Bert Attention Analysis
Natural Language Processing 2
Mihai Matei 511DS

The project was implemented with the latest tensorflow 2.4 and their own keras implementation using a 1070ti Nvidia graphics card. Pandas was used to per-process the dataset. My own custom ***matmih*** library was used for plotting, statistics and model building.

Transformer based language modeling has been seen as a recent advancement in the field of natural language processing, especially regarded as a way to learn discriminative features of the language. It's configurable high capacity modeling can be used to learn a vast amount of data that exists in a certain language domain and can be easily applied using transfer learning to other task.

This project tries to train the vanilla Google's implementation of a Transformer, based on their own implementation of the Self-Attention mechanism on a Romanian dataset. https://github.com/tensorflow/models/tree/master/official/nlp.

The focus of the project is not on the accuracy of the classification results but rather on how per-training the language model influences the results and especially the learned attention that the model focuses on the sample text sentences.

Due to hardware constraints I was able to due the analysis only on a small subset of the MOROCO dataset https://github.com/butnaruandrei/MOROCO used in VarDial 2021 competition which tries to discriminate Romanian and Moldavian dialects.

Experiment setup

As mentioned Google's EncodeStack implementation used in their Bert model was used using the default options used for Bert. The only difference is that the number of layers was limited to 4 from 12 (Bert's implementation) which is similar to RoBERTA which is an out of the box Bert implementation for the Romanian language.

The MOROCO dataset was loaded and only the cultural type of texts were kept. Text per-processing was done in the sense that the text was lowered, split into sentences of maximum size of 128 and the punctuation was removed.

I created 2 type of models – one for language model per-training and one for classification. The language model was trained on all of the data with different masking and the resulted weights were transferred to the classification model for dialect identification. The normal machine learning results were also recorded, such as the accuracy, ROC curve and so on, but the focus was on the attention heatmaps learned and their transformation after classification. For this some changes needed to be done to the default Google's EncoderStack implementation so that the output values at each of the Self-Attention layer were recorded.

Language Model per-training

The first model used 4 EncoderStack and a final 128x8289 (vocabulary size) dense layer using softmax activation. As such the model would learn the probability of a vocabulary word to be on a position of the sentence. The input are the vocabulary identifiers of the words in a sentence fed into and 768 size Embedding layer to learn weights for each word. The only difference is that Bert offers a smart was of training a language model by randomly mask specific words in the input and force the model to learn the missing words. This means that the cross-entropy validation loss should be applied only to the masked words in the input sentence as we do not care about the other words prediction. Different masking techniques were used, each applied randomly at each epoch for each sentence:

- randomly mask 15% of the entire sentence

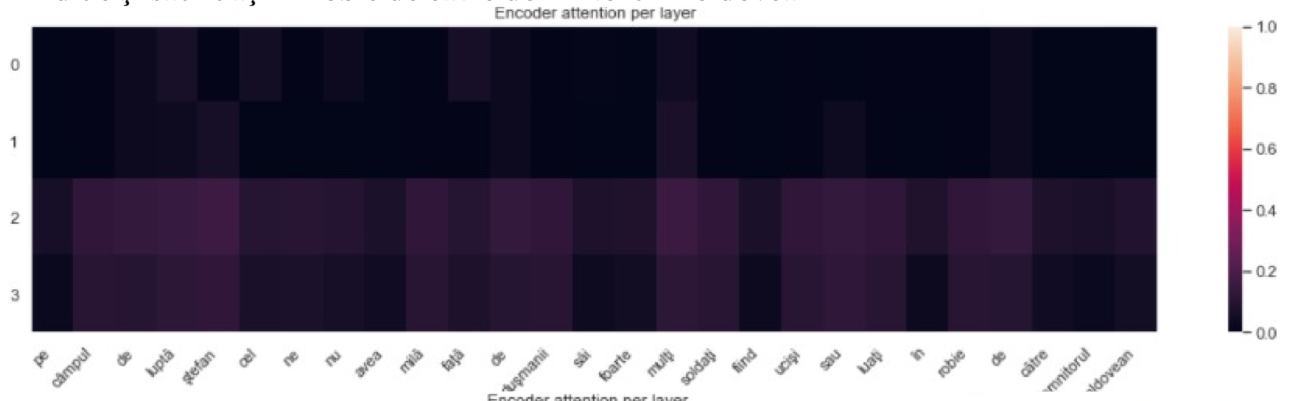
- randomly mask 15% of only the words with length >5
- randomly mask 15% of the words with length <4

