CS 397 Final Report: An Exploration into Parametric Modifications in the Attention Mechanism of the Decoder Model

Group 8

Abstract

In this project we work with changing six different parameters related to attention heads in the decoder of the transformer architecture and examine their effects on the performance of the model. Inspired by the paper "Attention Is All You Need", we aim to show how different configurations of attention heads - an architecture central to attention mechanism of the model - can help increase performance of the model. We find that three of the parameters improve upon the base model perplexity at the cost of training time while others do not for various reasons.

1 Introduction

In the famous paper "Attention is All You Need", a novel architecture called the transformer has been introduced. The transformer is an attention-based alternative to recurrent neural networks that has achieved state-of-the-art across a number of NLP tasks. Faithful to the name of the paper, it has been proven that an architecture with only attention-mechanism outperforms previously state-of-the-art recurrent neural networks.

A vanilla transformer transforms a sequence into another with an ensemble of an encoder and decoder. Throughout the years many variations of the transformer have been used for various purposes: decoder-only GPT-2, encoder-only BERT, etc. Yet, any transformer model utilizes the attention mechanism and thus it is of most importance to better understand this part of the model.

We examine this part of the decoder-only transformer in more detail. In particular, we work with varying the attention dropout ratio, embedding and hidden layer size, number of attention heads, and number of hidden layers. We also observe the impact of different inner feed forward layer size.

2 Model

Inspired by GPT-2, the model used is the base transformer model only consisting of a decoder. To test different configurations, we changed the selected configurations a number of times to values lower and higher than the base and compared the results. In light of the goal of this work, we mostly focused on self-attention mechanism of the decoder.

In the model, dropout is used for regularization on the output of each sub-layer before it is added to the sub-layer input and normalized. We inspected the results of changing the attention dropout rate from base value of 0.1 to values [0.02, 0.07, 0.20, 0.30, 0.50].

In regards to self-attention mechanism, we also examined different dimensionalities of embedding and hidden layers by considering values of [300, 804, 1002, 1302] instead of the base 600. We went further by changing the number of hidden layers to [2, 4, 8, 10, 12] and comparing it to the model with base value of 6.

We also changed the number of attention heads themselves to values [2, 4, 8, 10, 12] instead of the base 6.

The softmax function is central to computing the attention function. We experimented by introducing the temperature value to the softmax function which was taken to be [0.2, 0.6, 5, 20, 50] instead of the base neutral value of 1.

Apart from modifying the self-attention mechanism, we changed the inner dimensionalities of the feed-forward neural network in the decoder. In contrast to the base dimensionality of 2400, the dimensionalities considered for the feed-forward net were [500, 1000, 1800, 2800, 3500].

3 Experiments and Data

The experiments were quite straightforward in nature, since the core concept of our project was simple. Our overall goal was to measure the effect that changing parameters of attention heads and feed-forward network would have on performance. First, we trained the base transformer model to build a language model on the WikiText-2 data set in order to get a benchmark set of data points, focusing mostly on metrics such as training run time, evaluation run time, and of course perplexity. We then trained the model repeatedly, using different sets of changes for six different parameters within attention heads: dropout ratio, embedding and hidden layer dimensionality, inner feed-forward layer dimensionality, number of attention heads, number of hidden layers, and finally softmax function temperature. For each of the six parameters tested, the value of the parameter in the base model was somewhere in the middle of the range of values explored during our tests.

We used the WikiText-2 dataset instead of the WMT-2014 English to German dataset used in the original ”Attention Is All You Need” paper that this project was inspired by. This was for time-saving purposes, as we had neither the time nor the computational hardware needed to carry out that task quickly, and also needed to train 30 models. For reference, the model in that paper was trained on 8 GPUs in 12 hours - our base model was trained in 1.5 hours and, save for an exception with temperature that will be discussed later, all variations were easily within an order of magnitude of that value.

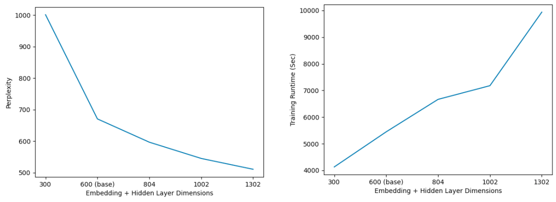
4 Results and Analysis

All models were trained for 870 steps, with 2318 samples being used for training and 240 for evaluation. Then, for the relevant values shown in the tables (eval samples, eval steps, train samples, train steps), the values shown are the processing rates **per second** of the aforementioned parameters.

