Leave No Context Behind:
Efficient Infinite Context Transformers with Infini-attention

Tsendsuren Munkhdalai, Manaal Faruqui and Siddharth Gopal
Google
tsendsuren@google.com

arXiv:2404.07143v1 [cs.CL] 10 Apr 2024

Abstract

This work introduces an efficient method to scale Transformer-based Large Language Models (LLMs) to infinitely long inputs with bounded memory and computation. A key component in our proposed approach is a new attention technique dubbed Infini-attention. The Infini-attention incorporates a compressive memory into the vanilla attention mechanism and builds in both masked local attention and long-term linear attention mechanisms in a single Transformer block. We demonstrate the effectiveness of our approach on long-context language modeling benchmarks, 1M sequence length passkey context block retrieval and 500K length book summarization tasks with 1B and 8B LLMs. Our approach introduces minimal bounded memory parameters and enables fast streaming inference for LLMs.

Local and global attention in same block

1 Introduction

Memory serves as a cornerstone of intelligence, as it enables efficient computations tailored to specific contexts. However, Transformers (Vaswani et al., 2017) and Transformer-based LLMs (Brown et al., 2020; Touvron et al., 2023; Anil et al., 2023; Groeneveld et al., 2024) have a constrained context-dependent memory, due to the nature of the attention mechanism.

The attention mechanism in Transformers exhibits quadratic complexity in both memory footprint and computation time. For example, the attention Key-Value (KV) states have 3TB memory footprint for a 500B model with batch size 512 and context length 2048 (Pope et al., 2023). Indeed, scaling LLMs to longer sequences (i.e. 1M tokens) is challenging with the standard Transformer architectures and serving longer and longer context models becomes costly financially.

Compressive memory systems promise to be more scalable and efficient than the attention mechanism for extremely long sequences (Kanerva, 1988; Munkhdalai et al., 2019). Instead of using an array that grows with the input sequence length, a compressive memory primarily maintains a fixed number of parameters to store and recall information with a bounded storage and computation costs. In the compressive memory, new information is added to the memory by changing its parameters with an objective that this information can be recovered back later on. However, the LLMs in their current state have yet to see an effective, practical compressive memory technique that balances simplicity along with quality.
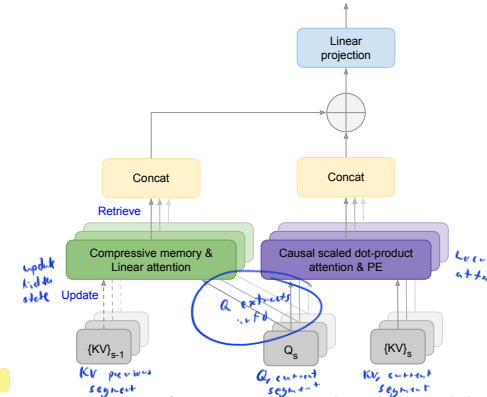

Figure 1: Infini-attention has an additional compressive memory with linear attention for processing infinitely long contexts. $\{KV\}_{s-1}$ and $\{KV\}_s$ are attention key and values for current and previous input segments, respectively and Q_s the attention queries. PE denotes position embeddings.

In this work, we introduce a novel approach that enables Transformer LLMs to effectively process infinitely long inputs with bounded memory footprint and computation. A key component in our proposed approach is a new attention technique dubbed Infini-attention (Figure 1). The Infini-attention incorporates a compressive memory into the vanilla attention mechanism (Bahdanau et al., 2014; Vaswani et al., 2017) and builds in both masked local attention and long-term linear attention mechanisms in a single Transformer block.

Such a subtle but critical modification to the Transformer attention layer enables a natural extension of existing LLMs to infinitely long contexts via continual pre-training and fine-tuning.

Our Infini-attention reuses all the key, value and query states of the standard attention computation for long-term memory consolidation and retrieval. We store old KV states of the attention in the compressive memory, instead of discarding them like in the standard attention mechanism. We then retrieve the values from the memory by using the attention query states when processing subsequent sequences. To compute the final contextual output, the Infini-attention aggregates the long-term memory-retrieved values and the local attention contexts.

In our experiments, we show that our approach outperforms baseline models on long-context language modeling benchmarks while having 114x comprehension ratio in terms of memory size. The model achieves even better perplexity when trained with 100K sequence length. A 1B LLM naturally scales to 1M sequence length and solves the passkey retrieval task when injected with Infini-attention. Finally, we show that a 8B model with Infini-attention reaches a new SOTA result on a 500K length book summarization task after continual pre-training and task fine-tuning.

