Exploration of Position Embedding Methods and Attention Mechanism on Small Transformer Classifiers

Sungwon Kim
Johns Hopkins University
3900N Charles Street APT 1105, Baltimore, MD
skim434@jhu.edu

Abstract

Transformers are currently the state-of-the-art architecture in natural language processing. They have been widely used in multiple boundaries and improved performance in various NLP tasks. However, the exact factors contributing to their effectiveness is not completely known. Since the publish of paper *Attention is All You Need*, numerous researchers have made adjustments to the attention mechanism, aiming to enhance its performance.

Nonetheless, these researches often use large models such as GPT3 and T5 as a baseline comparison.The typical training configuration for these models involves thousands, if not tens of thousands of compute hours on A100s. Even though the compute power for training is less than commercial LLMs like ChatGPT or Bard, such computing power is beyond reach for most individuals.

In our research, we examine the impact of various modifications in the attention layer. Specifically, our focus lies on the implications of different position embedding methodologies and attention calculation techniques on smaller transformers. We hope that this comprehensive review of potential modifications can serve as a valuable resource for those looking to train smaller models from scratch.

1 Introduction / Methods

Transformer models were originally introduced by [Vaswani et al., 2017] for machine translation task and have shown to be effective on variety of tasks dealing with natural language processing [Devlin et al., 2018], audio [Verma and Berger, 2021], computer vision [Dosovitskiy et al., 2020] and more. Despite the widespread use, the exact factors contributing to their effectiveness is not completely known. Furthermore, the research of these factors often test their hypothesis on large models such as GPT3 and T5, using compute resource on scale beyond reach for most individuals.

We aim to test the effect of modifications in attention layer within smaller models that are more accessible for individuals. To achieve this, we have decided to develop a transformer heavily resembling BERT. The baseline model can be found here. We took inspiration from HuggingFace's implementation of BERT where each layers are set as a class. This modularity allows for easy modification. Furthermore, we used "bert-base-uncased" tokenizer from HuggingFace for data tokenization.

We used IMDB dataset for smaller models and used BoolQ dataset for larger models. Our modifications can be divided into two categories: different positional information incorporation and attention computation modifications. Both modification implementation detail can be seen in the github repo here.

With the datasets and adjusted models, we trained the model for 10+ epochs and compared their training, validation accuracies and losses. Through this methodology, we hope to shed light on the effects of attention modifications.

We would like to note that our implementation of transformer does not strictly follow the HuggingFace's implementation of BERT model. We observe two significant differences. While the BERT model masks the padded tokens so that it does not contribute to attention, our model does not mask them. Second, HuggingFace's implementation of BERT contains a dropout layer in self-attention. Our model doesn't include that dropout layer.

2 Position Incorporating Methods

Multiple position incorporating methods have been proposed since the attention layer proposition. The original attention suggested using sinusoidal positional embedding. This embedding scheme adds combination of sin and cosine value for each position.

An alternative proposal, [Shaw et al., 2018] incorporates use of relative position embedding in attention layer. This paper incorporates the use of trainable relative position weights and shows that it achieves better results compared to the original absolute position embedding.

Recently, [Press et al., 2021] proposed position incorporating scheme named AliBi. In this method, biases are set that grow inversely proportional to the relative distance and are added to the attention values prior to softmax. The paper shows that AliBi achieves better perplexity and speed compared to other position incorporating methods, and performs unrivaled on inference on text which are longer than training texts.

Furthermore, [Haviv et al., 2022] shows that transformer models without positional encodings still learn positional information. In particular, the paper shows that the model without any positional encodings act similarly to that of models which uses absolute learned position embeddings.

While there are numerous other position incorporating methods proposed and tested in the literature, we have chosen to concentrate our investigation on these four methods.

2.1 Position Incorporating Implementation

For Sinusodial , we add the following values into our embeddings.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

Addition of radial embeddings happen only once in the initial embedding layer of the transformer. Each even indexed token adds a positional embedding its word embedding calculated with respect to $\sin$ function and position. Each odd indexed token adds a positional embedding calculated with respect to $\cos$ function and position.

