Evaluating the Memory-Performance Tradeoffs Between Hopfield Networks and Standard Attention Mechanisms

Siddarth Ijju
si2462

May 5, 2025

Abstract

This project investigates the trade-offs between Hopfield networks and linear attention mechanisms, specifically in the handling of long-context memory tasks. Traditional attention mechanisms are limited in their capability to model long sequences, which Hopfield networks attempt to address. I evaluate the performance and memory capabilities of Hopfield attention across several benchmarks, namely LAMBADA, WikiText103 Word Retrieval, Long Range Arena, and my own custom-designed Memory Copying benchmark. I performed experiments with several types of models, specifically recurrent neural networks (RNNs), linear attention, and Hopfield attention networks, and evaluated each at several sequence lengths varying up to 8192 tokens. Although Hopfield attention may underperform compared to linear attention at short sequence lengths, it consistently excels at long-context tasks, particularly those requiring in-context retrieval and memory recall capabilities. Linear attention remains competitive, especially in tasks that emphasize semantic retrieval and structural understanding. This work provides insight into when and why Hopfield-based architectures may be preferred for tasks that require extended memory. The code can be found at: https://github.com/sidijju/Hopfield

Background

Attention models have been well-established as state of the art for language understanding tasks. However, these models are still largely limited by sequence length, and so have difficulty performing tasks requiring large amounts of memory. [Hua24] In this regard, several innovations have been made, but a particularly promising one is that of Hopfield networks. [Ram+21], which introduces a generalized version of the attention mechanism that closely aligns with Hopfield networks. Hopfield networks have shown strong capabilities in regards to memory, especially for tasks that involve in-context learning. [Hop82] The aim of this project was to provide insight into what factors allow Hopfield networks to outperform normal attention mechanism on certain tasks involving long-context memory. Specifically, I aim to study the limitations of the Hopfield network ability to store memory patterns in the context of in-context retrieval, semantic memory, and computational efficiency.

Implementation

All experiments were carried out on a NVIDIA A10 GPU with 30 virtual CPUs and 200 GB of RAM. The GPU was accessed through Lambda Labs, and runs were logged through Wandb. The implementation of my experiments was structured into two main sections.

Data

Many of the datasets used in my experiments were long-context, or in other words, the training samples exceeded the sequence length of the model (1024 at minimum) by a large margin. I used an overlapping method with truncation to ensure the model had context across samples and didn't encounter issues at sample boundaries during training. This procedure only was applied to training data - the testing data was usually well within the maximum sequence length of the models.

Next, I implemented each benchmark individually and model-agnostic. Since some of the benchmarks required different tasks (next-token prediction vs. sequence classification), I implemented multiple versions of training and evaluation methods for the benchmarks. Since I limited the number of layers for my models, generalization remained difficult. As such, the execution of each benchmark required both a training step and an evaluation step, in the belief that this would help to acquire more informative results.

For the LAMBADA benchmark, I implemented a simple next-token predictor with each backbone. The labels for each input were simply the inputs shifted by 1 token for this dataset, since the objective is to predict the final token. During training, I used CrossEntropy loss on all tokens, but the evaluation was only on the final token.

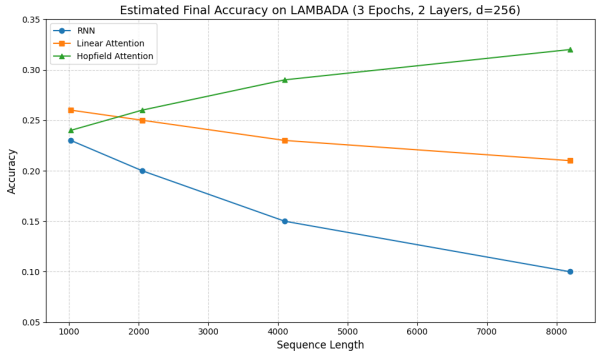
For the WikiText103 Word Retrieval, I implemented a masked token predictor using each backbone. Each input had certain tokens masked, and the labels were the unmasked version of the input with non-masked tokens included. During training, I used CrossEntropy loss on the masked tokens, and in this case evaluation remained the same. I extracted relevant named entities from the input text using spacy and masked tokens accordingly. I only kept samples and named entities where the named entity appeared more than once, and only masked the occurrences of the named entity after the first one, in order to properly assess the model's text retrieval capability given enough context.