In summary, our work makes the following contributions:

1. We introduce a practical and yet powerful attention mechanism – Infini-attention with long-term compressive memory and local causal attention for efficiently modeling both long and short-range contextual dependencies.

2. Infini-attention introduces minimal change to the standard scaled dot-product attention and supports plug-and-play continual pre-training and long-context adaptation by design.

3. Our approach enables Transformer LLMs to scale to infinitely long context with a bounded memory and compute resource by processing extremely long inputs in a streaming fashion.

2 Method

Figure 2 compares our model, Infini-Transformer, and Transformer-XL (Dai et al., 2019). Similar to Transformer-XL, Infini-Transformer operates on a sequence of segments. We compute the standard causal dot-product attention context within each segment. So the dot-product attention computation is local in a sense that it covers a total N number of tokens of the current segment with index S (N is the segment length).

The local attention (Dai et al., 2019), however, discards the attention states of the previous segment when processing the next one. In Infini-Transformers, instead of leaving out the old KV attention states, we propose to reuse them to maintain the entire context history with a compressive memory. So each attention layer of Infini-Transformers has both global compressive and local fine-grained states. We call such an efficient attention mechanism Infini-attention, which is illustrated in Figure 1 and described formally in the following sections.

2.1 Infini-attention

As shown Figure 1, our Infini-attention computes both local and global context states and combine them for its output. Similar to multi-head attention (MHA), it maintains H number
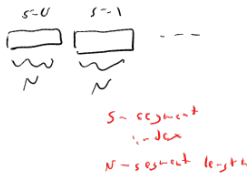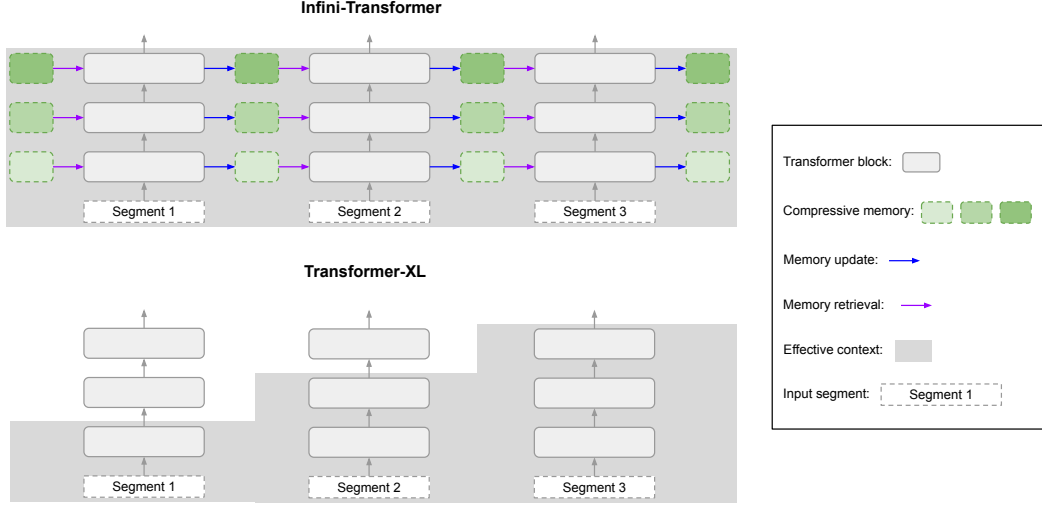
Figure 2: Infini-Transformer (top) has an entire context history whereas Transformer-XL (bottom) discards old contexts since it caches the KV states for the last segment only.

of parallel compressive memory per attention layer (H is the number of attention heads) in addition to the dot-product attention.

2.1.1 Scaled Dot-product Attention

The multi-head scaled dot-product attention (Vaswani et al., 2017), specially its self-attention variant (Munkhdalai et al., 2016; Cheng et al., 2016), has been the main building block in LLMs. The MHA's strong capability to model context-dependent dynamic computation and its conveniences of temporal masking have been leveraged extensively in the autoregressive generative models.

A single head in the vanilla MHA computes its attention context $A_{dot} \in \mathbb{R}^{N \times d_{value}}$ from sequence of input segments $X \in \mathbb{R}^{N \times d_{model}}$ as follows. First, it computes attention query, key, and value states:

$$K = XW_K, \ V = XW_V \text{ and } Q = XW_Q. \tag{1}$$

Here, $W_K \in \mathbb{R}^{d_{model} \times d_{key}}$, $W_V \in \mathbb{R}^{d_{model} \times d_{value}}$ and $W_Q \in \mathbb{R}^{d_{model} \times d_{key}}$ are trainable projection matrices. Then, the attention context is calculated as a weighted average of all other values as

$$A_{dot} = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{model}}}\right) V. \tag{2}$$

For MHA, we compute H number of attention context vectors for each sequence element in parallel, concatenate them along the second dimension and then finally project the concatenated vector to the model space to obtain attention the output.