For Relative position incorporation, we decided to include a trainable relative position in all of the self attention layer. The relative position parameters are independent in all layers. We multiplied the relative position parameter with query vector and added it to QK^T prior to softmax. Each attention values before softmax is transformed as follows.

$$q_i k_j^T \to q_i k_j^T + q_i p_{i-j}$$

AliBi adds a constant linear biases before softmax in each layer. The attention is now calculated using

$$\textbf{softmax}(q_i K^T + m \cdot [-(i-1), \cdots, -2, -1, 0])$$

Where m is a constant value different for each head.

2.2 Result on small model

the small model uses three attention layers followed by pooling layer. The model used IMDB dataset with max length of 512 tokens. It was trained on 2000 text / label data and tested on 1000 text / label data. The configuration can be seen in Github page.

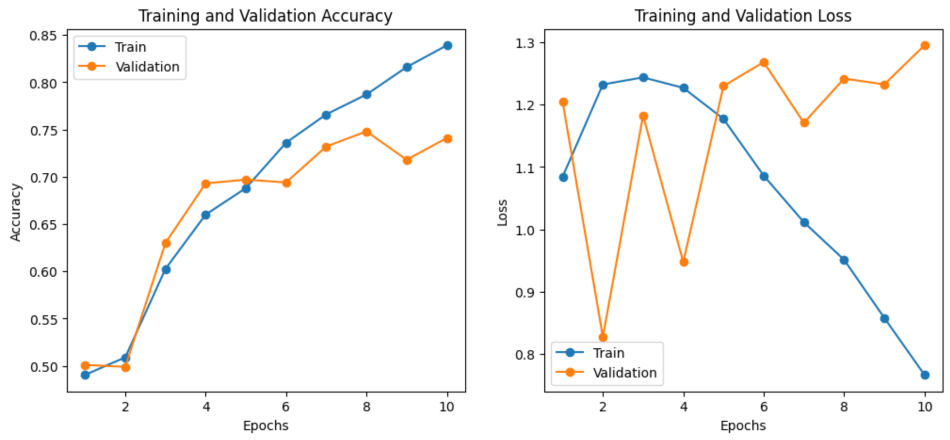


