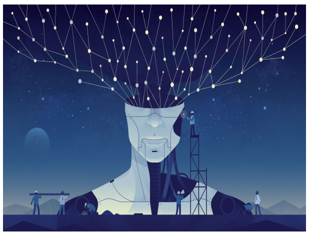
Shakespeare style transformer



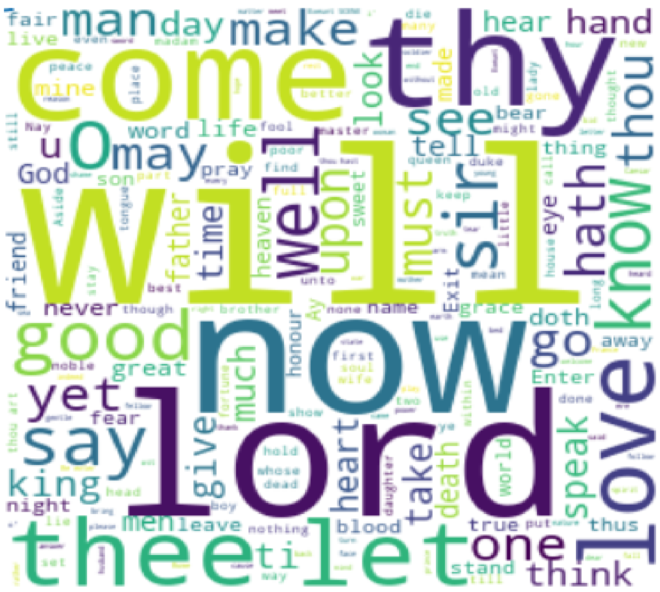
Executive summary and conclusions

The aim of the task was to create a transformer language model which will generate a text with Shakespeare style. The model should get the length of desired text (about 400 - 600 chars) and generate a text close to the length. Shakeseare dataset used in this report is from here. As a result of time and space constraints,we have decided to train the transformer model with 3 epochs which in turn led to high Crossentropy loss (0.7), so random guesses of the next word is better than using the below described transformer model. Also predicting the text with 400-600 characters isn't possible at the moment, as the algorithm is weak and after some text generation it says that there is higher probability that there are no other words. So we can conclude that further analysis on a computer's memory efficient management during the training of a deep learning model and transformer training with more epoches can raise the performance of the model.

Exploratory data analysis

As the aim of the project is to generate new words, we have used only the 'PlayerLine' column from the Shakespeare dataset, where we have 111.396 rows of strings. Without removing stop words and punctuation marks there are 27.381 unique tokens in the dataset and after the removal of mentioned tokens we have 27.245 unique words. Inspite of these facts we will set vocabulary size 2000 in our model. This has been done just for understanding the structure of the dataset, because in the transformer model we will also use stop words and punctuation marks to make generated text more similar to normal texts. Figure 1 below shows the word cloud of the Shakespearian vocabulary.

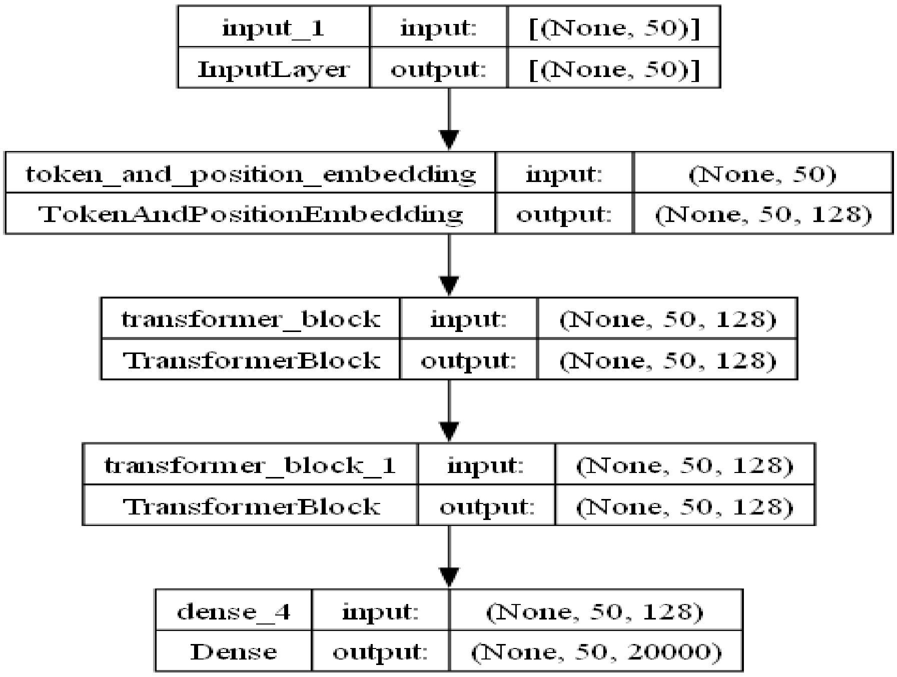
Figure 1. Wordcloud of Shakespearean vocabulary



