Are You Using Retentive Networks?

Anonymous Author(s)
Affiliation
Address
email

Abstract

The Retentive Network (RetNet) has been offered as a drop-in replacement for the Transformer. RetNet claims to have generation quality comparable to Transformer, while reducing inference time complexity from $O(n)$ to $O(1)$. We train Transformer and RetNet large language models with varying hyperparameters and conduct an in-depth benchmark analysis to evaluate their output quality. We present new hyperparameter guidelines for training Retentive Networks. We investigate the trade-offs between energy cost and performance in RetNet models.

1 Introduction

The advent of the Transformer architecture[13] in Large Language Models (LLMs), like ChatGPT[8] and LLama 2 [12], marked a significant evolution in the field of Artificial Intelligence (AI), particularly in Natural Language Processing (NLP). Transformer-based LLMs are renowned both for their parallelized training procedure and high quality output, achieving state-of-the-art on most NLP tasks today. Given their performance, it is no surprise that academia, industry, and governments are striving to develop and integrate Transformer-based LLMs. Unfortunately, these models are prohibitively compute-hungry and only growing more so [5]. This has lead to a myriad of strategies to maintain output quality while reducing model size and inference time, such as model distillation [6], model quantization [10], and even novel architectures [14, 1, 11]. This paper delves into one such proposed architecture, the Retentive Network (RetNet).

Transformers constituted a generational leap from previous model architectures. Unlike traditional Recurrent Neural Networks, Transformers use self-attention instead of hidden states, enabling a more effective grasp of context. This feature is particularly beneficial for complex language tasks like logical reasoning, summarizing, and natural language generation. Like a Transformer, RetNet trains in parallel and generates high-quality output, but can reduce inference cost from the Transformer's $O(n)$ time/memory complexity per token to $O(1)$ per token.

Through empirical analysis, we examine the strengths and limitations of RetNet and Transformer architectures, providing insights into optimal application scenarios under different parameters and scenarios. The evaluation will not only focus on technical performance and quality of output but also consider practical aspects such as ease of integration into existing systems, computational resource requirements, and scalability. We hope that this comprehensive comparison will shed light on the potential of these models to enhance and transform AI-driven language processing in various real-world applications.

This paper makes the following contributions:

- In Section 3, we evaluate the performance of Transformers based on the following criteria: contextual understanding, long-term dependency, and adaptability to different language tasks.

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

- In Section 3.5, we show that the RetNet architecture doesn't hold long-term word dependencies as well as its Transformer counterpart; the recurrent method of generation loses information from the past.
- In Section 3.7, we investigate the trade-offs between RetNet energy savings and performance. We claim that while RetNet models take less energy to train, their performance loss in comparison to a similarly scaled Transformer model makes them less viable.
- In Section 3.8, we propose a set of guidelines for RetNet initialization. When context length is doubled and embedding size is halved, RetNet achieves better results than Transformers.

2 Methods

2.1 Training Pipeline

We implemented a PyTorch training pipeline that used model architectures in Microsoft's Torchscale library[7]. From Torchscale, we used a standard decoder-only Transformer architecture as well as the official RetNet architecture [11]. We trained all models on the C4 dataset [3].

We used 240 GPU hours, across 8 Nvidia A100 GPUs, to train all of our models.

2.2 Hyperparameter Selection

In order to better understand the corresponding strengths and weaknesses of the Transformer and RetNet architectures, we performed a grid search across varying hyperparameters. The hyperparameters we explored included the Learning Rate, Embedding Dimension Size, Feed Forward Dimension, Number of Heads, and the Sequence Length. We then recorded loss, model size, time to train, and other metrics for data evaluation, which are analyzed in Section 3.

2.3 Evaluation Suite

Once each model finished training, it was passed through a bespoke evaluation suite. We aim to reduce bias towards architectures through publishing the results of all models across each task in our evaluation suite. The evaluation results are found in Section 3.

We now expound upon the evaluation suite developed to measure model performance. Because we used the architectures published in the GitHub repository referenced in the original RetNet publication[11], we had to convert both the Transformer and RetNet models into HuggingFace-compatible models to pass through the evaluation suite. We used a grid search approach in order to find desired hyperparameters. With each set of hyperparameters, we trained models with both architectures. Models ranged from 1.3 billion to 7 billion parameters. With this process, we searched for the answer to two questions: 1) How do Transformer and RetNet models compare with "similar" hyperparameters? 2) Can either model get a substantial advantage (in training time, inference time, inference space) with their respective best-found hyperparameters? Each model was passed through our evaluation suite and results were recorded.