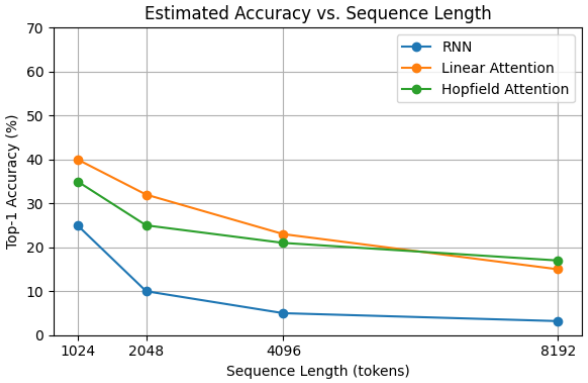
The Long Range Arena benchmark is split into several subtasks. I opted to use the Image, Listops, Pathfinder32, Retrieval, and Text subtasks. I did not perform experiments on the Pathfinder128 subtask since it requires special image patch based tokenization, and the Pathfinder32 subtask was suitable for the experiment.

For the Memory Copying benchmark, I dynamically generated data by generating random sequences of tokens for the input. Instead of memorizing the whole sequence, I repeated the sequence twice, with the task being to copy a portion of the first sequence exactly. Although this is a simple task at smaller sequence lengths, my hypothesis is that for longer sequence lengths, this task would become more challenging and would require specialized architectures related to memory.

Models

I implemented the various network backbones used in the benchmarking experiments by hand. My implementation of a recurrent neural network was standard and used PyTorch's internal implementation. I implemented basic linear attention (i.e. not flash attention from PyTorch) and Hopfield attention from scratch. For linear attention, I opted to use an ELU-based feature map. [Cho+22].





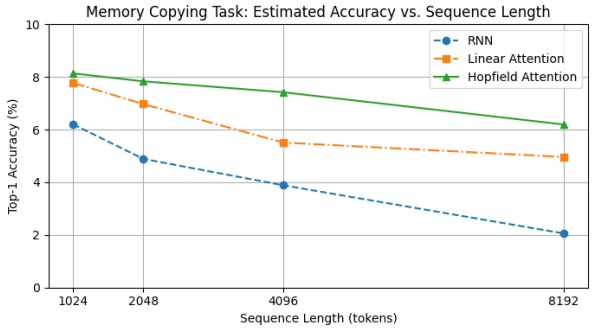
Each network was implemented as a generic backbone that could be used for both a next-token prediction task and a sequence classification task.

Results

I ran experiments on LAMBADA, WikiText103, Long Range Arena, and my own custom Memory Copying Benchmark. For each benchmark, I begin with a default experiment of 2 layers, 3 epochs of training, a hidden dimension of 256, a batch size of 4, and a sequence length of 1024 (this is the default sequence length for GPT2). I eventually ran experiments over sequence lengths of 1024, 2048, 4096, and 8192. There was a discrepancy in my accuracy calculations from my progress report. I rectified this error, which resulted in lower accuracy numbers overall, but with a similar pattern as before.

The LAMBADA dataset is primarily focused on long-context understanding. Since the task is to only predict the final word, this benchmark is relatively easier for the experiments I conducted, which showed in the final results. As you can see in the figure on the left above, RNN models struggled as the sequence length increased. Interestingly, at lower sequence lengths (i.e. 1024, the default sequence length for GPT2), linear attention outperformed Hopfield attention, which suggests that the benefits of Hopfield attention only become apparent at larger sequence lengths. We can also observe that as the sequence length increased, Hopfield attention models improved whereas both RNN models and linear attention models performed worse, suggesting that Hopfield models also exhibit increased learning capabilities at longer sequence lengths.

The WikiText103 dataset is also focused on long-context understanding, but with an emphasis on recognizing important information (i.e. named entity retrieval). This benchmark is more difficult than the LAMBADA benchmark, so I allowed for more training epochs in this case 10 as opposed to 3. As you can see in the figure on the right above, RNNs still performed the worse overall, as expected. Furthermore, even though Hopfield attention does eventually perform at a similar rate to linear attention at long context lengths, for most context lengths linear attention is the clear winner. This suggests that for arbitrary word retrieval and understanding tasks, linear attention still holds the advantage over Hopfield attention.

