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

April 23, 2024

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

Tsendsuren Munkhdalai et al., Google

https://arxiv.org/abs/2404.07143

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

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

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

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.

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.

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.

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.

Infini-attention overview

- Addresses long context scaling with transformers
- Seems to combine 3 concepts:
  - Transformer XL divided text into segments and processed two consecutive segments at a time
  - Fast Weight Memory (coauthored by Munkhdalai) creates a compressive, associative memory using matrix multiplication
    - This associative memory uses keys to store values
    - Here (I think) queries are used during retrieval, implicitly performing a similarity function
  - Linear Attention approximated attention with a recurrent calculation that afforded linear scaling
- The result is memory unit parameter count that is constant and processing cost that is linear with respect to the input length

Infini-attention method

- Transformer-XL processed two actual input segments at a time, caching the previous segment and loading the current segment
- I believe the key difference here is that we replace the previous segment with a linearized attention on keys stored in our memory
- The new loop looks like this:
  - Run regular QKV attention on the current segment
  - Retrieve old memory contents using Queries, performing linear attention
  - The above two results are weighted summed with a learned parameter β
  - The result is output and also performs a memory unit update using Keys to store Values
  - The cumulative, updated memory is ready to be used with the next segment

Infini-attention figures [1]

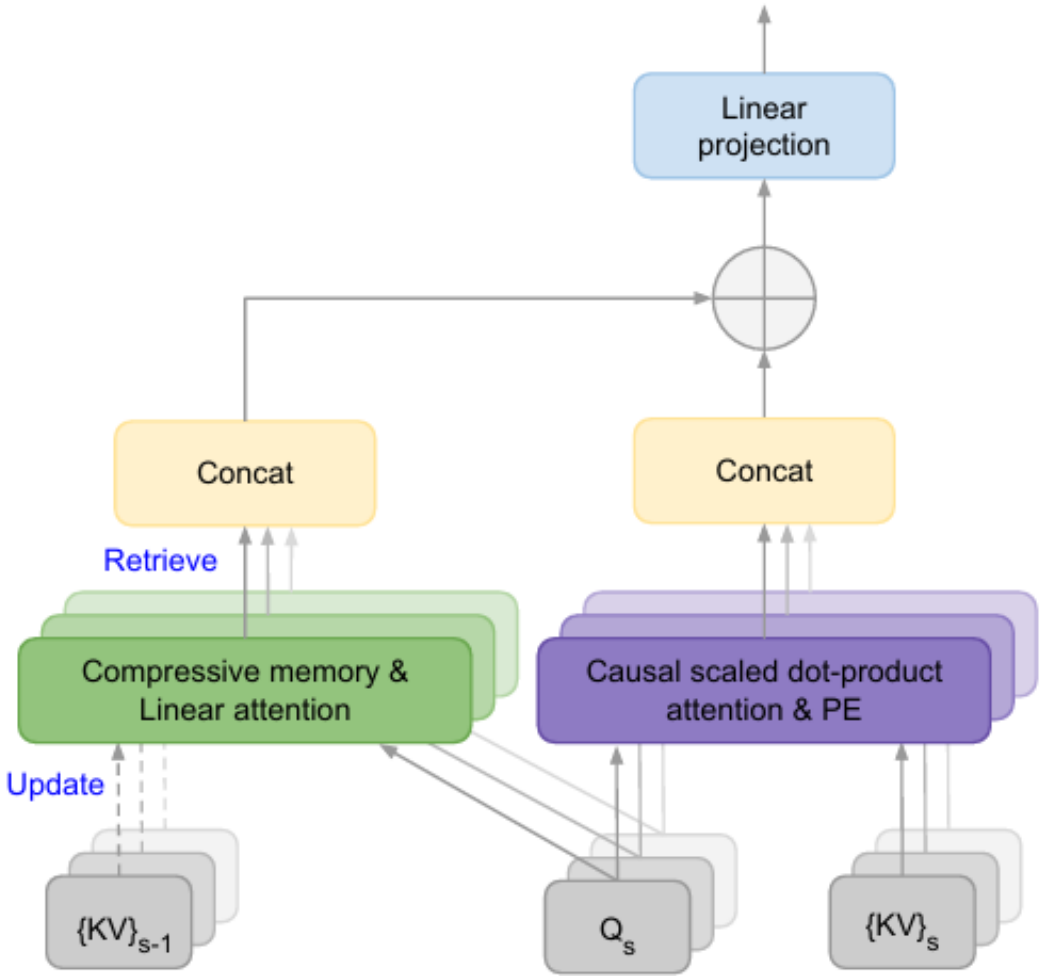
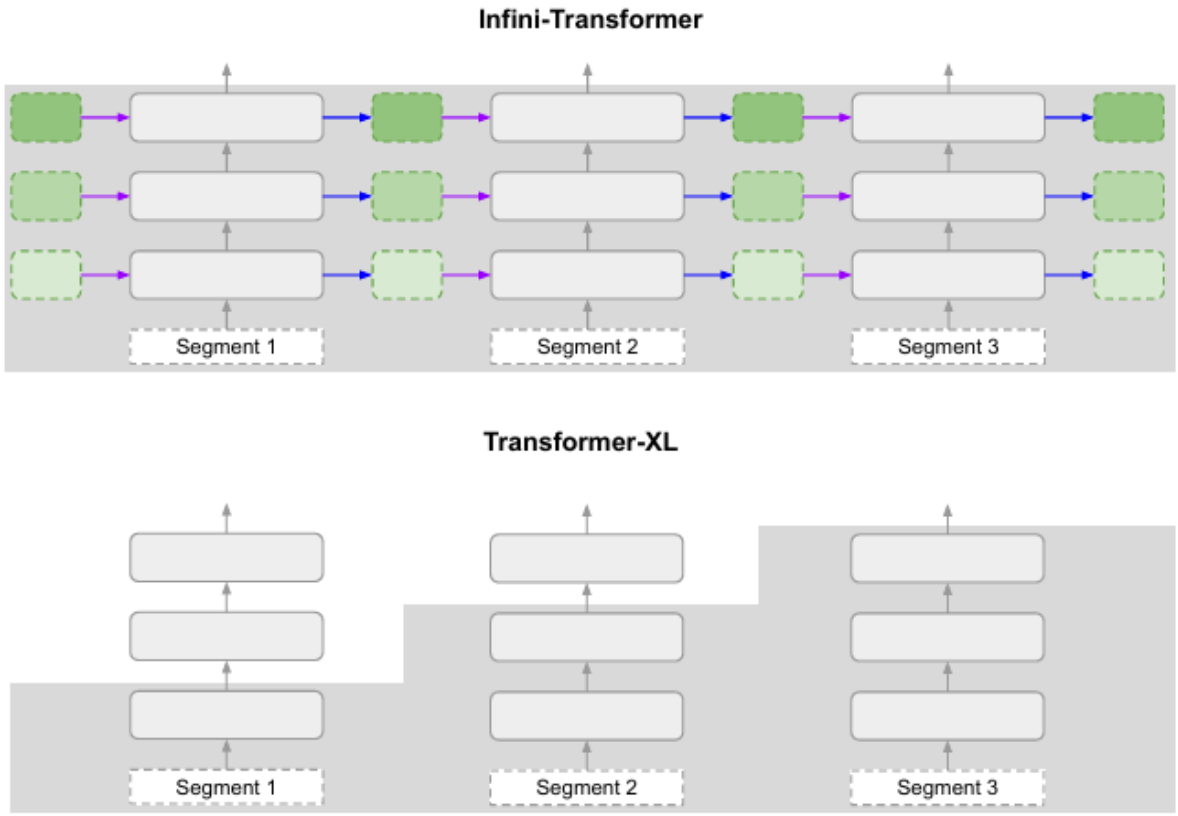


Figure 1: Infini-attention has an additional compressive memory with linear attention for processing infinitely long contexts. $\{KV\}_{s-1}$ and $\{KV\}_s$ are attention key and values for current and previous input segments, respectively and Q_s the attention queries. PE denotes position embeddings.