2.1.2 Compressive Memory

In Infini-attention, instead of computing new memory entries for compressive memory, we reuse the query, key and value states (Q, K and V) from the dot-product attention computation. The state sharing and reusing between the dot-product attention and compressive memory not only enables efficient plug-in-play long-context adaptation but also speeds up training and inference. Similar to the prior work (Munkhdalai et al., 2019), our goal is to store bindings of key and value states in the compressive memory and retrieve by using the query vectors.

3

While there are different forms of compressive memory proposed in the literature (Hopfield, 1982; Kanerva, 1988; Schlag et al., 2019; Munkhdalai et al., 2019), for simplicity and computational efficiency, in this work we parameterize the memory with an associative matrix (Schlag et al., 2020). This approach further allows us to cast the memory update and retrieval process as linear attention mechanism (Shen et al., 2018) and to leverage stable training techniques from the related methods. Specially, we adopt the update rule and retrieval mechanism by Katharopoulos et al. (2020) mainly due to its simplicity and competitive performance.

Memory retrieval. In Infini-attention, we retrieve new content $A_{mem} \in \mathbb{R}^{N \times d_{value}}$ from the memory $M_{s-1} \in \mathbb{R}^{d_{key} \times d_{value}}$ by using the query $Q \in \mathbb{R}^{N \times d_{key}}$ as:

$$A_{mem} = \frac{\sigma(Q)M_{s-1}}{\sigma(Q)z_{s-1}}. \qquad = \frac{\sigma(Q)\left[M_{s-1} + \sigma(K^T)V\right]}{\sigma(Q)\left[z_s + \sum_{t=1}^{N}\sigma(k_t)\right]} \qquad (3)$$

Here, σ and $z_{s-1} \in \mathbb{R}^{d_{key}}$ are a nonlinear activation function and a normalization term, respectively. As the choice of the non-linearity and the norm method is crucial for training stability, following Katharopoulos et al. (2020) we record a sum over all keys as the normalization term z_{s-1} and use element-wise ELU + 1 as the activation function (Clevert et al., 2015).

Memory update. Once the retrieval is done, we update the memory and the normalization term with the new KV entries and obtain the next states as

$$M_s \leftarrow M_{s-1} + \sigma(K)^T V \text{ and } z_s \leftarrow z_{s-1} + \sum_{t=1}^{N}\sigma(K_t). \qquad (4)$$

The new memory states M_s and z_s are then passed to the next segment $S + 1$, building in a recurrence in each attention layer. The right side term $\sigma(K)^T V$ in Eq. (4) is known as an associative binding operator (Smolensky, 1990; Hebb, 2005; Schlag et al., 2020).

Inspired by the success of delta rule (Munkhdalai et al., 2019; Schlag et al., 2020; 2021), we have also incorporated it into our Infini-attention. The delta rule attempts a slightly improved memory update by first retrieving existing value entries and subtracting them from the new values before applying the associative bindings as new update.

$$M_s \leftarrow M_{s-1} + \sigma(K)^T \left(V - \frac{\sigma(K)M_{s-1}}{\sigma(K)z_{s-1}}\right). \qquad (5)$$

This update rule (*Linear + Delta*) leaves the associative matrix unmodified if the KV binding already exists in the memory while still tracking the same normalization term as the former one (*Linear*) for numerical stability.

Long-term context injection. We aggregate the local attention state A_{dot} and memory retrieved content A_{mem} via a learned gating scalar β:

$$A = sigmoid(\beta) \odot A_{mem} + (1 - sigmoid(\beta)) \odot A_{dot}. \qquad (6)$$

This adds only a single scalar value as training parameter per head while allowing a learnable trade-off between the long-term and local information flows in the model (Wu et al., 2022).

Similar to the standard MHA, for the multi-head Infini-attention we compute H number of context states in parallel, and concatenate and project them for the final attention output $O \in \mathbb{R}^{N \times d_{model}}$:

$$O = [A^1; \ldots A^H]W_O \qquad (7)$$

where $W_O \in \mathbb{R}^{H \times d_{value} \times d_{model}}$ is trainable weights.

