using these metadata.

[5] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, David Belanger, Lucy Colwell, and Adrian Weller. Rethinking attention with performers, 2022.

[6] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs), February 2016. URL http://arxiv.org/abs/1511.07289. arXiv:1511.07289 [cs].

[7] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context, June 2019. URL http://arxiv.org/abs/1901.02860. arXiv:1901.02860 [cs, stat].

[8] Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 2). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase2, 2024.

[9] Ronen Eldan and Yuanzhi Li. TinyStories: How Small Can Language Models Be and Still Speak Coherent English?, May 2023. URL http://arxiv.org/abs/2305.07759. arXiv:2305.07759 [cs].

[10] Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

[11] Sebastian Jaszczur, Aakanksha Chowdhery, Afroz Mohiuddin, Łukasz Kaiser, Wojciech Gajewski, Henryk Michalewski, and Jonni Kanerva. Sparse is enough in scaling transformers, 2021.

[12] Pentti Kanerva. *Sparse Distributed Memory*. MIT Press, 1988. ISBN 978-0-262-11132-4.

[13] Pentti Kanerva. What we mean when we say "what's the dollar of mexico?": Prototypes and mapping in concept space. 01 2010.

[14] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention, August 2020. URL http://arxiv.org/abs/2006.16236. arXiv:2006.16236 [cs, stat].

[15] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer, 2020.

[16] Bart Kosko. Bidirectional associative memories. *IEEE Trans. Syst. Man Cybern.*, 18:49–60, 1988. URL https://api.semanticscholar.org/CorpusID:59875735.

[17] Tsendsuren Munkhdalai, Manaal Faruqui, and Siddharth Gopal. Leave No Context Behind: Efficient Infinite Context Transformers with Infini-attention, April 2024. URL http://arxiv.org/abs/2404.07143. arXiv:2404.07143 [cs].

[18] Raghavendra Pappagari, Piotr Żelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. Hierarchical transformers for long document classification, 2019.

[19] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.

[20] Imanol Schlag, Tsendsuren Munkhdalai, and Jürgen Schmidhuber. Learning Associative Inference Using Fast Weight Memory, February 2020. URL http://arxiv.org/abs/2011.07831. arXiv:2011.07831 [cs].

[21] Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear Transformers Are Secretly Fast Weight Programmers, June 2021. URL http://arxiv.org/abs/2102.11174. arXiv:2102.11174 [cs].

[22] Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient Attention: Attention with Linear Complexities, January 2024. URL http://arxiv.org/abs/1812.01243. arXiv:1812.01243 [cs].

[23] Stanford Online. Stanford Seminar - Computing with High-Dimensional Vectors, October 2017. URL https://www.youtube.com/watch?v=zUCoxhExe0o.

[24] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding, November 2023. URL http://arxiv.org/abs/2104.09864. arXiv:2104.09864 [cs].

[25] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long Range Arena : A

Benchmark for Efficient Transformers. October 2020. URL https://openreview.net/forum?id=qVyeW-grC2k.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2017. URL http://arxiv.org/abs/1706.03762. arXiv:1706.03762 [cs].

[27] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems, February 2020. URL http://arxiv.org/abs/1905.00537. arXiv:1905.00537 [cs].

[28] Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. Blimp: The benchmark of linguistic minimal pairs for english. *Transactions of the Association for Computational Linguistics*, 8:377–392, 2020. doi: 10.1162/tacl_a_00321. URL https://doi.org/10.1162/tacl_a_00321.

[29] Alex Warstadt, Leshem Choshen, Aaron Mueller, Adina Williams, Ethan Wilcox, and Chengxu Zhuang. Call for papers – the babylm challenge: Sample-efficient pretraining on a developmentally plausible corpus, 2023.

[30] Bernard Widrow and Marcian E. Hoff. Adaptive switching circuits. In *1960 IRE WESCON Convention Record, Part 4*, pages 96–104, New York, 1960. IRE.

[31] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Perric Cistac, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, October 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[32] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. Hi-transformer: Hierarchical interactive transformer for efficient and effective long document modeling, 2021.

[33] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big Bird: Transformers for Longer Sequences, January 2021. URL http://arxiv.org/abs/2007.14062. arXiv:2007.14062 [cs, stat].

A The Convergence of Gating Parameters Betas

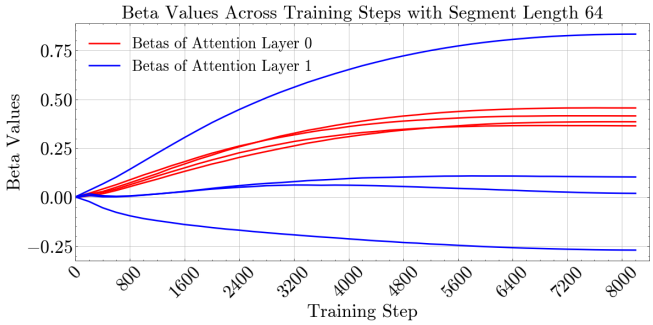

Figure 10: Beta values during a single epoch of fine-tuning with Infini-attention enabled.