Figure 1: No Position Embeddings

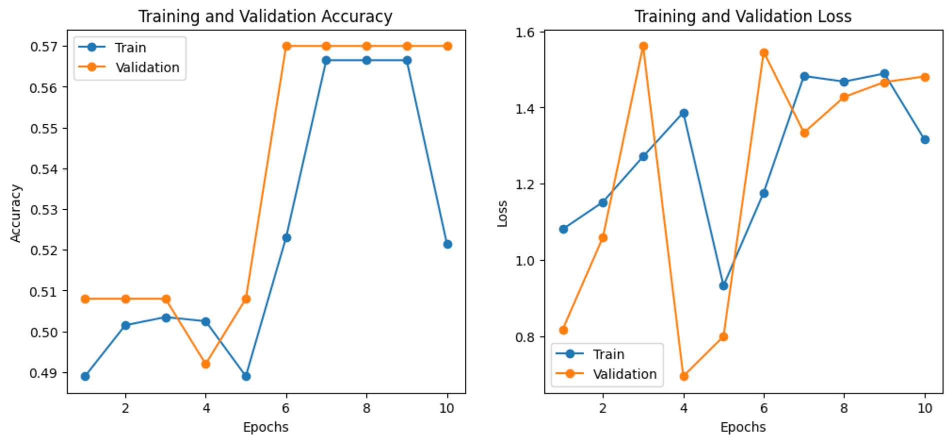


Figure 2: Sinusodial

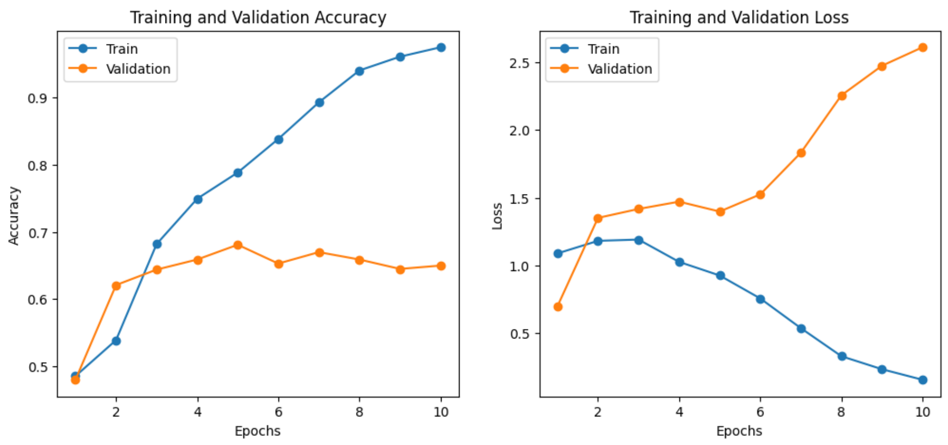


Figure 3: Alibi

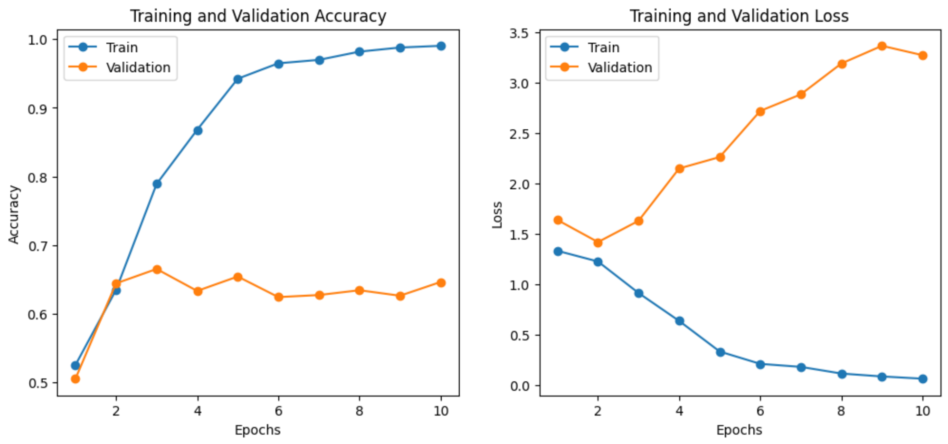


Figure 4: Relative Embedding

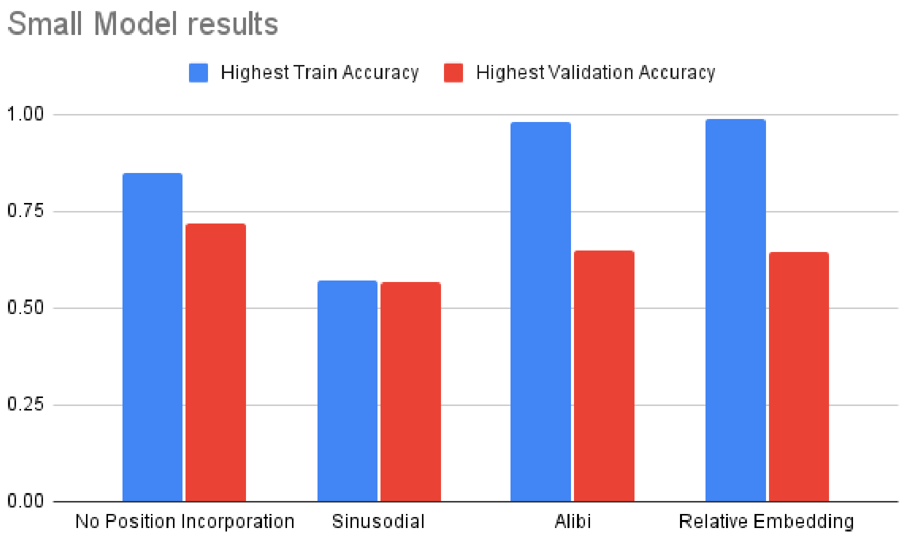


Figure 5: Summary of Small Model Results

We see from the figures above that position incorporating methods that use relative embeddings tend to overfit quickly compared to those which do not. It's also noticeable that sinusoidal embeddings do not exhibit any signs of learning. We suspect that the model may not be sufficiently deep to interpret the sinusoidal embeddings.

We also observe that Alibi and Position Embedding method show similar performance in accuracy and loss. However, relative embeddings required up to 50 percent more training time compared to other models as it had to train an additional embedding for each layer.