2.2 Memory and Effective Context Window

Our Infini-Transformer enables an unbounded context window with a bounded memory footprint. To illustrate this, Table 1 lists the previous segment-level memory models with

Model	Memory (cache) footprint	Context length	Memory update	Memory retrieval
Transformer-XL	$(d_{key} + d_{value}) \times H \times N \times l$	$N \times l$	Discarded	Dot-product attention
Compressive Transformer	$d_{model} \times (c + N) \times l$	$(c \times r + N) \times l$	Discarded	Dot-product attention
Memorizing Transformers	$(d_{key} + d_{value}) \times H \times N \times S$	$N \times S$	None	kNN + dot-product attention
RMT	$d_{model} \times p \times l \times 2$	$N \times S$	Discarded	Soft-prompt input
AutoCompressors	$d_{model} \times p \times (m + 1) \times l$	$N \times S$	Discarded	Soft-prompt input
Infini-Transformers	$d_{key} \times (d_{value} + 1) \times H \times l$	$N \times S$	Incremental	Linear attention

Table 1: Transformer models with segment-level memory are compared. For each model, the memory size and effective context length are defined in terms of their model parameters (N: input segment length, S: the number of segments, l: the number of layers, H: the number of attention heads, c: Compressive Transformer memory size, r: compression ratio, p: the number of soft-prompt summary vectors and m: summary vector accumulation steps).

their context-memory footprint and effective context length defined in terms of model parameters and input segment length. Infini-Transformer has a constant memory complexity of $d_{key} \times d_{value} + d_{key}$ for storing compressed context in M_s and z_s for each head in single layer while for the other models, the complexity grows along with the sequence dimension - the memory complexity depends either on the cache size for Transformer-XL (Dai et al., 2019), Compressive Transformer (Rae et al., 2019) and Memorizing Transformers (Wu et al., 2022) or on the soft-prompt size for RTM (Bulatov et al., 2022) and AutoCompressors (Ge et al., 2023).

Transformer-XL computes attention over KV states cached from the last segment in addition to the current states. Since this is done for each layer, Transformer-XL extends the context window from N to $N \times l$ tokens with an additional memory footprint of $(d_{key} + d_{value}) \times H \times N \times l$. Compressive Transformer adds a second cache to Transformer-XL and stores compressed representations of past segment activations. So it extends the Transformer-XL's context window by $c \times r \times l$ but still has a large context-memory complexity. Taking the idea further, Memorizing Transformers opt to store the entire KV states as context for input sequences. Since the storage becomes prohibitively expensive in this case, they restrict the contextual computation to a single layer only. By utilizing a fast kNN retriever, Memorizing Transformers then build a context window covering the entire sequence history of length $N \times S$ at an increased cost of storage. Our experiments show that Infini-Transformer LM can achieve more than 100x compression rate on top of Memorizing Transformers while further improving the perplexity score.

Figure 3: There are two types of heads emerged in Infini-attention after training: specialized heads with gating score near 0 or 1 and mixer heads with score close to 0.5. The specialized heads either process contextual information via the local attention mechanism or retrieve from the compressive memory whereas the mixer heads aggregate both current contextual information and long-term memory content together into single output.

RMT and AutoCompressors allow for a potentially infinite context length since they compress the input into summary vectors and then pass them as extra soft-prompt inputs for the subsequent segments. However, in practice the success of those techniques highly depends on the size of soft-prompt vectors. Namely, it is necessary to increase the number of soft-prompt (summary) vectors to achieve a better performance with AutoCompressors (Chevalier et al., 2023) and with that, the memory and compute complexity grow quickly resulting in diminished efficiency. It was also observed in AutoCompressors (Chevalier et al., 2023) that an efficient compression objective is needed for training such prompt compression techniques (Ge et al., 2023).
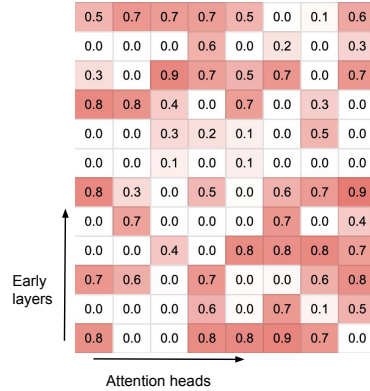
PPL

Model	Memory size (comp.)	XL cache	Segment length	PG19	Arxiv-math
Transformer-XL	50M (3.7x)	2048	2048	11.88	2.42
Memorizing Transformers	183M (1x)	2048	2048	11.37	2.26
RMT	2.5M (73x)	None	2048	13.27	2.55
Infini-Transformer (Linear)	1.6M (114x)	None	2048	**9.65**	2.24
Infini-Transformer (Linear + Delta)	1.6M (114x)	None	2048	9.67	**2.23**

Table 2: Long-context language modeling results are compared in terms of average token-level perplexity. Comp. denotes compression ratio. Infini-Transformer outperforms memorizing transformers with memory length of 65K and achieves 114x compression ratio.