3 Results

ONE LINE SPECIFIC SUMMARY OF WHAT RELEVANT RESULTS ARE FOUND BELOW

3.1 Efficiency

3.1.1 Time Complexity

One of the biggest claims in favor of RetNet is the reduced time complexity and, by association, energy costs. The original paper on RetNet [11] claims that this type of architecture achieves a time complexity of $O(1)$ during inference. This should manifest in constant generation speeds notwithstanding the number of tokens in the context window. Such generation speeds should also be accompanied by smaller energy costs. This contrasts with expectations for a traditional Transformer model, which has a time complexity of $O(n)$. We can see that in FigureCHANGE THIS TO A

FIGURE that these claims are largely supported. As our context window grows logarithmically, we see the Transformer model's compute time grow as well. The compute time for RetNet stays stable, however, within $\pm 5\%$ of a fixed compute time. These results demonstrate RetNet really does have the hoped-for time complexity of $O(1)$.

3.1.2 Space Complexity

It is well documented that Transformer models require a lot of space in order to find relations between the tokens in the input. This results in a space complexity of $O(n^2)$ during the Softmax operation. On the other hand, RetNet bypasses this restriction by replacing the Softmax with a Swish gate [9]. Because of this innovation, RetNet attains a theoretical space complexity of $O(1)$. As we see in Figure, there is only a very small amount of space used for any sized output. Furthermore, the memory used by the RetNet model doesn't seem to scale with the context window size. We can see that the Transformer model performs as expected with a polynomial space complexity of $O(n^2)$.

3.1.3 Energy Efficiency/Cost Reduction

As discussed in Section 3.1.4, RetNet reaches convergence faster than similar Transformer models. On average, RetNet took 80% as long to train as a comparative Transformer on the same hardware configuration. This translates to 80% as much energy consumption and a 20% reduction in training costs. With RetNet's ability to infer in $O(1)$ time complexity, RetNet takes just 10% as much energy to compute an average length output (as defined in Section 3.3) than a similar Transformer model during inference time. This equates to a cost reduction of 90% for RetNet!

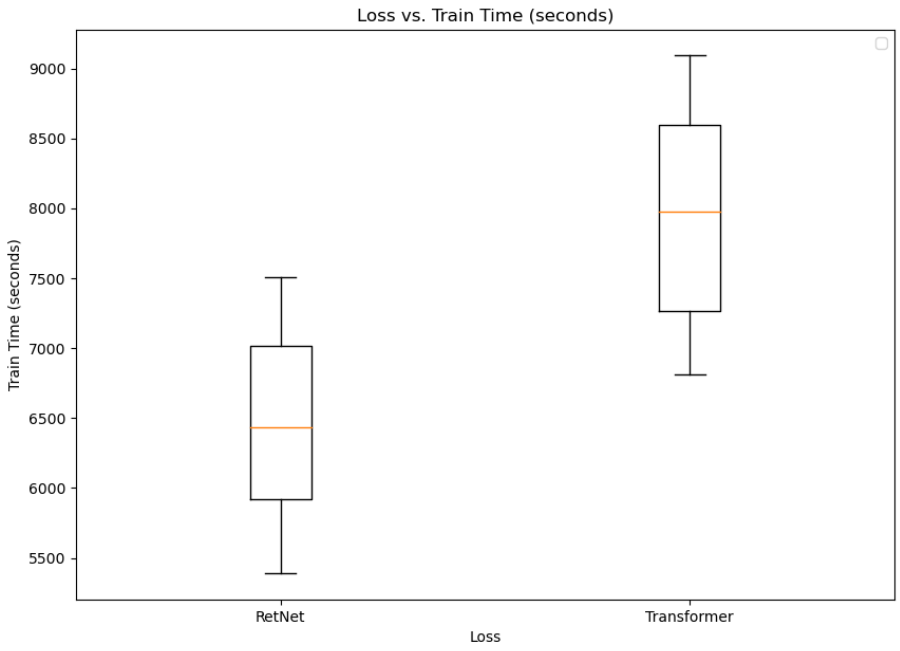

Figure 1: Training Time

3.1.4 Training Time

When training the RetNet and Transformer models, there was a significant difference in training time. We built 72 models with varying hyperparameters during grid search for each architecture. We observed that the training time for RetNet averaged 6442.23 seconds while the Transformer exhibited a notably longer average training time of 7926.7 seconds. This difference, amounting to approximately 1500 seconds, highlights a significant efficiency advantage for the RetNet architecture

in terms of computational time required for training. These results are in line with the claims by the original publishers of the RetNet paper[11].

