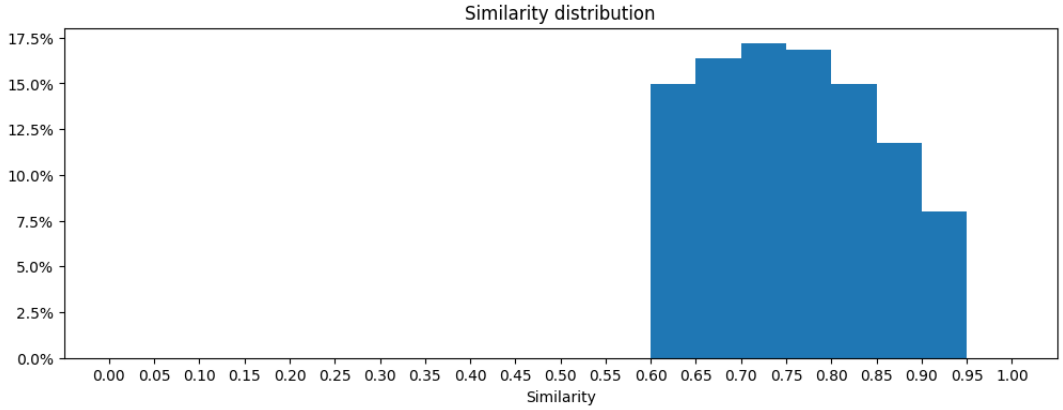
PMLDL Assignment 1, Solution building

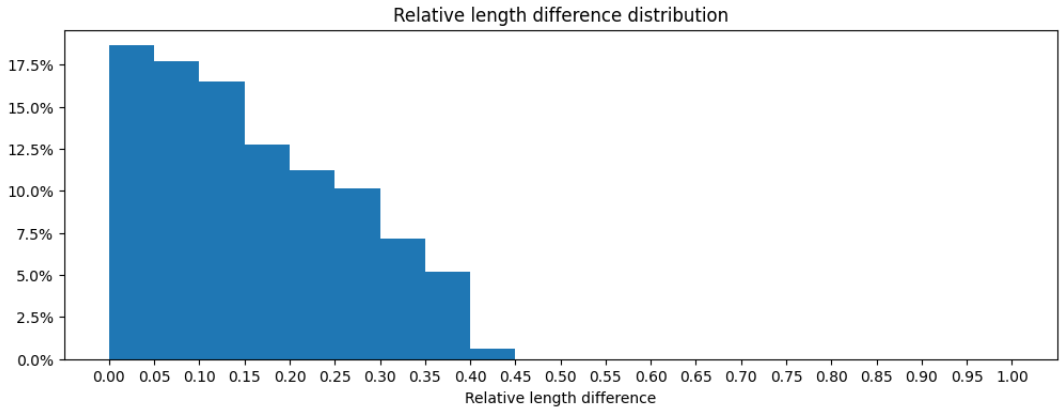
Ilia Milioshin, i.mileshin@innopolis.university, B20-RO-01