3 Experiments

We evaluated our Infini-Transformer models on benchmarks involving extremely long input sequences: long-context language modeling, 1M length passkey context block retrieval and 500K length book summarization tasks. For the language modeling benchmark, we train our models from scratch while for the passkey and book summarization tasks, we continually pre-train existing LLMs in order to highlight a plug-and-play long-context adaptation capability of our approach.

3.1 Long-context Language Modeling

We trained and evaluated small Infini-Transformer models on PG19 (Rae et al., 2019) and Arxiv-math (Wu et al., 2022) benchmarks. Our setup closely resembles that of Memorizing Transformers (Wu et al., 2022). Namely, all our models have 12 layers and 8 attention heads of dimension 128 each and FFNs with hidden layer 4096.

We set the Infini-attention segment length N to 2048 for all attention layers and the input sequence length to 32768 for training. This allows the Infini-attention to unroll over 16 steps w.r.t its compressive memory states. For the RMT baseline, we performed several runs with summary prompt lengths 50, 100 and 150 and sequence lengths 4096, 8196 and 32768. RMT with 100 summary vectors gave the best result when trained on 8196 length sequences.

The main results from the language modeling experiments are summarized in Table 2. Our Infini-Transformer outperforms both Transformer-XL (Dai et al., 2019) and Memorizing Transformers (Wu et al., 2022) baselines while maintaining 114x less memory parameters than the Memorizing Transformer model with a vector retrieval-based KV memory with length of 65K at its 9th layer.

100K length training. We further increased the training sequence length to 100K from 32K and trained the models on Arxiv-math dataset. 100K training further decreased the perplexity score to **2.21** and **2.20** for *Linear* and *Linear + Delta* models.

Gating score visualization. Figure 3 visualizes the gating score, $sigmoid(\beta)$ for the compressive memory for all attention heads in each layer. There are two types of heads emerged in Infini-attention after training: specialized heads with a gating score near 0 or 1 and mixer heads with a score close to 0.5. The specialized heads either process contextual information via the local attention computation or retrieve from the compressive memory whereas the

	Zero-shot				
	32K	128K	256K	512K	1M
Infini-Transformer (Linear)	14/13/98	11/14/100	6/3/100	6/7/99	8/6/98
Infini-Transformer (Linear + Delta)	13/11/99	6/9/99	7/5/99	6/8/97	7/6/97
	FT (400 steps)				
Infini-Transformer (Linear)	100/100/100	100/100/100	100/100/100	97/99/100	96/94/100
Infini-Transformer (Linear + Delta)	100/100/100	100/100/99	100/100/99	100/100/100	100/100/100

Table 3: Infini-Transformers solved the passkey task with up to 1M context length when fine-tuned on 5K length inputs. We report token-level retrieval accuracy for passkeys hidden in a different part (*start/middle/end*) of long inputs with lengths 32K to 1M.

Model	Rouge-1	Rouge-2	Rouge-L	Overall
BART	36.4	7.6	15.3	16.2
BART + Unlimiformer	36.8	8.3	15.7	16.9
PRIMERA	38.6	7.2	15.6	16.3
PRIMERA + Unlimiformer	37.9	8.2	16.3	17.2
Infini-Transformers (Linear)	37.9	8.7	17.6	18.0
Infini-Transformers (Linear + Delta)	**40.0**	**8.8**	**17.9**	**18.5**

Table 4: 500K length book summarization (BookSum) results. The BART, PRIMERA and Unlimiformer results are from Bertsch et al. (2024).

mixer heads aggregate both current contextual information and long-term memory content together into a single output. Interestingly, each layer has at least a single short-range head, allowing a forward-propagation of input signal up until the output layer. We also observed an interleaving of long and short-term content retrievals throughout the forward computation.

3.2 LLM Continual Pre-training

We performed a lightweight continual pre-training for long-context adaptation of existing LLMs. The pre-training data includes the PG19 and Arxiv-math corpus as well as C4 text (Raffel et al., 2020) with length more than 4K tokens. The segment length N was set to 2K throughout our experiments.

1M passkey retrieval benchmark. We replaced the vanilla MHA in a 1B LLM with Infini-attention and continued to pre-train on inputs with length of 4K. The model was trained for 30K steps with batch size of 64 before fine-tuning on the passkey retrieval task (Mohtashami & Jaggi, 2024).