Infini-attention figures [2]



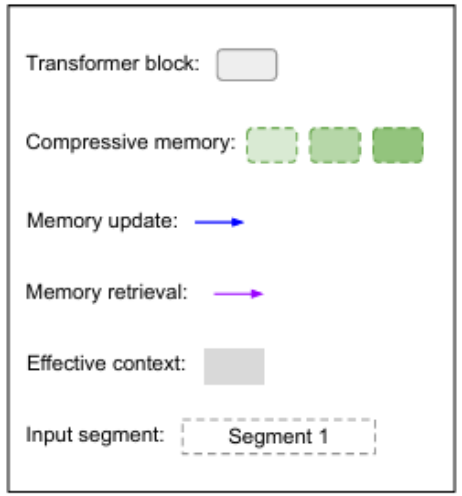
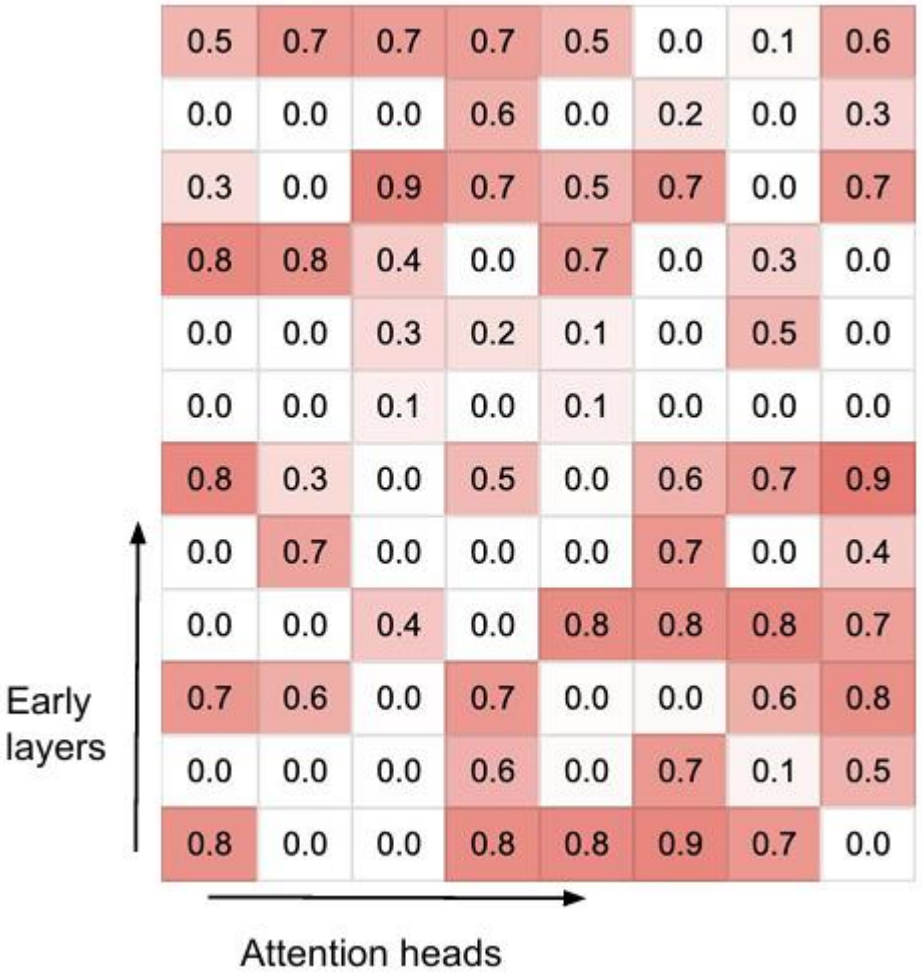


Figure 2: Infini-Transformer (top) has an entire context history whereas Transformer-XL (bottom) discards old contexts since it caches the KV states for the last segment only.

Infini-attention figures [3]



Values of sigmoid(β), where the β parameter is trained separately for each head of each layer

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.

Infini-attention conclusion

- Transformer-XL had a simple scheme to break text into segments and process two at a time, but it didn't perform that well
  - Information from 1st segment would propagate to the Lth segment in layer L
- Linear Attention Transformer had a nice linear scaling, but didn't perform that well
- Infini-attention packs way more information into the previous segment than Transformer-XL
  - By using a memory, information from 1st segment is immediately available to all later segments in the same layer (to the extent that the memory and linear attention can work as well as full attention)
- The associative memory and linear attention have good scaling
  - Memory is constant, regardless of sequence length
  - Compute is linear with respect to sequence length

References

- Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context
  Zihang Dai et al. (2019)
  https://arxiv.org/abs/1901.02860
- Learning Associative Inference Using Fast Weight Memory
  Imanol Schlag et al. (2020)
  https://arxiv.org/abs/2011.07831
- Transformers are RNNs: Fast Autoregressive Transformers with Linear Attention
  Angelos Katharopoulos et al. (2020)
  https://arxiv.org/abs/2006.16236