Report 1 for Text Detoxification task

`0.0-loading-and-preprocessing-data.ipynb` contains initial thoughts about the data and preprocessing function that will be updated further

The model chosen for the task is Encoder-Decoder model with Dot Product Attention mechanism.

`1.0-first-runnable-prototype.ipynb` implements this model.

Its structure:

1. **Data downloading and preprocessing**
    a. Preprocessing part entirely from `0.0-loading-and-preprocessing-data.ipynb`
    b. Constituting dataset (input dataset contains sentences that have greater tox score among the two)
    c. Fitting and applying tokenizer
    d. Padding
    e. Train-test split (80/20)
    f. Forming batches
2. **Model implementation**
    a. Encoder implementation
    b. Dot Product Attention layer implementation
    c. Decoder implementation
3. **Training**
    a. Training and validation function implementations
    b. Code to run them

Development process

1. **Dataset filtering and preprocessing improve**

   `1.0-first-runnable-prototype.ipynb` was indeed runnable but took forever to train. This was due to large sequence length. Maximum sequence length (maximum number of token in some sentence in the dataset) was 199.

   It was decided to filter out sentences that had more than 10 words. But the maximum sequence length did not become 10, in fact it was 31. Sentences were preprocessed which might have increased the number of tokens. Let's see the wrong sentences:

```
<start> but he was tricked . . . . . captured . . . . . brought down to hell . . . . . corrupted . <end>
<start> but he was tricked , trapped , toppled into hell . . . corrupted . <end>
<start> they . . . killed her . . . and tried . . . to . . . feed her . . . to me ! <end>
<start> they killed him . . . and wanted . . . to . . . eat him ! <end>
<start> our military . . . strength . . . is . . . in this case . . . . . useless . <end>
<start> the military force is useless in this case . <end>
<start> you . . . . . will take nothing . . . . . from me . . . . . dwarf . <end>
<start> you won't take anything from me , dwarf . <end>
31 28
```

   So the problem was in preprocessing function which was fixed.

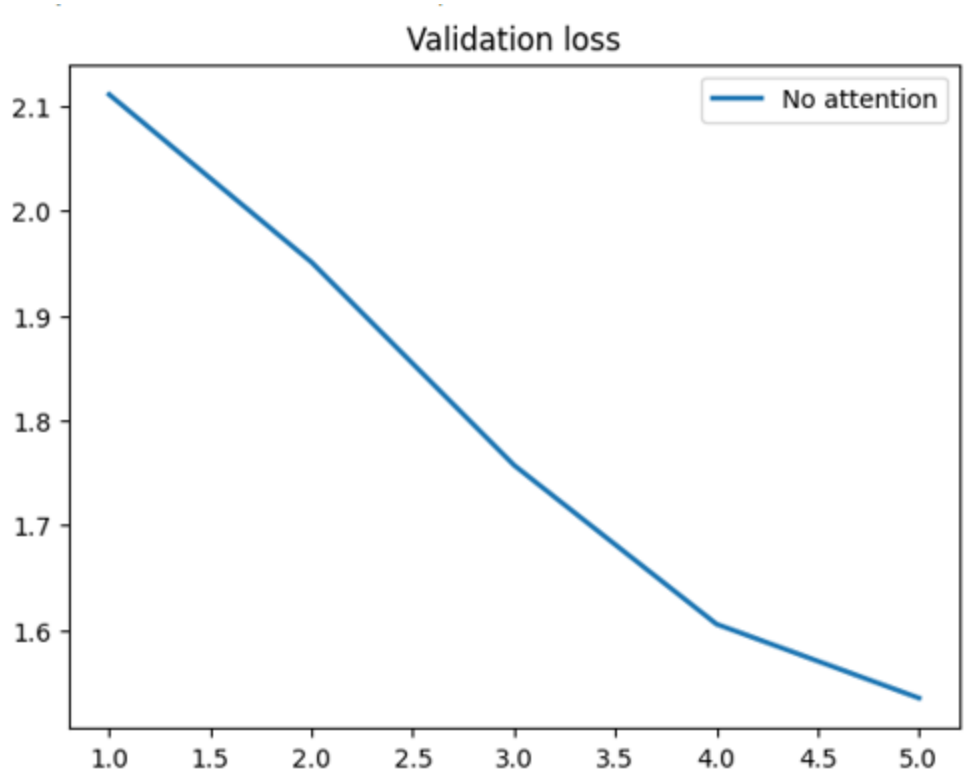
2. **Parameters changing**

   Now we we able to run 5 epochs on 30000 sentences.