4.1 Attention Dropout Ratio

Table 1: Attention Dropout Ratio

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 0.10	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
0.02	6.51	41.34	5.81	0.73	4605	1.51	0.19	668.91
0.07	6.51	36.77	6.53	0.82	4891	1.42	0.18	669.91
0.20	6.51	37.54	6.39	0.80	4634	1.50	0.19	673.18
0.30	6.52	42.09	5.70	0.71	4656	1.49	0.19	676.17
0.50	6.53	44.84	5.35	0.67	4631	1.50	0.19	684.63



(a) (b)

Figure 1: (a) Perplexity and (b) Training Time for the embedding and hidden layer dimensionality experiment. All parameter modifications that improve perplexity follow this trend.

As expected, changing the attention dropout ratio doesn't seem to have too much effect on run times, as the actual size of the model does not change with this different configuration. While perplexity does not change too much, it does trend towards getting slightly worse as dropout increases to higher than normal levels for a language model.

Since dropout is an intended to prevent overfitting to any given model, it makes sense that we wouldn't see too much of an effect within runs on the same dataset. To see the results of dropout, we would need to train on a separate corpus.

4.2 Embedding and Hidden Layer Dimensionality

Table 2: Embedding + Hidden Layer Dimensionality

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 600	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
300	6.91	29.93	8.02	1.00	4131	1.68	0.21	1001.49
804	6.39	52.24	4.59	0.57	6666	1.04	0.13	596.65
1002	6.30	56.55	4.24	0.53	7180	0.97	0.12	545.03
1302	6.24	83.33	2.88	0.36	9941	0.70	0.09	510.60

In contrast to the previous section, changing the dimensionality of embedding and hidden layers actually has a noticeable effect on our parameters of interest. As we can see in the figure showing perplexity and training time, perplexity continues to increase as dimensionality does. However, it's worth noting that this effect does seem that the effect seems to level out a bit as dimensionality continues to increase. At the same time, both training time and running time continue to increase, even increasing in rate.

4.3 Inner Feed Forward Layer Dimensionality

Similarly to changing the dimensionality of the embedding and hidden layers, increasing the dimensionality of the inner feed forward layer also seems to lower perplexity at the cost of some increase in training and evaluation time. However,

Table 3: Inner FF Layer Dimensionality

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 2400	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
500	6.63	31.53	7.61	0.95	4166	1.67	0.21	755.57
1000	6.56	39.10	6.14	0.77	4544	1.53	0.19	709.18
1800	6.53	39.68	6.05	0.76	4491	1.55	0.19	686.61
2800	6.50	47.01	5.11	0.64	4661	1.49	0.19	664.73
3500	6.49	42.50	5.65	0.71	5173	1.34	0.17	656.84

the effect is noticeably more tempered this time. The increase from 500 to 3500 for this parameter (a factor of 7) only increased training time by about 25 percent, while a smaller increase in the previous section more than doubled training time. Admittedly, the decrease in perplexity was also not as steep.

4.4 Number of Attention Heads

Table 4: Number of Attention Heads

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 6	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
2	6.51	38.55	6.23	0.78	4481	1.55	0.19	673.76
4	6.51	34.57	6.94	0.87	4662	1.49	0.19	671.68
8	6.51	40.14	5.98	0.75	5360	1.30	0.16	670.42
10	6.51	44.79	5.36	0.67	5655	1.23	0.15	670.58
12	6.51	45.32	5.30	0.66	7185	0.97	0.12	669.33

This result with number of attention heads was surprising in that it seemed to do nothing at all, lowering perplexity by a minimal amount while increasing training time significantly. Since the input is split into pieces before being fed into the attention heads, we definitely thought this would have a stronger effect since the number of attention heads directly affects the size of the split. While we're not sure of the exactly explanation for this, one possibility is that it is task and dataset dependent - WikiText-2 is composed of individual tokens taken from Wikipedia articles. It's possible that changing the number of attention heads would perform differently on a task where sentences must be taken as a whole, or on a translation task.

