RetFormers: Hybrid Attention-Retention Networks for Faster Inference

Ishank Agrawal and Alex Hu

Motivation

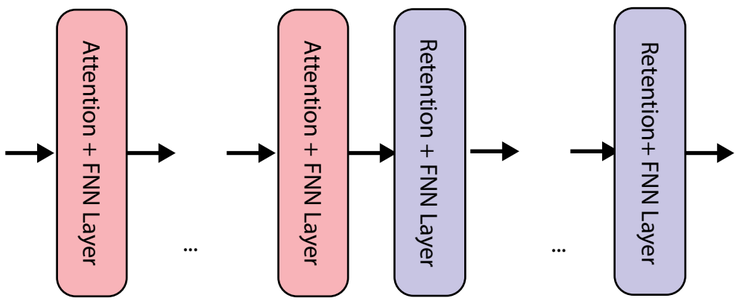
Transformers are the one of the most successful architectures for language modeling and related tasks. However the these models have a quadraic inference time complexity in their sequence lengths, which severely limits content length. Variaous linear time architectures have been proposed such as RetNets, RWKV, S4, etc. However these models are not sufficiently expressive enough to enjoy the performance that transformers do.

Here we examine hybrid retention-attention networks to decrease the constant factor in inference time. We suitably replace attention heads with retention heads and show that perplexity can be maintained with a decrease in model size and evaluation time.

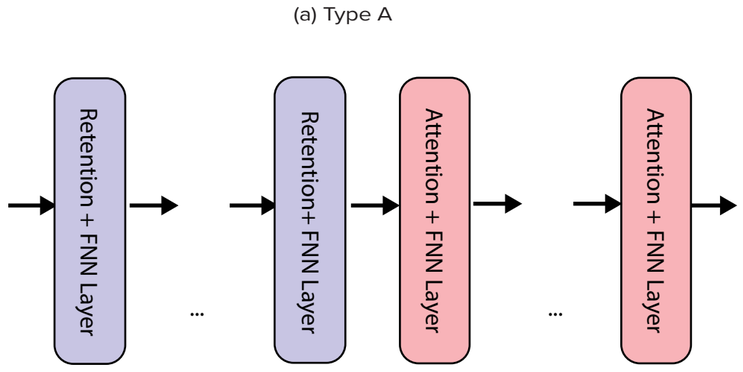
Architecture

We use three types of layers as fundamental building blocks for our architectures. The first two are simply attention and retention layers. The third is a hybrid retformer layer. The architecture of all three are given in Figure 1

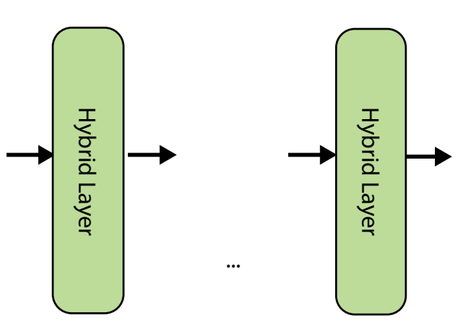
We combine these three types of layers in different ways to form three types of RetFormer architectures. These are given in Figure 2.