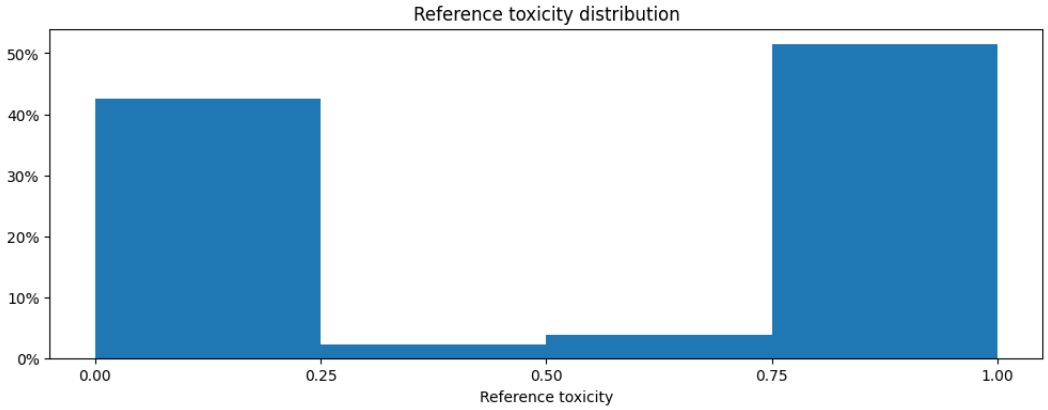
October, 2023

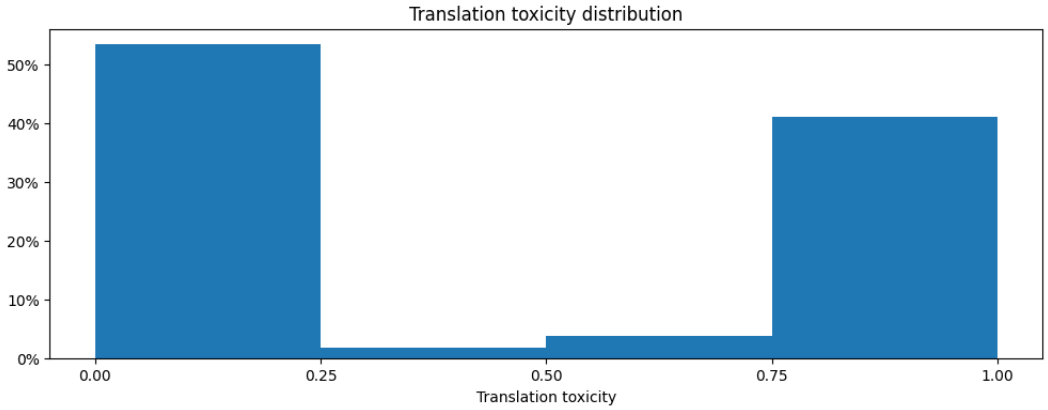
1 Data exploration

At first, I decided to explore the given dataset. I built several distributions: similarity, relative length difference, reference toxicity, and translation toxicity.



