KALLEL Mahdi
mahdi.kallel@telecom-paris.fr

Lab session # 2
ALTEGRAD 2020

12/04/20

1 Question 1

As a reminder, when making training the language model, the target sentence is the input translated by one. When encoding for a single word, the transformer encoder uses self attention over it's **entire** input to have a more contextual representation of that word. The mask serves such that we cannot use attention of words we haven't predicted yet.
Unlike it's RNN counterpart , an attention encoder-decoder does not leverage any information about the relative ordering of the words. So we add positional encoding to the output of the embedding layer so that the word encodings take into account it's position in a sentence.
For example if we take these two sentences :

"This movie was not good."
" I did not imagine it would be this good".

The further the "Not" from "Good" the higher the chance that it does not apply to it.

2 Question 2

In the "language modeling" task, consists of predicting the next word of a sentence, the classification head must have as many outputs as the size of the vocabulary.

For the classification task we look at the whole sentence and output a binary value saying if it's a positive or negative review, this requires having only two output neurons.

3 Question 3

The embedding layer is a matrix of size [n-token ,hidden-size] = [50001,200]

For positional encoding, we have a matrix of sines and cosine values for n-hidden different frequencies. [n-pos,n-hidden]=[5000,200]
The rest of the encoder consists of x4 Transformer blocks each block with x2 multi-head attention layers.
For one attention block we have :

- 6 Linear layers with a size of n-head * 3 * [n-hidden,n-hidden] correspond for the Key,Value,Query matrices. One for each head.
- Two batch normalization layers, one for each head n-head * [n-hidden,2] = 2*[200,2] (200 for the biases , 200 for the variances).
- One [n-head * n-hidden , n-hidden]= [600,200] + 200 [biases] layer (the feed forward matrix)

Finally we have the classification head which consists of :

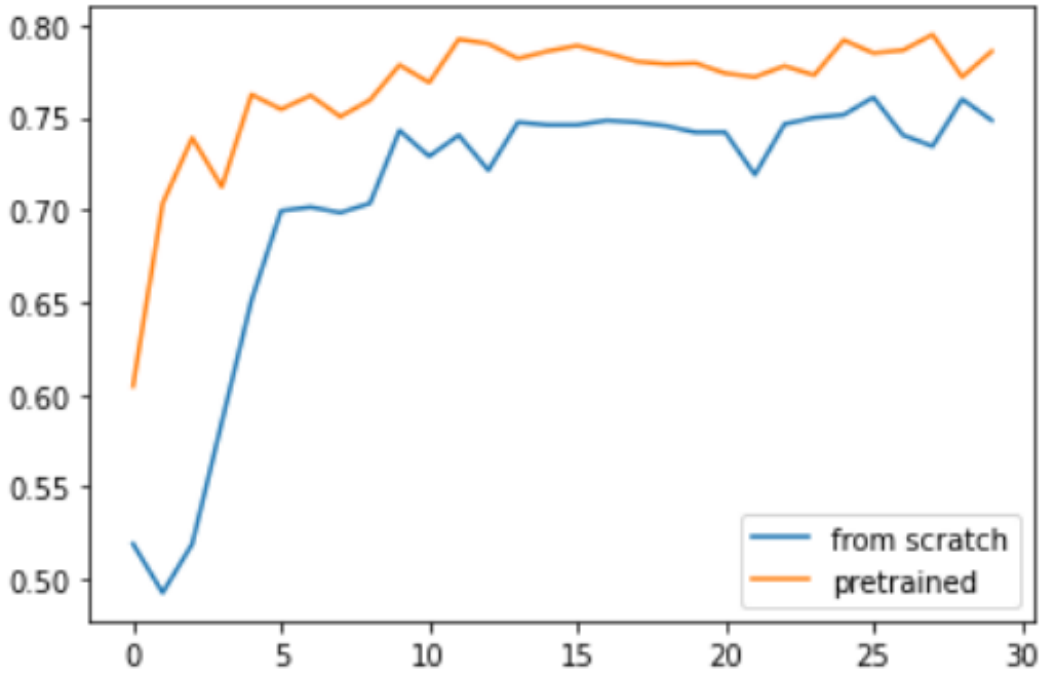
- A linear layer of size [n-hidden,n-token] + n-token (biases) in the case of "language modeling"
- A linear layer of size [n-hidden,2] + 2 (biases) in the case of "sentiment analysis"

4 Question 4

In the figure below we report on our validation accuracy for both the models "trained from scratch" and "pretrained".

The pretrained model easily outperforms it's counterpart. This is especially significant because our training dataset is small (around 3000 examples). Having pretrained on a general text corpus allows our model to find

a more general representation of words and thus the sentence. It is also possible that it encounters some of the test sentences in the ”self supervised” training corpus.
The pretrained model reached it’s peak performance after only 10 epochs while the one from scratch takes around 25 epochs to reach it’s peak.



5 Question 5

In the transformer network, each token can only attend to previous tokens.

Such restrictions are sub-optimal for sentence-level tasks, and could be very harmful when applying fine tuning based approaches to token-level tasks such as question answering, where it is crucial to incorporate context from both directions.

Albeit, these restrictions come more handy in the case of text generation (like GPT, and the Transformer network) where BERT is simply not adapted.

In the Bert paper, the authors solve this problem by training the model on two objectives. The masked word objective where the goal is to predict a randomly masked word from a sentence (using the rest of the sentence as context) and the ”IsNext” objective where the goal is to classify two sentences a consecutive or not.

References