Axel Dumont
axel.dumont@ensta-paris.fr

Lab session # 3
ALTEGRAD 2023

10/31/23

1 Question 1

The square mask in our Transformer model implementation serves the purpose of preventing illegal connections in the self-attention layers. Specifically, it masks out the subsequent positions so that self-attention only looks at previous positions when generating the representation for the current position.
The mask is a square matrix with 0s on the diagonal and bottom triangle, and -inf in the upper triangle. During self-attention computation, this mask ensures that:

- Each position can attend to itself (the diagonal 0s)
- Each position can only attend to previous positions, not future ones (the -inf values prevent attention to future positions)

This allows the self-attention layers to respect the sequential order of the text and prevents using future context when encoding the current position.
The positional encoding provides the model with information about the position of each token in the sequence. Since the model has no innate notion of order, adding positional encodings gives the model knowledge of the ordering of the sequence. This allows the model to distinguish the relative position of tokens and encode positional information. The positional encodings are added to the token embeddings before feeding into the Transformer layers.

2 Question 2

The main difference between the language modeling and classification tasks is the output layer/head of the model.
In language modeling, the model is trained to predict the next token given the previous context. So the output layer is a linear layer with number of outputs equal to the vocabulary size (*ntoken* in the code). This allows the model to predict a probability distribution over all possible next tokens.
In classification, the model needs to predict a class label. So the output layer is a linear layer with number of outputs equal to the number of target classes (*nclasses* in the code). This allows the model to predict a probability distribution over the class labels.
Therefore, the classification head needs to be replaced between pretraining and fine-tuning stages. During pretraining on language modeling, a vocab-sized output layer is used. For classification, this is replaced with a class-sized output layer initialized randomly.
The base Transformer model and word embeddings can be transferred between tasks, but the final task-specific output layer needs to change to match the different targets.

3 Question 3

First we need to count the parameters in the encoder. This is quite simply $p_1 = n_{hid} \times n_{tokens}$
For the rest, we ignore the parameters induces by the biases of the different layers. We are taking for reference the part 3 of the article [2]
In the transformer layers, we have a self-attention layer with the Query, Key and Value matrices of dimension $n_{hid} \times n_{hid}$. We also have the out matrix W^O of size $n_{hid} \times n_{hid}$ parameters
Hence for the self-attention layer we have : $p_2 = 4 \times n_{hid} \times n_{hid}$
We then have two linear layers with the feed forwards that hold both $n_hid \times n_{ff} = n_{hid}^2$ that make in total $p_3 = n_{hid}^2$ parameters. Both are normalized which add up n_{hid} parameters to the count so $p_4 = 2 \times n_{hid}$
Hence for the language modeling task we have :

$$n_1 = \sum_{i=1}^{4} p_i = n_{hid} \times n_{tokens} + 4 \times n_{hid}^2 + 2 \times n_{hid}^2 + 2 \times n_{hid}$$

$$n_1 = 6n_{hid}^2 + n_{hid}(n_{tokens} + 2)$$

For the classifier task we must add the parameters associated with the ClassificationHead, which is just $p_5 = n_{hid} \times n_{tokens}$, so the total numbers of parameters for the classification task is :

$$n_2 = 6n_{hid}^2 + 2n_{hid}(n_{tokens} + 1)$$

4 Question 4

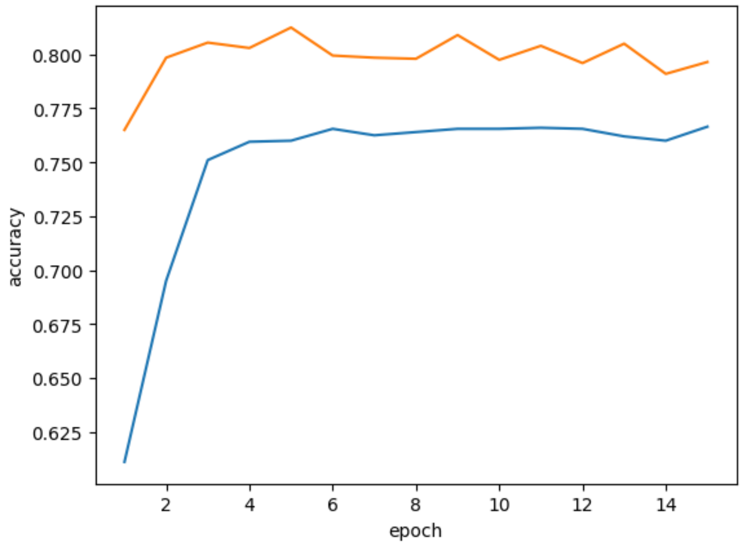

Figure 1: Accuracy of the two models

We see that the model from scratch is learning fast in the first epochs but is beginning to reach a cap at the 4th epoch (towards 76% accuracy). However the pretrained model, as expected, is performing way better and in a somehow constant accuracy.

5 Question 5

One of the limitations of the language modeling objective used in the notebook is that it only uses the left context to predict the next token. This means that the model cannot leverage the information from the right context, which might be useful for disambiguation or coherence. For example, given the sentence “He went to the bank to ...”, the model cannot use the word “deposit” that might appear later in the sentence to predict the missing word.
The masked language model objective introduced in [1] overcomes this limitation by randomly masking some tokens in the input and predicting them using both left and right contexts. This way, the model can learn from bidirectional dependencies and capture more semantic and syntactic information.

References

[1] Kenton Lee Jacob Devlin, Ming-Wei Chang and Kristina Toutanova. Attention is all you need. In *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2018.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.