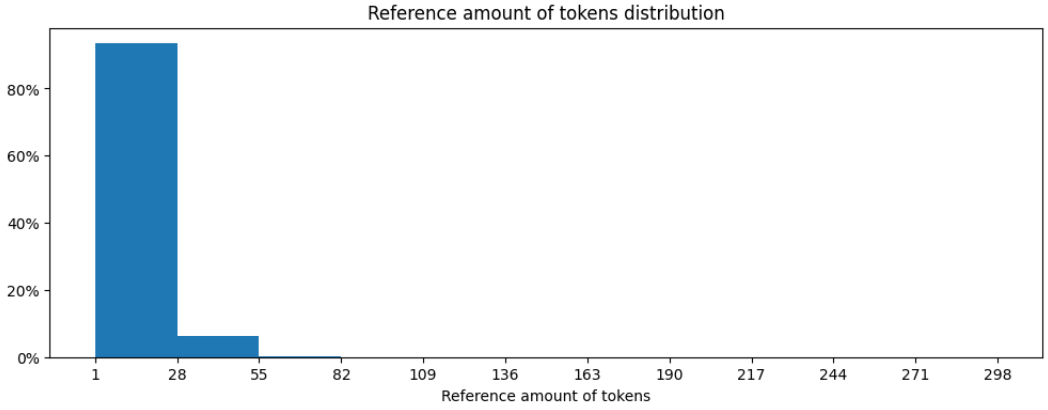


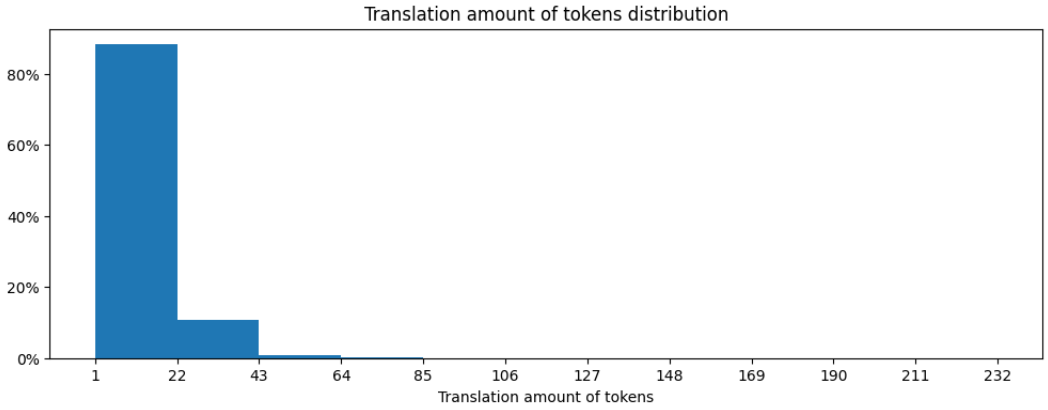




After exploring the distributions, I saw that almost 95% of all data was either highly toxic or not toxic at all. Moreover, I did not have much GPU computation power to process all pairs of sentences. Thus, I decided to take only such pairs where the reference has toxicity greater than 0.75 and the translation has toxicity lower than 0.25.

Now let's consider the amount of tokens distributions.





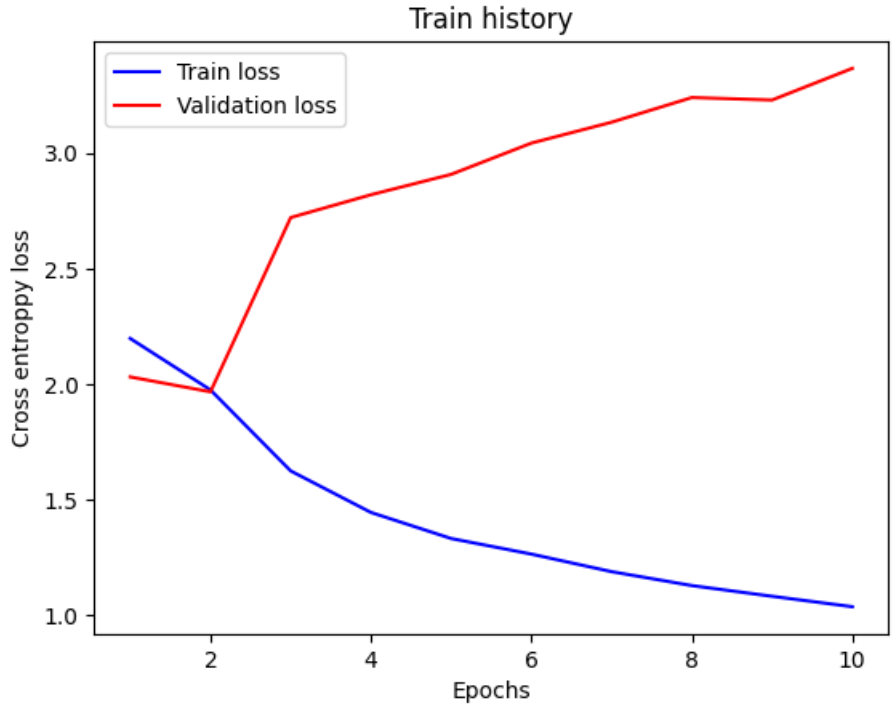
We can see that almost all sentences contain no more than 50 tokens. So, I decided to take into training datasets only sentences with up to 75 or 150 tokens.

To conclude, I reformulated the task of de-toxification to the translation problem. Due to the high similarity of reference and translation, the task seemed handleable. "High similarity" means that cosine similarity is close to 1 and the relative length difference is low.

2 Encoder-decoder

My first attempt was to use encoder-decoder architecture. Based on this tutorial I created the model with the Bahdanau attention mechanism. I used custom vocabulary with a custom embedding. As the last layer, I decided to put a feedforward network. The loss function was a cross-entropy that ignored the padding index. For validation, the Sacrebleu score was utilized.

I trained this network on 50000 pairs with ten epochs. Below you can see the training history.