The passkey task hides a random number into a long text and asks it back at the model output. The length of the distraction text is varied by repeating a text chunk multiple times. The previous work (Chen et al., 2023a) showed that a 8B LLaMA model can solve the task up to 32K length when fine-tuned with the same 32K length inputs with Position Interpolation. We take this challenge further and fine-tune on only 5K length inputs to test on 1M length regime.

Table 3 reports the token-level accuracy for test subsets with input lengths ranging from 32K to 1M. For each test subset, we controlled the position of the passkey so that it is either located around the beginning, middle or the end of the input sequence. We reported both zero-shot accuracy and fine-tuning accuracy. Infini-Transformers solved the task with up to 1M context length after fine-tuning on 5K length inputs for 400 steps.

500K length book summarization (Book-Sum). We further scaled our approach by continuously pre-training a 8B LLM model with 8K input length for 30K steps. We then fine-tuned on a book summarization task, BookSum (Kryściński et al., 2021) where the goal is to generate a summary of an entire book text.

Figure 4: Infini-Transformers obtain better Rouge overall scores with more book text provided as input.

We set the input length to 32K for fine-tuning and increase to 500K for evaluating. We use a generation temperature of 0.5 and $top_p = 0.95$ and set the number of decoding steps to 1024 to generate a summary of each book.
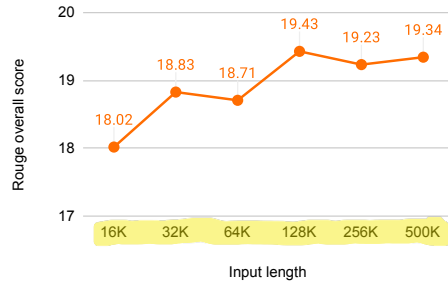
Table 4 compares our model against the encoder-decoder models that were built particularly for the summarization task (Lewis et al., 2019; Xiao et al., 2021) and their retrieval-based long-context extension (Bertsch et al., 2024). Our model outperforms the previous best results and achieves a new SOTA on BookSum by processing the entire text from book. We have also plotted the overall Rouge score on validation split of BookSum data in Figure 4. There is a clear trend showing that with more text provided as input from books, Our Infini-Transformers improves its summarization performance metric.

4 Related Work

Compressive memory. Inspired by the plasticity in biological neurons (Munkhdalai & Yu, 2017a; Miconi et al., 2018), compressive memory approaches cast parameterized functions as memory to store and retrieve information (Hinton & Plaut, 1987; Schmidhuber, 1992; Ba et al., 2016; Munkhdalai et al., 2019). Unlike the Transformer KV memory array (Vaswani et al., 2017; Wu et al., 2022), which grows with input sequence length, compressive memory systems maintain a constant number of memory parameters for computational efficiency. The parameters are modified with an update rule to store information, which is then retrieved via a memory reading mechanism (Graves et al., 2014; Sukhbaatar et al., 2015; Munkhdalai & Yu, 2017b).

Compressed input representations can be viewed as a summary of past sequence segments (Rae et al., 2019; Chevalier et al., 2023). Along this direction, more recent works have been utilizing a Transformer LLM itself to compress input sequence for efficient long-context modeling (Bulatov et al., 2022; Chevalier et al., 2023; Ge et al., 2023; Mu et al., 2024). However, the previous segment-level compression methods, including Compressive Transformers (Rae et al., 2019) still discard the memory entries of old segments in order to free up space for the new ones, limiting their context window to the most recent segments. This is in contrast to our Infini-attention that computes incremental memory updates to a fixed amount of memory parameters in a recurrent fashion.

Long-context continual pre-training. There is a line of work that extends the do-product attention layers and continues to train LLMs for long-context (Xiong et al., 2023; Fu et al., 2024). The attention extensions include incorporating sparsity into the attention layer (Chen et al., 2023b; Ratner et al., 2022; Mohtashami & Jaggi, 2024) as well as manipulating the position encodings (Chen et al., 2023a; Peng et al., 2023) Although the position encoding-based methods such as position interpolation techniques (Chen et al., 2023a) can be data efficient as they only adjust the positional bias in the attention layer, they are still costly for inference.

The attention mechanism is also prone to the issues of attention sink (Xiao et al., 2023) and lost-in-the-middle (Liu et al., 2024). Consequently, they struggle in a regime where context length is longer than what was observed during training (Press et al., 2021; Kazemnejad et al., 2024). The proposed Infini-attention addresses those issues by enabling a segment-level streaming computation over long sequences with a fixed local attention window. Our Infini-Transformers successfully extrapolate to 1M input length regimes when trained on 32K and even 5K length sequences.

