Memory mapped Memory for transformers

Bastian Apportin[*]
Joomo GmbH

July 1, 2025

Abstract

The success of LLM's is mainly based on the astonishing capabilities of the Transformer Architecture. Which does a great job in catching long range dependencies.

The positional encoding used in Transformers solves the problem of conbtext awarness, but limits the distance a transformer can look. What we want is aktually to keep the context awarness of the architecture, like distance memories coming back, influencing your current thoughts.

We've tried a lot of different aproaches like ALiBi or placing cnn's in the key/query calculations, but none of them worked like we expected. All the limited experiments we did worked significantly better when using RoPE like positin encodings. But maybe, we don't need to remove them.

The RoPE embedding can be seen as an offset plus content adressing. The offset is the rotation and the mixture of the frequencies can adress sharper or broader position back from the current position.

1 Introduction

The transformer architecture is self attention as its core. The worst and best part of that is, that there is no local context. Therefore most Transformer architectures apply some kind of positional encoding. This positional encoding solves the problem of the missing local context but introduces the problem that for long distances the 'context' or distance matters as well. Therefore the view of site can not be infinite, which effectivly limits memorizing things by that mechanism.

The RoPE embedding can be seen as an offset plus content adressing. The offset is the rotation and the mixture of the frequencies can adress sharper or broader position ranges back from the current position and the content are the original values of the queries and keys.

When we have positional adressing an idea came to mind, the memory mapped IO, where registers to access secondary hardware are mapped to main memory. One would write the block id of a harddrive to a memory position and then the content of that block will be mapped to a specific position in memory.

2 Methodology

we use the llama 3.2 1b model for our experiment, since that is a size we can still handle.

the llama model uses RoPE. RoPE implements an influence of relative distance. So it's not important what actual position(id) the token has. The result will be the same if we shift the wohle sequence.

We think of the current position as the anchor and place the mapped memories at a fixed negative offset. For ease of understanding we place the current position at 4096 and the memory map at 0. This way we

[*] Corresponding author: bastian.apportin@joomo.de

can store the unrotated keys and query them using a kind of LSH. The algorithm is an aproximation and the model ist not trained for that explicitly. We'll see if that actually works.

We can now store unrotated older key,value pairs in an LSH like strukture and search with the rotated queries. The queried results can be appended to the key and value tensors before the attention calculation. This way relevant memories from the storage can 'come to mind' and influence the generated sequence.

This mechanism can probably improved when the model is trained (another fine tuning step maybe) with that mechanism in place. In addition this mechanism can be used to memory map other mechanisms. For example we have a transformer model that generates limb movements from camera inputs and an llm to understand and generate text. We can now memory map those models into each other - potentially with a linear transformation in between. So the text model has a limited view on some key,values from the vision/limb model and vise versa.

For now we want to place memories in the storage and ask questions where the answer can show that knowledge from the memory was aktually used.

3 Experiments

To do useful experiments we will use fairy tales that we will put to the memory and a set of commands we ask the model like versions of 'tell me a story about ...' - define set of memories to use - define set of questions, that use this memories.

4 Results

5 Discussion

6 Conclusion

We successfully removed the positional encodings from the transformer architecture by adding cnn driven local context and thereby offered a way to have infinite context size. This can lead to large language models that do actually have a memory.

7 Related Work

Acknowledgments

A Appendix Title