4.5 Number of Hidden Layers

Table 5: Number of Hidden Layers

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 6	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
2	6.58	25.20	9.53	1.19	2904	2.40	0.30	721.78
4	6.53	32.68	7.34	0.92	3788	1.84	0.23	685.69
8	6.50	49.66	4.83	0.60	6225	1.12	0.14	662.20
10	6.49	48.09	4.99	0.62	6815	1.02	0.13	655.70
12	6.48	56.91	4.22	0.53	8224	0.85	0.11	652.30

The number of hidden layers experiment showed a similar trend to them embedding and hidden layer dimensionality and inner feed forward layer dimensionality experiments. As perplexity decreases, training and evaluation time both increase. This makes sense, as the model physically has more space to fine tune itself (perhaps at the risk of overfitting). However, this experiment was distinct from the others with a similar trend in that the decrease in perplexity was very mild, especially in the face of the sharp increase in training time. That cost probably makes this modification a lower priority for further exploration.

4.6 Softmax Function Temperature

Table 6: Softmax Function Temperature

	Loss	Eval Runtime	Eval Samples	Eval Steps	Train Runtime	Train Samples	Train Steps	PPL
(base) 1	6.51	41.95	5.72	0.72	5447	1.28	0.16	670.57
0.2	6.55	416.62	0.58	0.07	37637	0.19	0.02	697.88
0.6	6.51	290.88	0.83	0.10	42574	0.16	0.02	671.09
5	6.51	35.25	6.81	0.85	43804	0.16	0.02	672.68
20	6.51	214.50	1.12	0.14	42851	0.16	0.02	673.67
50	6.51	58.86	4.08	0.51	43391	0.16	0.02	673.86

Taken at face value within itself, the change in softmax doesn't seem to have too much effect on the success of language modeling. While changing the softmax function has a visibly marked effect on the function itself, it is not too different in a practical sense once the value reaches a certain threshold, so this result is expected.

However, if we compare the training times to the previous sections, it becomes clear there is a problem. It's less likely that this is a problem with the model itself and more likely to do with the hardware we trained on, which will be explored further in the next section.

5 Discussion and Future Work

One prominent issue that is worth addressing is the inconsistency in training times, which is more likely than not due to the fact that we trained our model on Moore, which is a shared resource with the rest of the school. The first of two main points of evidence for this is that the base model, which was trained separately from the rest, is unique in that its training time doesn't fit in line with the other parameters well in most cases. For example, in the fourth experiment changing the number of attention heads, training time increases in a mostly linear fashion. Following this trend, the training time for the model should have been around 5000 seconds, but it was significantly higher at 5447.

The massive training times for the softmax function temperature and erratic evaluation times also support this theory - that test was run with the others, and while there is no way to concretely prove it, it's possible that someone else was also running an intensive task that increased the load on Moore. If we want training time to be a more reliable metric, it would have been useful to train on more consistent, dedicated hardware.

Aside from the strange behavior with training times, however, we did find some promising results among our experiments. The experiments tended to fall into one of two trends - either the training and evaluation time and perplexity all didn't change, or the perplexity was decreased at the cost of training and evaluation time. Only one experiment did not follow this trend, with perplexity staying the same at the cost of more computing time. Ultimately, increasing dimensionality on embedding and hidden layers, as well as on inner feed forward layers, and increasing the number of hidden layers themselves, all yielded promising improvements to perplexity achieved.

Useful future work on this project would be centered around the efficacy of the aforementioned promising experiments. A good immediate next step would be to repeat this experiment using a different dataset or given a different task, such as translation or sentence completion. While it makes sense that increasing the dimensionality of hidden layers increases performance for this one task, the obvious risk at hand is that of overfitting. Testing on different datasets or even different tasks would help to address this potential risk. Another possibility is that some of the parameters we found to not be useful to change here might see better results in another setting. Finally, another unexplored avenue could be to see how different parameters interact with each other when changed in tandem. For example, a linear combination of the results from increasing dropout while also increasing dimensionality suggests that perplexity could be improved while also addressing overfitting, but this is definitely a hypothesis that would need further testing.

6 Conclusion

Through this project, we examined many different facets of attention heads alongside feed-forward neural network within transformers, the current state of the art for language modeling and many other language tasks. Our experiments involved progressively changing six different parameters within the decoder-only transformer and measuring the resultant improvement (or lack thereof) on perplexity, our success measure for the task of language modeling. While some of the parameters we tested for did not yield significant improvements, we found promising decreases in perplexity (albeit at the cost of more training time), in several parameters. Lastly, we outlined potential future steps that would need to happen to verify the efficacy of these changes.