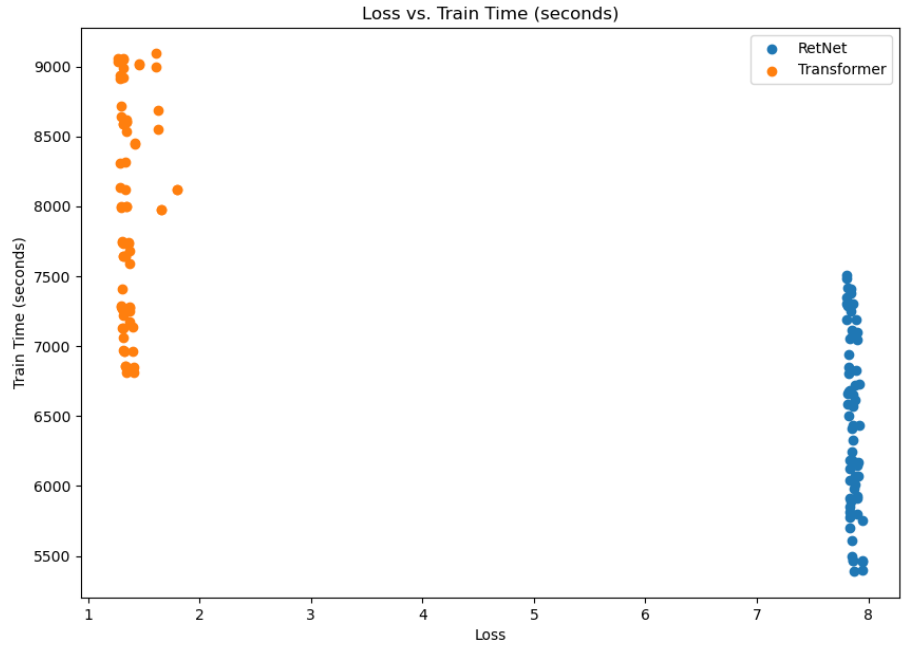

Figure 2: Loss vs Training Time

3.2 Loss

The analysis of average losses across 72 models of both RetNet and Transformer architectures, each tested with varying hyperparameters, revealed notable distinctions in performance. The Transformer models demonstrated superior efficiency in minimizing loss, with an average loss of 1.372 across different configurations. In contrast, the RetNet models exhibited a significantly higher average loss of 7.859. This disparity underscores a critical performance difference between the two architectures in terms of their capacity to reduce error or loss during training.

For context, both types of models were tested on all the combinations of the set of hyperparameters that are laid out in section 3.8

Delving deeper into the specifics of the best-performing models, the RetNet architecture achieved its lowest average loss with a set of hyperparameters including an embedding dimension (embed dim) of 1280, a feed-forward dimension (FFD) of 2048, 4 attention heads, and a sequence length of 256. The learning rate (LR) that contributed to this optimal performance was either 0.001 or 0.0005. On the other hand, the Transformer models attained their best average loss under similar yet distinct conditions. The optimal Transformer configuration shared the same embedding dimension and feed-forward dimension as the best RetNet model. However, it differed in having 8 attention heads (double that of RetNet's best model) and a longer sequence length of 512. The learning rate for the best Transformer model was set at 0.0005.

These findings suggest that while both architectures can be fine-tuned to achieve lower losses, the Transformer models have a fundamental advantage in minimizing loss when comparing models under 1.3 B parameters.

3.3 Benchmarks

We feel that the best measure of a language model's quality is its performance in the real world. Because of this, we wanted to find benchmarks that would be indicative of real-world performance. For our evaluation suite, we used the LM Eval Harness[4], from EleutherAI. This is a publicly available testing suite that includes many relevant LLM benchmarks. Despite RetNet's higher loss than transformer 3.2, we found that our benchmarks told a more nuanced story. In benchmarks that rely on accurate word prediction like (I DON'T KNOW WHAT GOES HERE), Transformer models hold a decisive lead. However, on conversational and logic based benchmarks like (I STILL DON'T KNOW), RetNet pulls ahead in more than a few cases, and where it trails Transformers, it isn't by much. These benchmarks show that, while RetNet struggles with loss, it is a very capable language model and there is certainly an argument to be made in favor of it.

3.4 Contextual understanding

3.5 Long-term dependency

3.6 Adaptability to other tasks

3.7 Performance

Calvin wants to write this section

3.8 Hyperparameter selection

"""Note: Maybe add the data ran on last Saturday for a placeholder

SO many plots, tables and figures. Remember that the captions should include all information needed to understand what the reader is looking at. The text of the Results section should tell a story about what is being shown in the figures."""