Finally, we notice that the highest validation accuracy was achieved with no positional embedding. Both AliBi and Relative Position methods also yielded significant results.

2.3 Result on BERT-size models

Now we test different configuration on BERT Size Model. For training and testing data, we used BoolQ dataset with max length of 512. The dataset consisted of 8000 samples for training and 2000 samples for testing.

2.3.1 HuggingFace Implementation Results

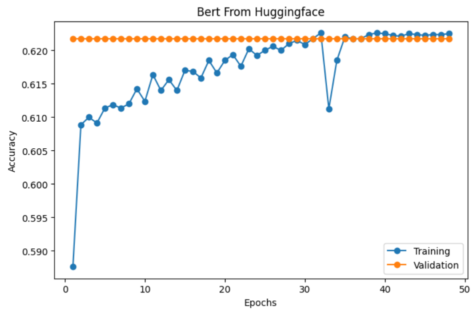


Figure 6: BERT From HuggingFace

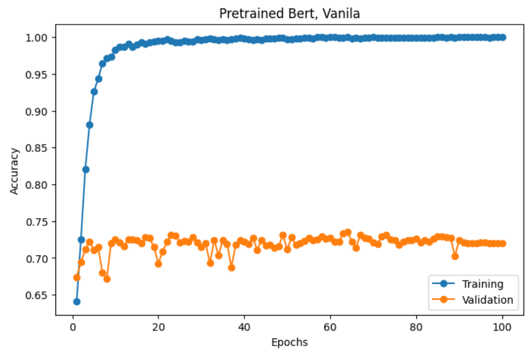


Figure 7: Pretrained BERT From HuggingFace

The default BERT implementation uses absolute trainable positional embedding in the embedding layer. Without pretraining, the validation accuracy is constant at 0.628 while training accuracy increases up to 0.63. On the other hand, pretrained BERT preforms significantly better, with accuracy of training reaching 0.99 and validation accuracy being well over 0.70.

