Benjamin Lapostolle
benjamin.jean.lapostolle@gmail.com

Lab session # 3
ALTEGRAD 2023

10/31/23

links:
https://machinelearningmastery.com/a-gentle-introduction-to-positional-encoding-in-transformer-models-part-1/ https://jalammar.github.io/illustrated-transformer/

1 Question 1

The role of the square mask is to prevent our model from accessing future information in the input sequences. By using a square mask, we manually set the attention scores between our current sequence with the future words to 0. This masking is crucial to prevent the model from "cheating" during training by looking at tokens from the future that it wouldn't have access to during the actual prediction.
Unlike recurrent neural networks (RNNs) or convolutional neural networks (CNNs), Transformers do not inherently preserve the order of the sequence. The positional encoding is therefore used to encode the position of the words in the sequence in our embeddings to provide our model with information about the positions. Cosine and sine functions are used to encode the position because the neural network can more easily learn thanks to the linear properties of these functions.

2 Question 2

We have to replace the classification head because we want to perform two different tasks using the same base model. This process is called transfer learning. As other examples in computer vision involves using pretrained convolutional network trained on ImageNet and only changing the last layer to benefit from the feature extraction, we use it here to change our goal from predicting the next words in a given sentence to predicting the sentiment of a whole sentence.
The main difference between the two tasks is that for the text prediction task, we want to obtain a vector of the size of the vocabulary to compute what is the next word among our vocabulary with the highest probability given our input. In the sentiment analysis task, we want to provide directly the full sentences as input and we want only a classification head with dimension 2 to predict the sentiment of the sentence.

3 Question 3

Base Model:

- **Encoder**: The size of the embedding layer is the size of the vocabulary (n_{tokens}) times the n_{hid}.
- **Positional Embedding**: No trainable parameters.
- **Transformer Encoder Layer**:
  - We have N_{layers} so we compute the number of parameters in one layer and then multiply by N_{layers}.
  - **Multihead Attention**: This comprises linear projections for input (Q, K, V) and output transformations. The parameters for each of Q, K, V are nhid $\times$ nhid, and the output linear transformation uses nhid $\times$ nhid parameters. Therefore, for a single attention head, the total parameter count amounts to $4 \times$ nhid $\times$ nhid. Given there are nhead heads, the total parameters for Multihead Attention equal $4 \times$ nhead $\times$ nhid^2.
  - **Feedforward Network:**: This section comprises two linear layers. The initial layer projects from nhid to dim_feedforward (equivalent to nhid), and the subsequent layer projects back. The total parameter count amounts to $2 \times$ nhid $\times$ nhid.

Classification Head:

- **Text prediction:** we have a linear layer with bias so $n_{hid}n_{tokens} + n_{tokens}$
- **Sentiment Classification:** we have a linear layer with bias so $2n_{hid} + 2$

Finally, we have:

$$N_{param} = n_{token}n_{hid} + N_{layers}(4n_{head}n_{hid}^2 + 2n_{hid}^2) + (n_{hid} + 1)n_{class}$$

For:

- $n_{tokens} = 50001$
- $n_{hid} = 200$
- $N_{layers} = 4$
- $n_{head} = 2$
- $n_{class} = 2$ or 50001

We have: $N_{text_prediction} = 21650401$ and $N_{sentiment} = 11600602$.

4 Question 4

As we can expect it, the accuracy for our model randomly initialized is close to 50% at the beginning, meaning that it is random. On the other hand, the accuracy of the pretrained model is 65%, meaning that even though there is still room for improvement because our model needs to train to learn the sentiment analysis task, it starts with a good accuracy because the feature extraction of this pipeline is already trained.
However, the differences between the two models start to vanish when the number of epochs increases because the weights of the base model starts to converge. To conclude, we see that using pretrained weights allow to have a faster convergence to the optimal state, which is really convenient when the training of the base model is demanding a lot of resources.

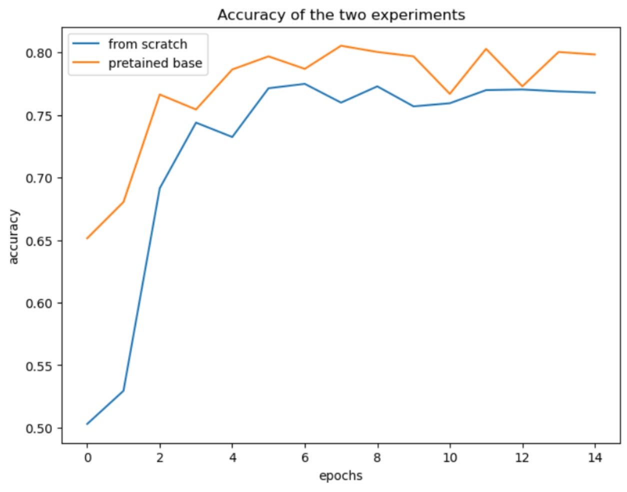

Figure 1: Accuracy of the pretrained and randomly initialized model

5 Question 5

In the paper [1], the authors use a similar architecture as the one presented in the "Attention is all you need" paper. In our notebook, language modeling objective employed relies solely on left-context for task execution, rendering it unidirectional. This limitation can significantly constrain tasks demanding a comprehensive comprehension of context or those involving contexts beyond a word's left side (such as resolving ambiguities or understanding pronoun references). To mitigate this issue in BERT, the authors apply mask to the input sequences, authorizing the model to use context in both directions.

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.