4 Broader Impacts/Ethics

The development and implementation of Retentive Networks (RetNets) promise a substantial shift in the accessibility and utility of large language models (LLMs). This section explores the broader implications, both beneficial and potentially challenging, of widespread RetNet adoption.

4.1 Increased Accessibility

The reduced computational requirements of RetNets could democratize access to advanced LLMs. This technology could become available on less powerful devices, including smartphones and personal computers, which would significantly expand the user base. By enabling more individuals and organizations to utilize LLMs, we can expect a surge in innovative applications, ranging from personalized educational tools to enhanced language translation services.

4.2 Data Privacy and Security

Running LLMs locally, as enabled by RetNets, could enhance data privacy. Users could process sensitive data on their own devices without sending it to remote servers. This local processing mitigates the risks associated with data breaches and unauthorized access common in cloud-based systems. However, this also raises concerns about the potential for misuse of these models, such as generating misleading information or deepfakes, and requires careful consideration of ethical guidelines and possibly new regulatory frameworks.

4.3 Equity and Fairness

While increased accessibility of LLMs through RetNets is a positive development, it also raises questions about digital divide and equity. There's a risk that the benefits of such technologies might be disproportionately available to those with access to the necessary hardware and technical knowledge. Ensuring equitable access and addressing disparities in technology adoption should be a priority.

4.4 Ethical Use and Governance

The ease of access to powerful LLMs necessitates a renewed focus on ethical guidelines and governance. Users of RetNets must be aware of and adhere to ethical standards, especially regarding content generation and data handling. It's essential to develop a framework for responsible use, addressing concerns like misinformation, bias in model outputs, and the potential impact on human labor.

4.5 Research and Collaboration

The feasibility of running LLMs locally could foster a new wave of research and development, as smaller institutions and independent researchers gain access to powerful models. This democratization can lead to more diverse and inclusive research, potentially driving innovation in unexplored directions.

In conclusion, while RetNets offer exciting opportunities in terms of performance and accessibility, it's imperative to navigate their broader impacts with a conscious understanding of ethical, societal, and environmental implications. As we stand on the brink of a significant shift in AI accessibility, it's crucial to address these challenges proactively, ensuring that the benefits of RetNets are realized responsibly and equitably.

If appropriate for the scope and focus of your paper, did you discuss potential negative societal impacts of your work? Please see the Paper Checklist Guidelines for detailed instructions and examples of points that you may choose to discuss. Enter yes, no, n/a, or an explanation if appropriate. Answers are visible to reviewers.

5 Limitations

While our study provides valuable insights into the performance of RetNet compared to Transformers, it is important to acknowledge certain limitations that should be considered when interpreting our findings.

5.1 Data Bias

Another limitation to consider is the potential bias in our dataset. The quality and diversity of datasets used for training and evaluation of language models can significantly impact the generalizability of our results. While we made efforts to use a representative dataset, it is possible that certain biases exist within it, which could influence the comparative performance of RetNet and Transformers. Future work can explore the effects of different datasets and data distributions on model performance to obtain a more comprehensive understanding of their capabilities and their influences on large language models.

5.2 Hyperparameter

Our study focused on a specific set of hyperparameters for both RetNet and Transformers. The choice of hyperparameters can profoundly affect model performance, and different configurations may yield different results. We acknowledge that further exploration of hyperparameter tuning could provide valuable insights into the relative strengths and weaknesses of these architectures since RetNet, especially is a much newer model with a lot less research done on the hyparameters and its scaling. Therefore, future research should consider a broader range of hyperparameters to ensure a more robust evaluation.

5.3 Scalability

One of the primary limitations of our work is scalability. Due to hardware and resource constraints, we were not able to scale our project to the extent we desired. Our experiments were conducted with a limited number of parameters and computational resources. As a result, our conclusions may not fully capture the behavior of RetNet and Transformers in scenarios with significantly larger models. Very often, the performance and behavior of large language models vary with the scale of parameters

and data. We recognize that in real-world applications, language models are frequently trained with larger datasets and more extensive architectures to achieve optimal performance. Our study did not and could not explore the full spectrum of scalability, and therefore, we cannot definitively assert whether RetNet outperforms Transformers as the model size increases further. Future research with access to greater computational resources should investigate this aspect further.

5.4 Task Specificity

The conclusions drawn in this study are based on a specific set of tasks and benchmarks. Different applications may have unique requirements and characteristics that could impact the relative performance of RetNet and Tranformers. It is important to recognize that our findings are limited to the tasks and datasets we have considered. Researchers and practitioners should exercise caution when extrapolating our results to different domains of studies or problem types in the real world.