(a) Type A



(b) Type B



(c) Type C

Figure 2: Type A, B, and C RetFormers

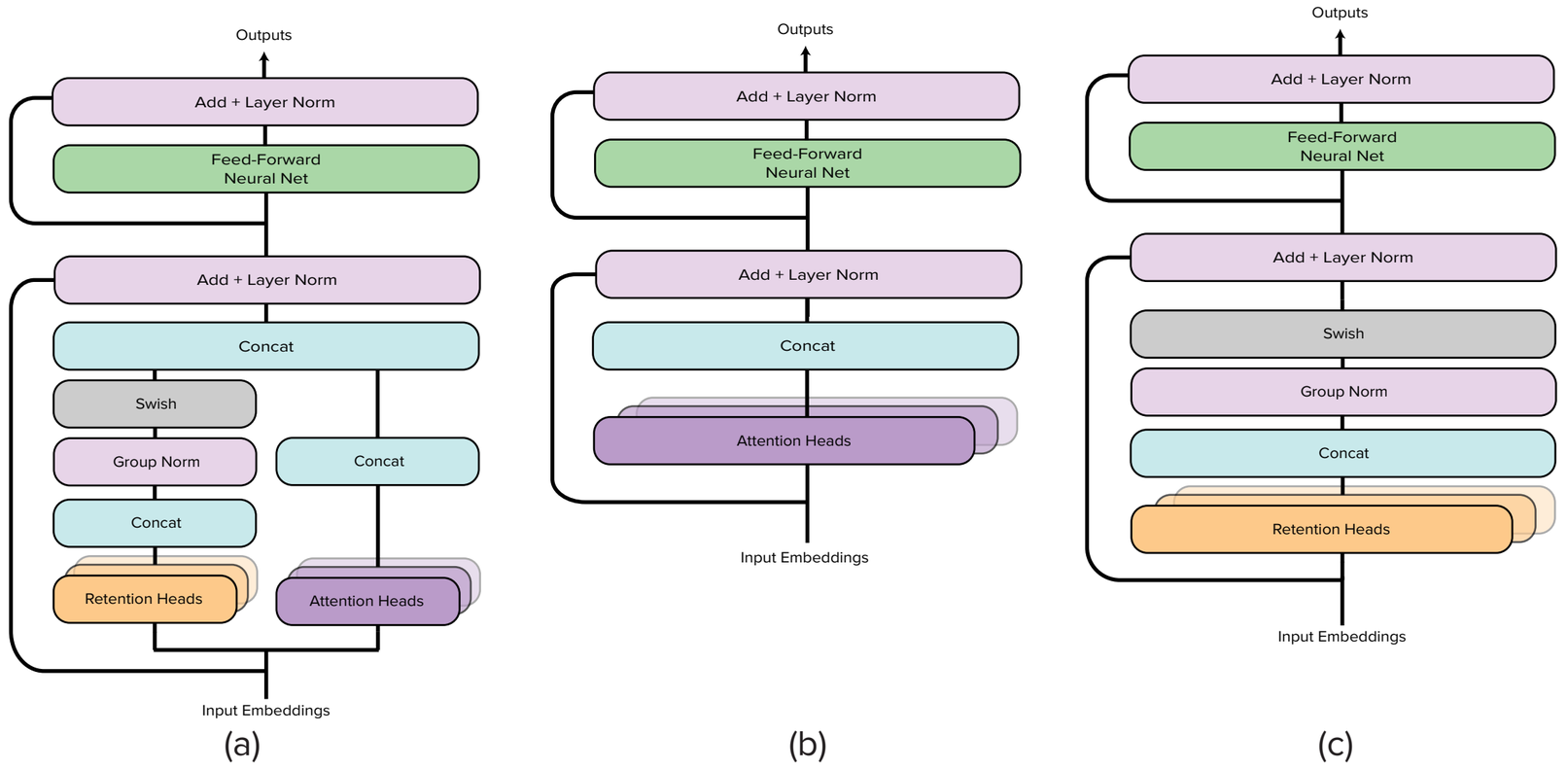


Figure 1: (a) Hybrid retformer layer (b) Multi-headed attention layer; (c) Multi-scale retention layer

For each RetFormer architecture, we define an AR ratio as the ratio of the number of attention heads to the number of retention heads.

$$\chi = \frac{\#\text{ OF ATTENTION HEADS}}{\#\text{ OF RETENTION HEADS}}$$

The larger the AR ratio, the more computational time the network takes. Thus our central goal was to minimize AR ratio while maintaining good performance.

Experimental Results

We trained all three types of retformers on a downsampled subset of the WikiText-2 dataset.

The AR-ratio, number of GigaMACS and train perplexity are given in table 3. We omit the validation perplexity as we use a downsampled subset and we were mainly interested in the expressive power of the network.

From these results, it is clear that for the same AR-ratio, Type A RetFormers perform better than type B RetFormers. Type C RetFormers performed the worst.

In fact, a Type A RetFormer with AR ratio 10:2 performed better than a vanilla transformer.

Architecture Type	AR ratio	# of GigaMACs	Train perpl.
Vanilla Transformer	1:0	10.305	4.251
Vanilla RetNet	0:1	6.709	23.984
Type-A Retfomer	1:11	7.009	13.426
Type-A Retfomer	2:10	7.308	34.811
Type-A Retfomer	3:9	7.608	6.942
Type-A Retfomer	4:8	7.908	5.162
Type-A Retfomer	5:7	8.207	5.885
Type-A Retfomer	6:6	8.507	4.845
Type-A Retfomer	7:5	8.806	4.364
Type-A Retfomer	8:4	9.10	4.291
Type-A Retfomer	9:3	9.406	3.955
Type-A Retfomer	10:2	9.705	3.209
Type-A Retfomer	11:1	10.005	4.762
Type-B Retfomer	1:11	7.004	12.725
Type-B Retfomer	2:10	7.304	9.483
Type-B Retfomer	3:9	7.604	7.508
Type-B Retfomer	4:8	7.904	11.826
Type-B Retfomer	5:7	8.204	10.549
Type-B Retfomer	6:6	8.504	8.345
Type-B Retfomer	7:5	8.804	4.81
Type-B Retfomer	8:4	9.104	4.74
Type-B Retfomer	9:3	9.405	5.31
Type-B Retfomer	10:2	9.705	5.226
Type-B Retfomer	11:1	10.005	4.807
Type-C Retfomer	1:7	8.563	40.775
Type-C Retfomer	2:6	8.732	13002.334
Type-C Retfomer	3:5	9.012	119.673
Type-C Retfomer	4:4	9.406	206.809
Type-C Retfomer	5:3	9.911	72.801
Type-C Retfomer	6:2	10.529	52.354
Type-C Retfomer	7:1	11.260	56.122

Table 3: Training perplexities of various architectures

Future Work

This preliminary analysis shows that there is much potential in using hybrid retention and attention networks. We were limited by computational power, and thus this work mainly analyzes expressive power. It would be interesting to explore how such models generalize to unseen testing datasets. It could also be worth exploring how performance of these RetFormers scale with even larger language models.

Acknowledgements

We are extremely grateful for Prof. Yoon Kim's guidance and help during this project as well as the entire NLP staff.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need.

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

Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. 2023. Roformer: Enhanced transformer with rotary position embedding.