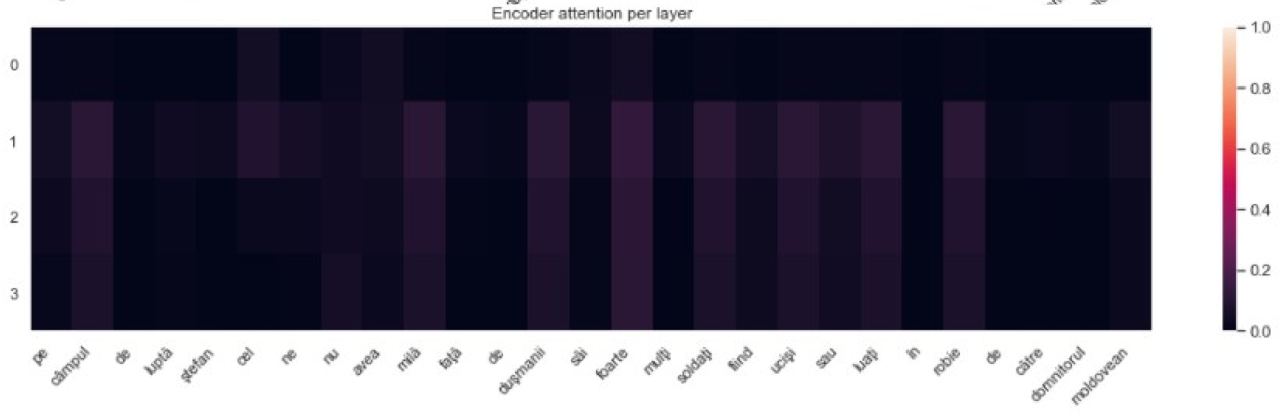
The 3 models were trained for 30 epochs for about 6 hours and yielded 0.22, 0.56, 0.48 accuracy respectively. This is enough for the model, given an input sentence with a single masked word to accurately predict that word, most of the time.

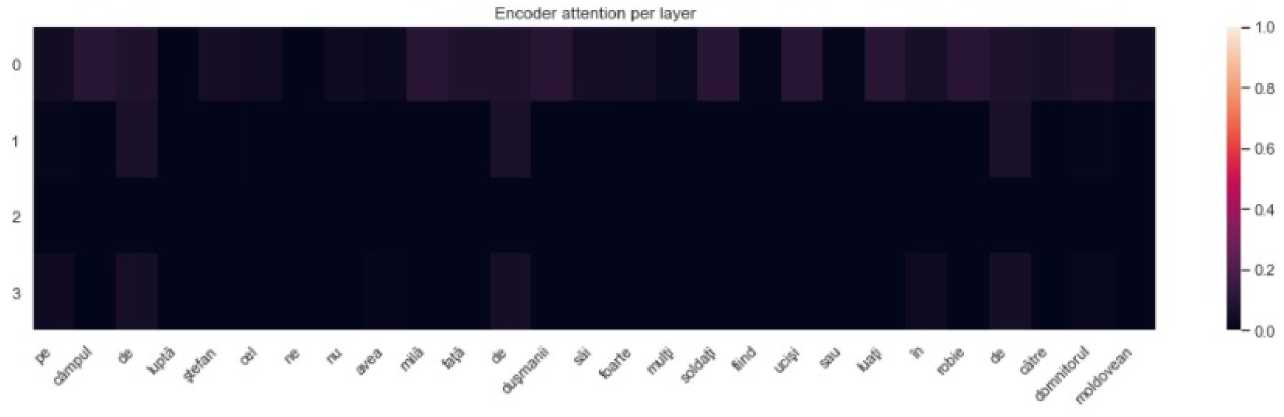
Since I was interested in how the attention changes depending on the masked words, the attention heatmaps were plotted for each level of the 4 Encoders (level 3 is the last one prior to prediction):

For the sentence:

Pe câmpul de luptă, Ştefan cel NE nu avea milă faţă de duşmanii săi, foarte mulţi soldaţi fiind ucişi sau luaţi în robie de către domnitorul moldovean