Transformer language model

For model creation these hyperparameters have been used: vocab_size = 20000, batch_size = 64, maxlen = 50, embed_dim = 128, num_heads = 2, feed_forward_dim = 256 and num_transformer_blocks = 2. Model's structure is represented below.

Figure 2. Model structure



input_1 | input: | [(None, 50)]
InputLayer | output: | [(None, 50)]

token_and_position_embedding | input: | (None, 50)
TokenAndPositionEmbedding | output: | (None, 50, 128)

transformer_block | input: | (None, 50, 128)
TransformerBlock | output: | (None, 50, 128)

transformer_block_1 | input: | (None, 50, 128)
TransformerBlock | output: | (None, 50, 128)

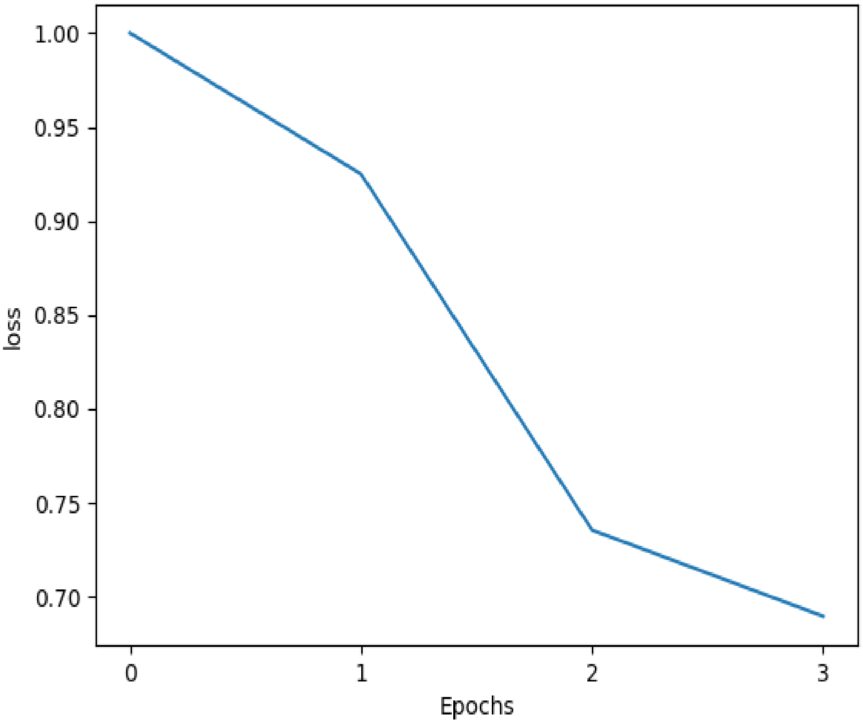
dense_4 | input: | (None, 50, 128)
Dense | output: | (None, 50, 20000)

The summary of the trained model is shown below. In Figure 3 cross-entropy loss for each epoch is represented.

Summary: Transformer model's summary

```
Model: "model"
_________________________________________________________________
 Layer (type)             Output Shape           Param #
=================================================================
 input_1 (InputLayer)       [(None, 50)]            0

 token_and_position_embeddi  (None, 50, 128)          2566400
 ng (TokenAndPositionEmbedd
 ing)

 transformer_block (Transfo  (None, 50, 128)          198400
 rmerBlock)

 transformer_block_1 (Trans  (None, 50, 128)          198400
 formerBlock)

 dense_4 (Dense)           (None, 50, 20000)        2580000

=================================================================
Total params: 5543200 (21.15 MB)
Trainable params: 5543200 (21.15 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 3. Cross-entropy loss for each epoch



The model was trained for 2.5 hours(see log_history.txt). During each epoch we generated text to see how the model is improving after each epoch. After the first epoch we have the sentence 'to be so much but i am so so much but i am so' and after the second epoch it changed to 'to be so much but i am so'. After training the model we tried to use the positive part ('To be') from the most famous citation of Shakespeare ("To be, or not to be: that is the question.") and our transformer gave us equally rhetorical answer 'and the king is well the prince i must be a'.