Efficient attention. The efficient attention techniques attempt to improve the efficiency of the dot-product attention with an approximation or a system-level optimization. Multiple directions have been explored for different forms of efficient attention approximation, including sparsity-based (Child et al., 2019; Beltagy et al., 2020; Sukhbaatar et al., 2021; Ding et al., 2023) and linear attention approximation (Shen et al., 2018; Katharopoulos et al., 2020; Schlag et al., 2021). Among those, the linear attention variants are closely related to the associative memory matrix (Schlag et al., 2020; 2021) and the metalearned neural memory (Munkhdalai et al., 2019), where KV bindings (Smolensky, 1990) are stored in Fast-Weights (Hinton & Plaut, 1987; Schmidhuber, 1992; Ba et al., 2016) that are modified in with respect to new contextual information. More recently, system-level optimization techniques have been proposed by leveraging specific hardware architecture to make the exact attention computation more efficient (Dao et al., 2022; Liu et al., 2023).

5   Conclusion

An effective memory system is crucial not just for comprehending long contexts with LLMs, but also for reasoning, planning, continual adaptation for fresh knowledge, and even for learning how to learn. This work introduces a close integration of compressive memory module into the vanilla dot-product attention layer. This subtle but critical modification to the attention layer enables LLMs to process infinitely long contexts with bounded memory and computation resources. We show that our approach can naturally scale to a million length regime of input sequences, while outperforming the baselines on long-context language modeling benchmark and book summarization tasks. We also demonstrate a promising length generalization capability of our approach. 1B model that was fine-tuned on up to 5K sequence length passkey instances solved the 1M length problem.

References

Rohan Anil, Andrew M Dai, Orhan Firat, Melvin Johnson, Dmitry Lepikhin, Alexandre Passos, Siamak Shakeri, Emanuel Taropa, Paige Bailey, Zhifeng Chen, et al. Palm 2 technical report. *arXiv preprint arXiv:2305.10403*, 2023.

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. *Advances in neural information processing systems*, 29, 2016.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document trans-former. *arXiv preprint arXiv:2004.05150*, 2020.

Amanda Bertsch, Uri Alon, Graham Neubig, and Matthew Gormley. Unlimiformer: Long-range transformers with unlimited length input. *Advances in Neural Information Processing Systems*, 36, 2024.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Aydar Bulatov, Yury Kuratov, and Mikhail Burtsev. Recurrent memory transformer. *Advances in Neural Information Processing Systems*, 35:11079–11091, 2022.

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

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

Jianpeng Cheng, Li Dong, and Mirella Lapata. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733*, 2016.

Alexis Chevalier, Alexander Wettig, Anirudh Ajith, and Danqi Chen. Adapting language models to compress contexts. *arXiv preprint arXiv:2305.14788*, 2023.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*, 2019.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35:16344–16359, 2022.

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

Yao Fu, Rameswar Panda, Xinyao Niu, Xiang Yue, Hannaneh Hajishirzi, Yoon Kim, and Hao Peng. Data engineering for scaling language models to 128k context. *arXiv preprint arXiv:2402.10171*, 2024.

Tao Ge, Jing Hu, Xun Wang, Si-Qing Chen, and Furu Wei. In-context autoencoder for context compression in a large language model. *arXiv preprint arXiv:2307.06945*, 2023.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, et al. Olmo: Accelerating the science of language models. *arXiv preprint arXiv:2402.00838*, 2024.

Donald Olding Hebb. *The organization of behavior: A neuropsychological theory*. Psychology press, 2005.

Geoffrey E Hinton and David C Plaut. Using fast weights to deblur old memories. In *Proceedings of the ninth annual conference of the Cognitive Science Society*, pp. 177–186, 1987.

John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

Pentti Kanerva. *Sparse distributed memory*. MIT press, 1988.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pp. 5156–5165. PMLR, 2020.

Amirhossein Kazemnejad, Inkit Padhi, Karthikeyan Natesan Ramamurthy, Payel Das, and Siva Reddy. The impact of positional encoding on length generalization in transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

Wojciech Kryściński, Nazneen Rajani, Divyansh Agarwal, Caiming Xiong, and Dragomir Radev. Booksum: A collection of datasets for long-form narrative summarization. *arXiv preprint arXiv:2105.08209*, 2021.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Hao Liu, Matei Zaharia, and Pieter Abbeel. Ring attention with blockwise transformers for near-infinite context. *arXiv preprint arXiv:2310.01889*, 2023.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.