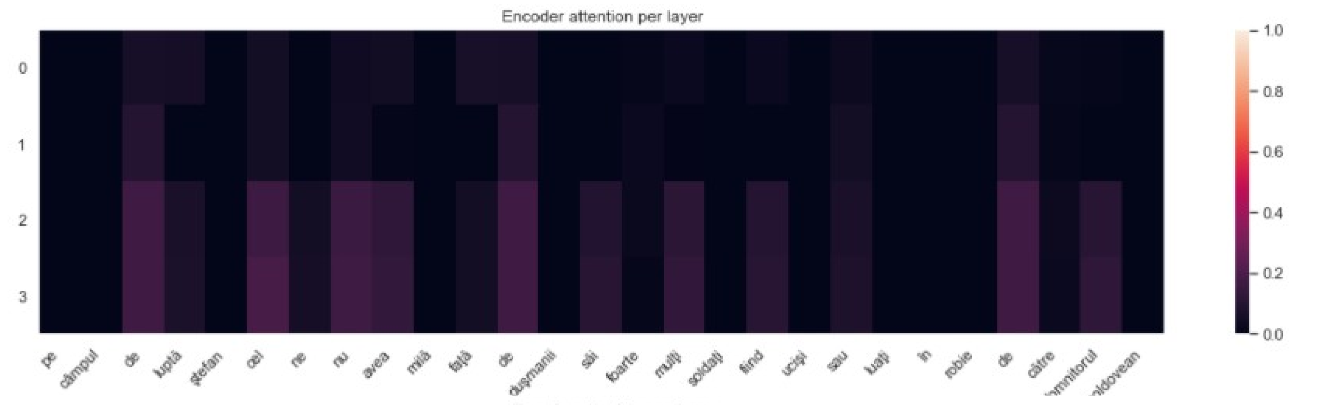


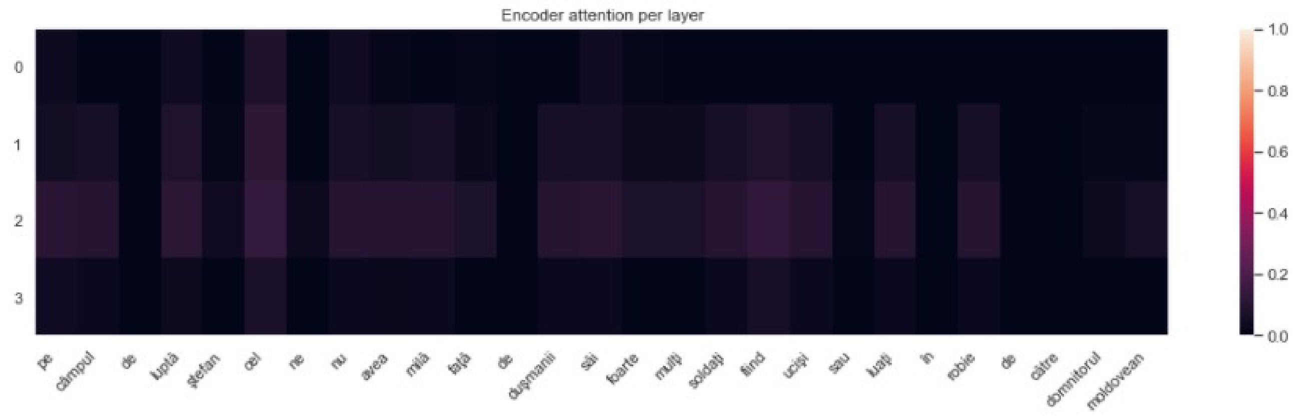


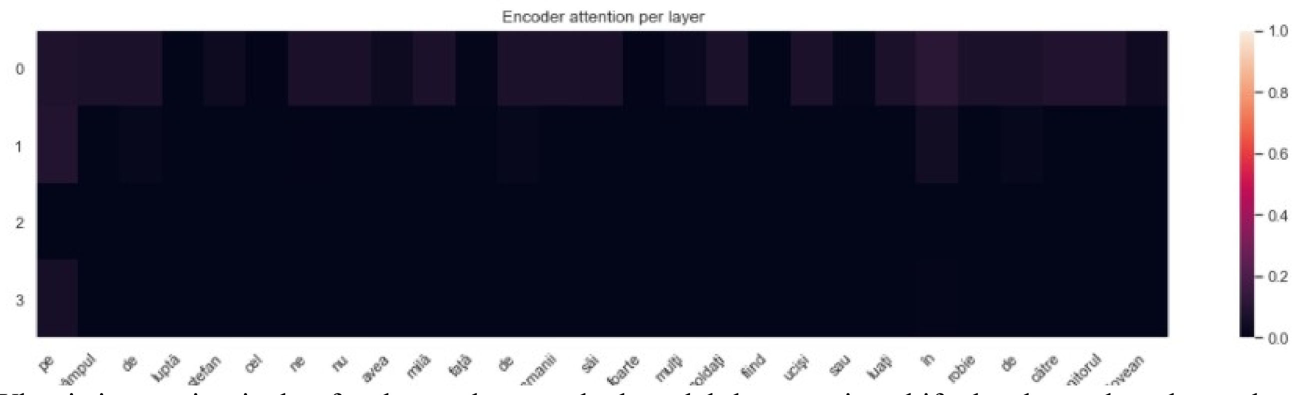
the above heatmaps were obtained. Clearly one can see that when randomly masking words the attention seems to be evenly divided between all words and when the masked words have length greater than 5 this is shown that only those words received higher outputs. When we mask words with length at most 3 all words receive more vague attention in the last layer but still the smaller words are a bit more important.

<u>Transfer learning – classification model</u>

The 3 learned language models above were used in a classification model for dialect identification. The resulting model over-fitted very fast in just 2 epochs. The test accuracy was around 0.86 which is way below the state of the art of ~0.95. But one must keep in mind that only a small part of the MOROCO dataset was used, namely the culture category. Nevertheless I was interested in how the attention heatmaps changed when the model was retrained starting from the learned language model weights but with a much smaller learning rate. The resulting attention heatmaps for all 3 models are:







What is interesting is that for the random masked model the attention shifted to lower length words like pe, de and some larger ones. For the model where only larger than 5 characters were masked the attention kept some of the larger words but also slightly moved to lower length words. For the last model trained by masking only words with a maximum of 3 character the attention did not change that much which much imply that the model did not need to learn different discriminative values.

Conclusion and future work

I have shown that masking specific words does count on what the model is paying attention to. Also by retraining the model on a specific task one can see by comparing the 2 attention heatmaps what words are actually discriminative for that specific task.

The models do need to be trained on a larger dataset for the results to be more conclusive. Also statistical tests must be done on the overall distribution of the attention mechanism.