2.3.2 Custom Implementation Results

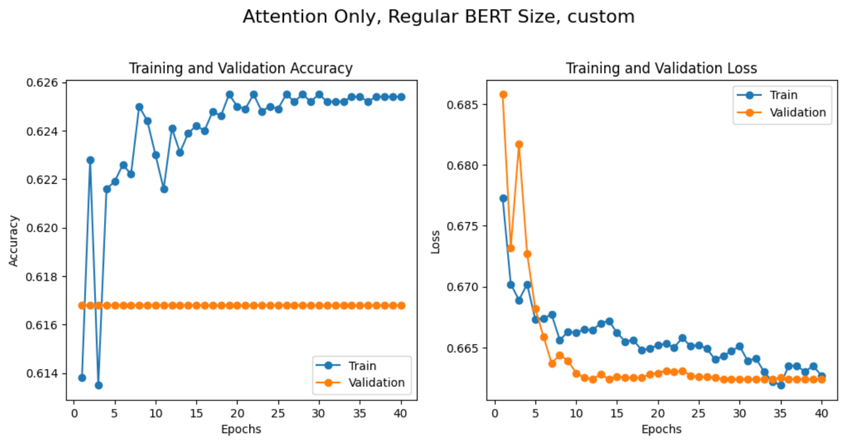


Figure 8: Attention Only, BERT-size

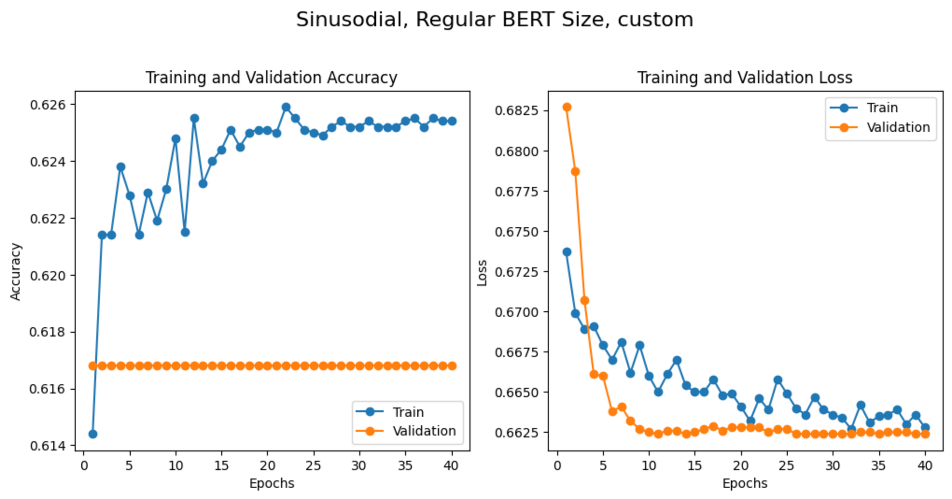


Figure 9: Sinusodial, BERT-size

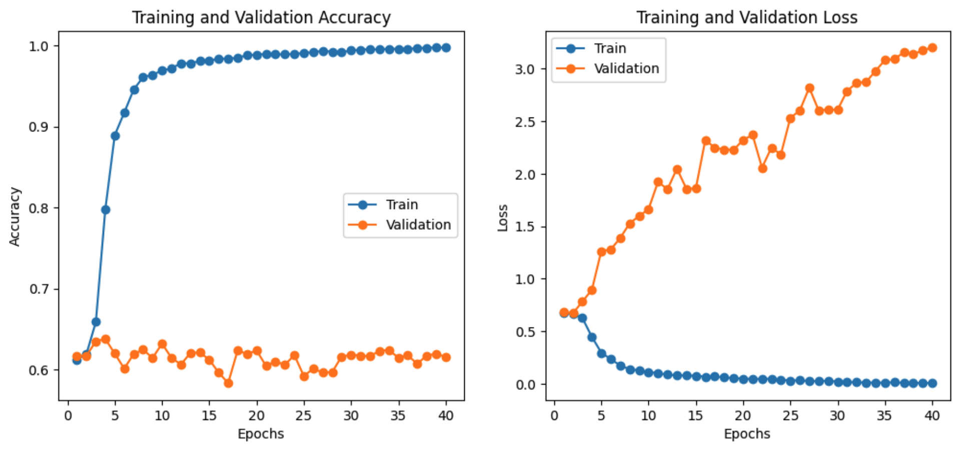


Figure 10: Relative Position Embedding, BERT-size

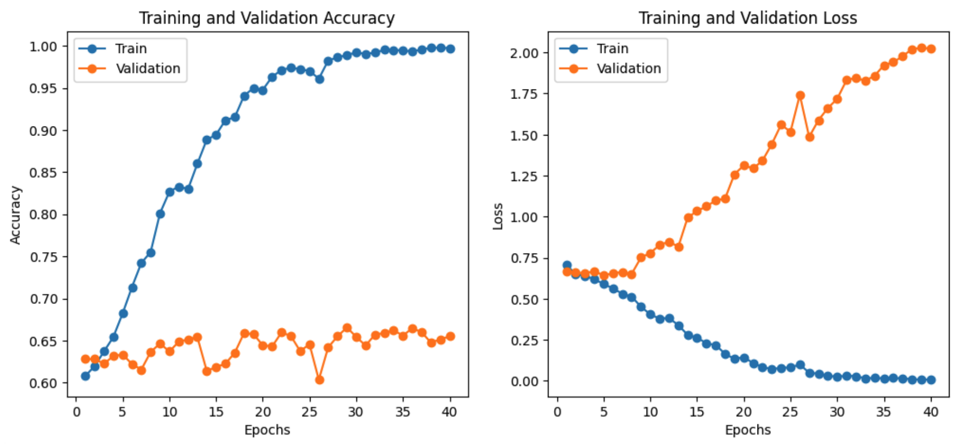


Figure 11: Alibi, BERT-size

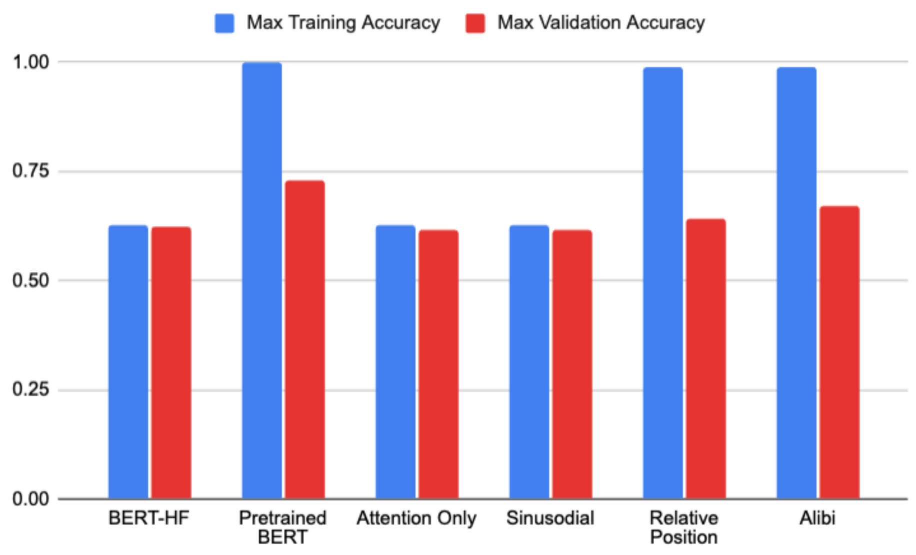


Figure 12: Accuracy Results

2.4 Discussion

We observe similar outcomes in untrained BERT from HuggingFace, Attention only, and Sinusodial positional embedding. Our results align with the finding of [Haviv et al., 2022] which suggested that attention with no positional embedding behaves similarly to attention with absolute position embedding. The training accuracy appears to plateau before 0.63 and validation stays constant. Furthermore, the validation outcome seems to be hard fixed to the majority label.

This finding puzzles us. Normally, if a model is overfitting, the validation accuracy maintains a steady state while the training accuracy escalates to 1. However, in these experiments, both training and validation accuracies remain low. We tried changing the learning rate from 5e-5 to 5e-3 and got same results. The reason behind constant validation and low training accuracy is a mystery to us. We would like to note that similar behaviors were observed when training on larger pretrained models such as Bloomz on HW6. We highly suspect that this is a consequence of small training size. Original BERT was trained on 3.3 billion words while our model was trained on less than 1.6 million words: we trained on less than 0.05 % of the original corpus.

We see that AliBi and relative position embedding methods both overfit quickly. The training accuracy reaches 1.00 while validation accuracy stays around 0.63. We also observe that AliBi gives better validation accuracy than learnable relative position and also trains faster. Therefore, we recommend using AliBi when training a model with a limited dataset.