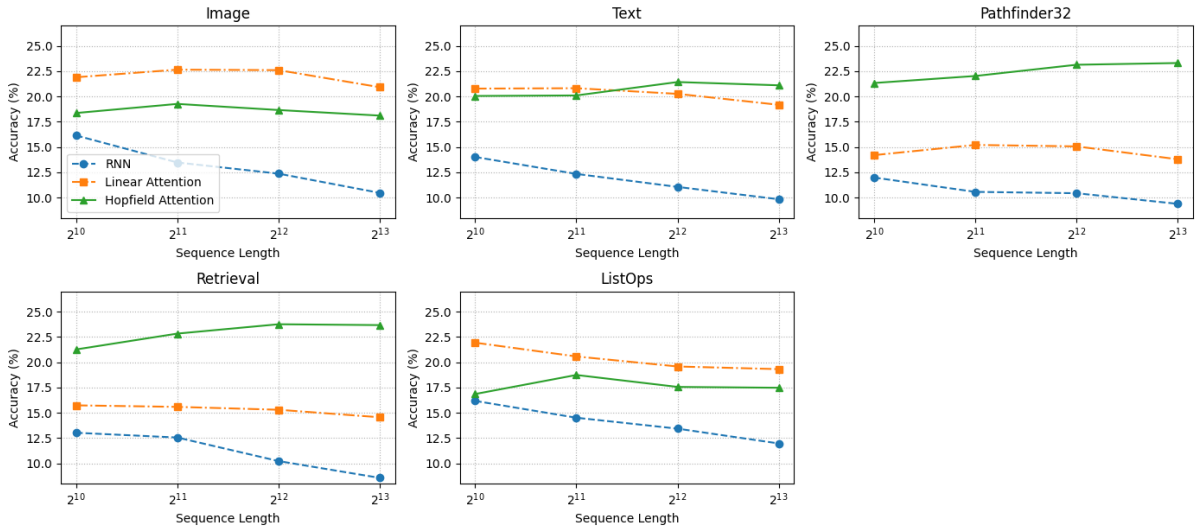


My custom Memory Copying benchmark was designed to specifically test each model's recall ability on exact sequences of tokens, with the hypothesis that at longer sequence lengths models would need more explicit memory structures to perform well. This is corroborated by the results below, which show that Hopfield networks do tend to perform better on this task, even though accuracy does tend to decrease as sequence length increases across all models. One possible explanation for why this does not occur for the LAMBADA benchmark is the fact that I used random sequences of tokens, which may be more difficult for the model to learn proper modeling relationships for as opposed to actual natural language.

For the Long Range Arena benchmark, long-context understanding is important, but is not all that is needed. Each task presents a different challenge for the model, which is represented in the results shown. For tasks that require nuanced understanding of the input, linear attention outperforms both RNNs and Hopfield attention, as you can see in the ListOps and Image task. For most other tasks, Hopfield attention outperforms the others, but this is especially apparent in the Retrieval and PathFinder32 tasks, which tended to deal with data that required longer sequence lengths and understanding. Interestingly, the performance appears to peak at a sequence length of 4096, suggesting that even at longer sequence lengths Hopfield attention does struggle for these tasks. Furthermore, both models appeared to perform similarly on the Text task, which may be related to the fact that this task didn't require any long context understanding, but did have longer sequences.

Conclusion

This project demonstrates that Hopfield attention mechanisms offer substantial advantages in long-context scenarios, especially where memory retrieval and pattern recognition over extended sequences are crucial. Although RNNs generally under perform as sequence length increases, linear attention and Hopfield attention show varying strengths. Hopfield networks surpass their counterparts in tasks like Memory Copying and certain Long Range Arena subtasks (e.g., Retrieval and Pathfinder32), particularly at longer sequence lengths. However, for tasks like semantic retrieval (e.g., WikiText103), linear attention still leads. These findings suggest that Hopfield networks introduce a valuable capability for associative recall, making them a strong candidate for future models targeting in-context learning or long-range dependencies, albeit with limitations in computational efficiency and generalization. The experiments underscore the importance of aligning architectural choices with task-specific memory demands, and provide empirical evidence of Hopfield attention's feasibility in the modern transformer landscape.



References

[Cho+22] Krzysztof Choromanski et al. *Rethinking Attention with Performers*. 2022. arXiv: 2009.14794 [cs.LG]. URL: https://arxiv.org/abs/2009.14794.

[Hop82] J J Hopfield. "Neural networks and physical systems with emergent collective computational abilities". en. In: *Proc. Natl. Acad. Sci. U. S. A.* 79.8 (Apr. 1982), pp. 2554–2558.

[Hua24] Jerry Huang. *How Well Can a Long Sequence Model Model Long Sequences? Comparing Architechtural Inductive Biases on Long-Context Abilities*. 2024. arXiv: 2407.08112 [cs.LG].

[Ram+21] Hubert Ramsauer et al. *Hopfield Networks is All You Need*. 2021. arXiv: 2008.02217 [cs.NE]. URL: https://arxiv.org/abs/2008.02217.