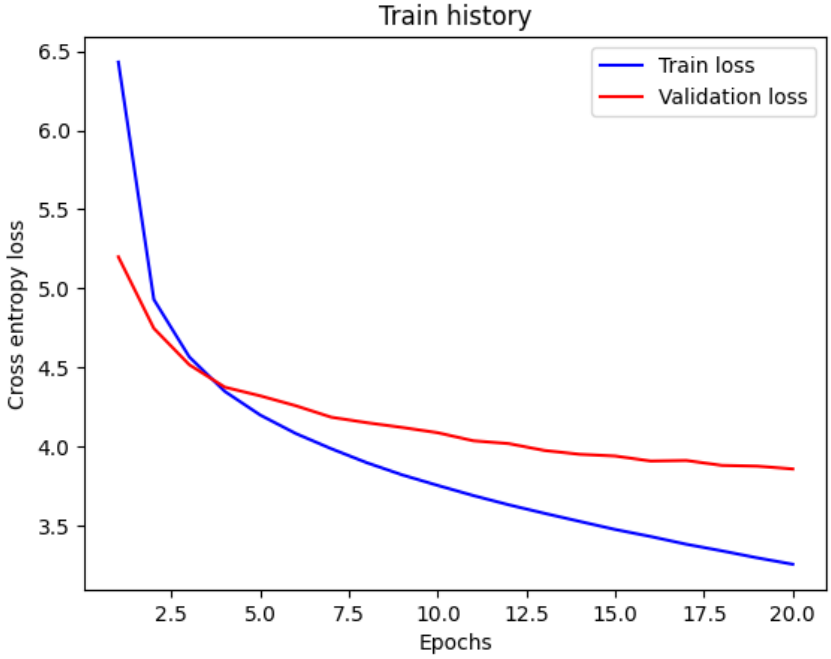
As we can see the model critically overfits. However, the Sacrebleu score increases from

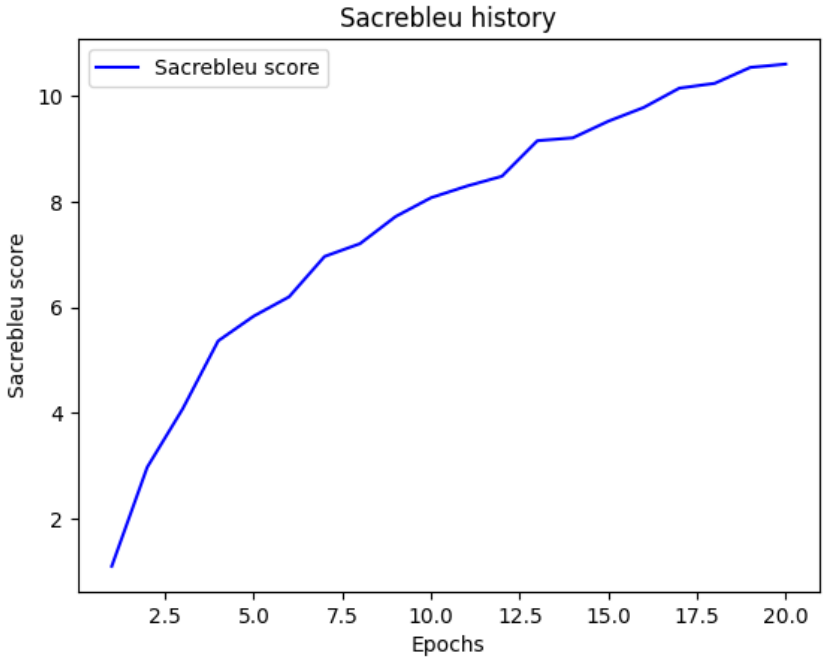
epoch to epoch. You can check it in the encoder-decoder.ipynb. I think a high score connected the model understood that translation should have the similar length.

Example prompt: *You are dirty bastard! >>You're a monster!*

3 Transformer

The second thought was a use the transformer. To build the model I used the PyTorch tutorial. At the initial stage, I tried to use the GloVe vector sets. However, due to its size, the feedforward layer became too large to be trainable. Thus, I returned to custom vocabulary and embeddings. The loss function, validation score, and training parameters were the same as in 2. (20 epochs were used)





The Sacrebleu score in the transformer case is higher. Example prompts:
You are good boy! >>You're a bad guy!
You are dirty bastard! >>You're the man.
Hitler is a nice guy! >>he's a bad guy!

4 Results

As we can see, the model made from scratch did not have good performance.