The highest validation accuracy was achieved using the pretrained BERT model. Hence, if the aim is to obtain high metrics, we advise training on pretrained weights.

3 Modification in Self-attention

The attention mechanism is considered the heart of transformer models. This mechanism allows the model to assign different levels of importance of different elements in input sequence. However, the model has some shortcomings. In response, [Katharopoulos et al., 2020] proposed a "linear transformer" which reduces the complexity of attention layer from $O(n^2)$ to $O(n)$. Another modification is the use of global attention, first suggested by [Bahdanau et al., 2014]. Models like Compressive transformer uses the idea of global attention and extend self-attention mechanism to dependencies from previous attention.

We conducted experiments on two innovative self-attention modifications and compared their results with those of BERT from HuggingFace.

3.1 Denser Attention

While examining the attention mechanism, we found that attention might not be dense enough. Let our single query and key entries be $q \in \mathbb{R}^{1,d}$, $k \in \mathbb{R}^{1,d}$ respectively. We noticed that q_i and k_j where $i \neq j$ only interacts with each other using addition and normalization during the forward pass. Since addition is seperated out during backpropogation, the only point where q_i, k_j are interconnected in backpropogation is during normalization. We considered the relationship between two items sparse and hypothesized that attention might learn more effectively if q_i, k_j were more intertwined.

To improve the relationship between q_i, k_j, we divide each q and k into w segments and calculate the outer product of the divided segments.