In summary, while our study contributes to the understanding of RetNet and Transformers in the context of our specific experiments, the limitations outlined above should be taken into account when interpreting our findings. Future research should aim to address these limitations and provide a more comprehensive assessment of these architectures across various dimensions of scalability, data bias, hyperparameter sensitivity, and task diversity.

Did you describe the limitations of your work? You are encouraged to create a separate Limitations section in your paper. Please see the Paper Checklist Guidelines for detailed instructions (https://neurips.cc/public/guides/PaperChecklist). Enter yes, no, or an explanation if appropriate. Answers are visible to reviewers. You are encouraged to create a separate "Limitations" section in your paper.

6 Future Work

6.1 Transformer to Retnet conversion

One issue that arises when disrupting the architecture of an established technology, is the integration of the new architecture. This is no different in replacing transformer-based models with retnet-based models. The cost of training a model that can be used in production is quite exorbitant. This might keep some people in the field from adopting retnet. We propose several solutions that can be studied in future work.

1) Oracle learning. It is a known technique to take a large model and train a smaller model from the larger model's outputs. This makes a more deployable model that is much easier to train and is comparable in performance. This technique can almost certainly be used to train a retnet model from a transformer model. The retnet can then be fine-tuned if there are any issues. This is a well-established method that is pretty certain to work. Retnet may provide some unknown advantages/disadvantages when it comes to oracle learning and it may be worth looking into.

2) Another, more unresearched topic, would be converting an existing transformer model into a retnet algorithmicly. This approach has some innate challenges because transformers and retnets have some fundamental architectural differences. However, transformers models and retnet models exist to achieve the same goal, so it stands to reason that there may be some transformation that can be performed to convert one into the other. The two main things that go into making a model that already has an established architecture are model weights and hyperparameters. It may be feasible for a neural network to take the hyperparameters and weights of a Transformer model and convert them into the weights and hyperparameters of a similar retnet model. Specifically, using a convolutional model to find the interactions between multiple layers of the input model may be effective.

6.2 Linear Transformers and Other Models

Something that we weren't able to get to in this paper was comparing linear transformers to retnets. There is some debate within the space of large language models as to whether or not retnets are a reincarnation of linear transformers. Based on our knowledge of linear transformers, we believe that our retnets are performing better than similar linear transformers would. However, we would like to make direct comparison to confirm this belief.

7 Related Work

The introduction of transformers [13] was a generational leap for language models and has continued as the state of the art ever since. Transformers allow language models to focus their attention on important things in a conversation.

The Longformer [1]is an alternative transformer architecture that scales linearly with sequence length instead of quadraticly. Retentive networks [11] also have linear attention as well as similar performance to transformers. The Linformer [14] also claims linear inference time, rather than quadratic.

Flash Attention [2] involves low-level code optimizations to transformer architecture to improve GPU efficiency.

8 Conclusion

In our paper we run a thorough evaluation suite between the RetNet and Transformer architectures. We show potential drawbacks of the RetNet architecture. Transformers handle long term context length dependencies better than RetNet. We also propose new initialization guidelines for RetNet as opposed to their transformer counterparts. In the future we would like to continue to scale RetNet models in order to compare their performance against equal sized Transformers, and use a larger and more comprehensive dataset to ensure adequate data access for each model. We would also like to run a thorough comparison against linear transformers regarding the same claims of reduction in inference and memory time complexity.

References

[1] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[2] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16344–16359. Curran Associates, Inc., 2022.

[3] Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. Documenting large webtext corpora: A case study on the colossal clean crawled corpus. *arXiv preprint arXiv:2104.08758*, 2021.

[4] Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 12 2023.

[5] Eva García-Martín, Crefeda Faviola Rodrigues, Graham Riley, and Håkan Grahn. Estimation of energy consumption in machine learning. *Journal of Parallel and Distributed Computing*, 134:75–88, 2019.

[6] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[7] Shuming Ma, Hongyu Wang, Shaohan Huang, Wenhui Wang, Zewen Chi, Li Dong, Alon Benhaim, Barun Patra, Vishrav Chaudhary, Xia Song, and Furu Wei. TorchScale: Transformers at scale. *CoRR*, abs/2211.13184, 2022.

[8] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[9] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[10] Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

[11] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2023.

[12] 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.

[13] 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.

[14] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. In *International Conference on Machine Learning*, pages 9244–9253. PMLR, 2020.