Thomas Miconi, Kenneth Stanley, and Jeff Clune. Differentiable plasticity: training plastic neural networks with backpropagation. In *International Conference on Machine Learning*, pp. 3559–3568. PMLR, 2018.

Amirkeivan Mohtashami and Martin Jaggi. Random-access infinite context length for transformers. *Advances in Neural Information Processing Systems*, 36, 2024.

Jesse Mu, Xiang Li, and Noah Goodman. Learning to compress prompts with gist tokens. *Advances in Neural Information Processing Systems*, 36, 2024.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *International conference on machine learning*, pp. 2554–2563. PMLR, 2017a.

Tsendsuren Munkhdalai and Hong Yu. Neural semantic encoders. In *Proceedings of the conference. Association for Computational Linguistics. Meeting*, volume 1, pp. 397. NIH Public Access, 2017b.

Tsendsuren Munkhdalai, John P Lalor, and Hong Yu. Citation analysis with neural attention models. In *Proceedings of the Seventh International Workshop on Health Text Mining and Information Analysis*, pp. 69–77, 2016.

Tsendsuren Munkhdalai, Alessandro Sordoni, Tong Wang, and Adam Trischler. Metalearned neural memory. *Advances in Neural Information Processing Systems*, 32, 2019.

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

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. *Proceedings of Machine Learning and Systems*, 5, 2023.

Ofir Press, Noah A Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *arXiv preprint arXiv:2108.12409*, 2021.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*, 2019.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.

Nir Ratner, Yoav Levine, Yonatan Belinkov, Ori Ram, Omri Abend, Ehud Karpas, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. Parallel context windows improve in-context learning of large language models. *arXiv preprint arXiv:2212.10947*, 2022.

Imanol Schlag, Paul Smolensky, Roland Fernandez, Nebojsa Jojic, Jürgen Schmidhuber, and Jianfeng Gao. Enhancing the transformer with explicit relational encoding for math problem solving. *arXiv preprint arXiv:1910.06611*, 2019.

Imanol Schlag, Tsendsuren Munkhdalai, and Jürgen Schmidhuber. Learning associative inference using fast weight memory. *arXiv preprint arXiv:2011.07831*, 2020.

Imanol Schlag, Kazuki Irie, and Jürgen Schmidhuber. Linear transformers are secretly fast weight programmers. In *International Conference on Machine Learning*, pp. 9355–9366. PMLR, 2021.

Jürgen Schmidhuber. Learning to control fast-weight memories: An alternative to dynamic recurrent networks. *Neural Computation*, 4(1):131–139, 1992.

Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Zhuoran Shen, Mingyuan Zhang, Haiyu Zhao, Shuai Yi, and Hongsheng Li. Efficient attention: Attention with linear complexities. *arXiv preprint arXiv:1812.01243*, 2018.

Paul Smolensky. Tensor product variable binding and the representation of symbolic structures in connectionist systems. *Artificial intelligence*, 46(1-2):159–216, 1990.

Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. End-to-end memory networks. *Advances in neural information processing systems*, 28, 2015.

Sainbayar Sukhbaatar, Da Ju, Spencer Poff, Stephen Roller, Arthur Szlam, Jason Weston, and Angela Fan. Not all memories are created equal: Learning to forget by expiring. In *International Conference on Machine Learning*, pp. 9902–9912. PMLR, 2021.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Yuhuai Wu, Markus N Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. *arXiv preprint arXiv:2203.08913*, 2022.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*, 2023.

Wen Xiao, Iz Beltagy, Giuseppe Carenini, and Arman Cohan. Primera: Pyramid-based masked sentence pre-training for multi-document summarization. *arXiv preprint arXiv:2110.08499*, 2021.

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

A   Additional Training Details

For the long-context language modeling task, we set the learning rate to 0.01 by performing small search over values of 0.003, 0.005, 0.01 and 0.03. We used the Adafactor optimizer (Shazeer & Stern, 2018) with linear warmup with 1000 steps, followed by cosine decay. We applied gradient checkpointing after each segment to save to save memory. The batch size was set to 64. For the LLM experiments, we set the learning rate to 0.0001 during continual pre-training and task fine-tuning.

B   Passkey Retrieval Task

Below we showed the input format of the passkey task.

There is an important info hidden inside a lot of irrelevant text. Find it and memorize them. I will quiz you about the important information there. The grass is green. The sky is blue. The sun is yellow. Here we go. There and back again. (repeat x times) The pass key is **9054**. Remember it. **9054** is the pass key. The grass is green. The sky is blue. The sun is yellow. Here we go. There and ack again. (repeat y times) What is the pass key? The pass key is