3.1.1 Calculation of Denser Attention

Normally, attention is calculated using

$$\text{Attention}(K, Q, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{1}$$

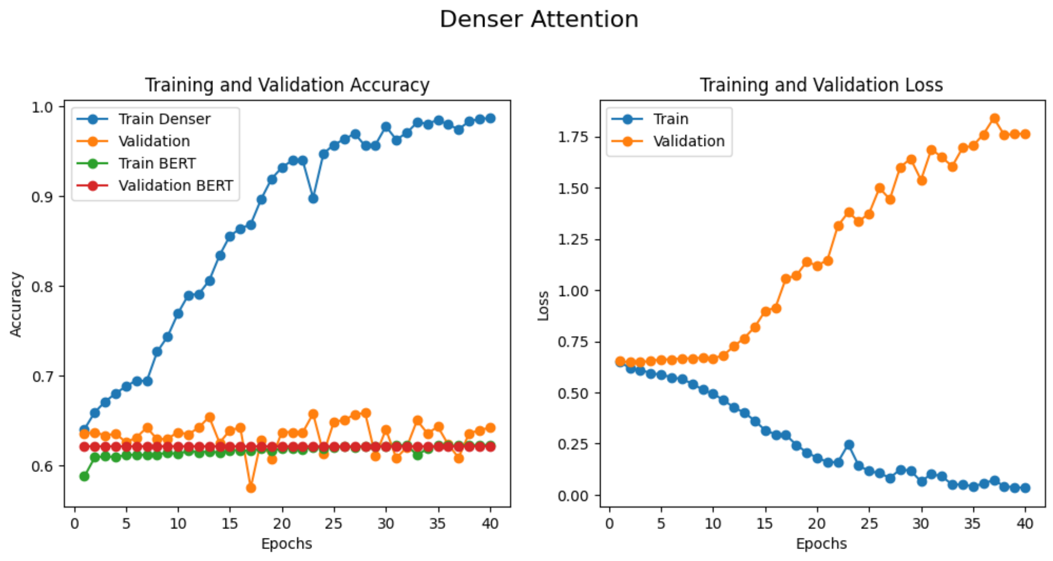
Instead, in denser attention, we calculate the following for each q, k items. q, k are segmented into q^i, k^i where $i \in \mathbb{N}, i \leq w$. Also, the attention output is now set of size equal to number of segments.

$$\text{attention}(k, q, v) = \{\frac{1}{w}\sum_{j=1}^{w} \text{softmax}(\frac{q^i(k^j)^T}{\sqrt{d_k}})v^i | 1 \leq i \leq w\} \tag{2}$$

Because of the segments, the output dimension of attention is now w times that of original attention.We added a dense layer to ensure that output shape is same as that of the input shape.

We set $w = 2$ for our experiment. Also, there are no position incorporating methods in this experiment. Furthermore, the model has only 4 layers of attention since we hypothesize that denser attention requires less layers to comprehend compared to normal attention.

3.1.2 Result for Denser Attention

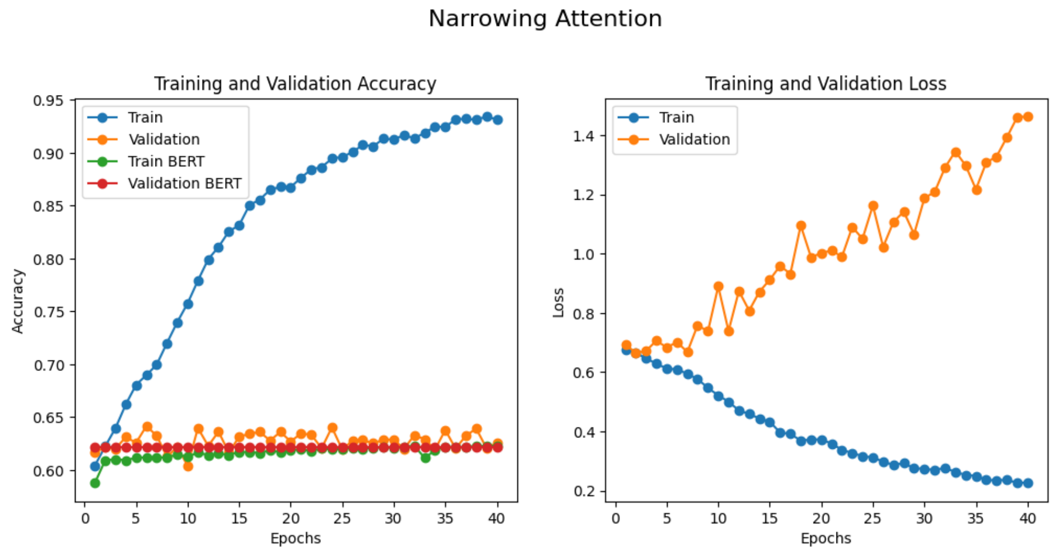


The model was able to overfit with the given data. We see that validation accuracy of denser attention is higher than that of both train and validation accuracy of BERT from HuggingFace. The additional memory use is less than twice per layer.

3.2 Narrowing Attention

During the construction of the transformer, attention layers are layered one on top of the other. Each of these attention layers typically comprises multiple heads, all of which frequently have identical weight dimensions across the stacks. Taking inspiration from the standard architecture of a Convolutional Neural Network, we arranged the attention layers so that the size of the hidden layers per head diminishes as the number of heads increases the deeper into the model. Specifically, with each attention layer, the number of heads doubles while the number of hidden layers per head reduces by half.

3.2.1 Result for Narrowing Attention



We see that Narrowing Attention model quickly overfits. We see that in average, the validation accuracy from narrow attention is higher than that of train accuracy and validation accuracy of BERT. The model, however, has some weakness. However, the model does come with certain shortcomings. In the implementation I conducted, the training time quadrupled compared to an attention model with an equivalent number of parameters. This model also consumed four times more memory compared to an attention model with the same parameter count. This could possibly be attributed to the increased connections in the narrower attention.

3.3 Conclusion

Both the denser attention and narrowing attention models demonstrate superior performance compared to the standard BERT model. However, they do have certain drawbacks. Both models require more training time compared to the standard BERT. Also, narrowing attention model also needs approximately four times more memory.

4 Possible Improvements and Future Steps

One major mistake in the project was that the dataset was biased. Overall, there the true to false ratio was around 62 % true. If we knew this from the start, we would have balanced the dataset or made confusion matrix for more precise results.

Furthermore, the BoolQ dataset or the IMDB dataset is small for model like BERT. BERT was originally trained on 3.3 Billion words total while our dataset only had millions of words. Furthermore, researches such as [Press et al., 2021], [Verma and Berger, 2021] only train the model with 1 or 2 epochs on a large corpus. We trained 40 epochs on same corpus. Next time, we should try following the fewer epoch training method and train on a larger corpus such as wikitext.

One major mistake in our implementation was the lack of attention masking. In encoder architecture, it is normal for the padded tokens to be masked so that they do not have influence in calculation of other attentions and trainable positional embeddings. Our model did not have such safeguarding tool. The code for attention masking should be included in future exploration.

All of the models above use same input max length and max embeddings length of 512. However, reducing the max input length to 128 greatly increases the effectiveness of the model. The attention only model achieves 0.628 accuracy for train data with max input length of 512. The same model achieves 0.97 accuracy for train data of input length of 128. The reason behind this discrepancy is a topic for future research.

Expanding on this, all models share identical configurations including the number of attention heads, dropout probability, encoders, learning rate, and optimizer. Future research may experiment with modifications to these configurations.

Further exploration into denser attention seems to hold potential. The denser attention layer demands the similar RAM capacity as a standard attention layer while marginally increasing the training time. We hypothesize that the increased training time is outweighed by the level of compression offered by denser attention.

References

D. Bahdanau, K. Cho, and Y. Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL `http://arxiv.org/abs/1810.04805`.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *CoRR*, abs/2010.11929, 2020. URL `https://arxiv.org/abs/2010.11929`.

A. Haviv, O. Ram, O. Press, P. Izsak, and O. Levy. Transformer language models without positional encodings still learn positional information, 2022.

A. Katharopoulos, A. Vyas, N. Pappas, and F. Fleuret. Transformers are RNNs: Fast autoregressive transformers with linear attention. *arXiv preprint arXiv:2006.16236*, 2020.

O. Press, N. A. Smith, and M. Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. *CoRR*, abs/2108.12409, 2021. URL `https://arxiv.org/abs/2108.12409`.

P. Shaw, J. Uszkoreit, and A. Vaswani. Self-attention with relative position representations. *CoRR*, abs/1803.02155, 2018. URL `http://arxiv.org/abs/1803.02155`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

P. Verma and J. Berger. Audio transformers: Transformer architectures for large scale audio understanding. adieu convolutions. *CoRR*, abs/2105.00335, 2021. URL `https://arxiv.org/abs/2105.00335`.