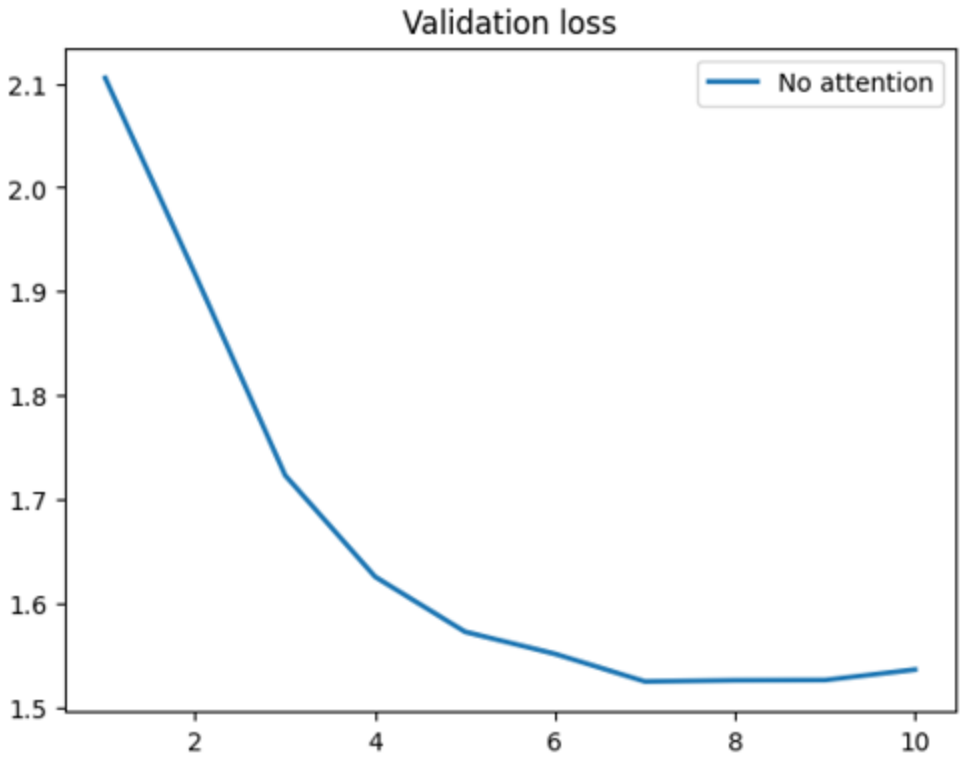
   Let's see the result:

```
print('Input: %s' % (sentence))
[29] print('Predicted translation: {}'.format(result))

Input: <start> fuck you <end>
Predicted translation: you are so annoying . <end>

[30] result, sentence = translate("get the hell outta here", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))

Input: <start> get the hell outta here <end>
Predicted translation: get off the fucking road . <end>

[32] result, sentence = translate("This house is fucking creepy", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))

Input: <start> this house is fucking creepy <end>
Predicted translation: this is ridiculous . <end>

[33] result, sentence = translate("I am mad at you", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))

Input: <start> i am mad at you <end>
Predicted translation: i mean you were a little man . <end>

result, sentence = translate("Don't be so fucking rude", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))

Input: <start> don't be so fucking rude <end>
Predicted translation: don't be silly . <end>
```

The loss graph:



Obviously there is a room for improve if we increase number of epochs. So let’s do this.

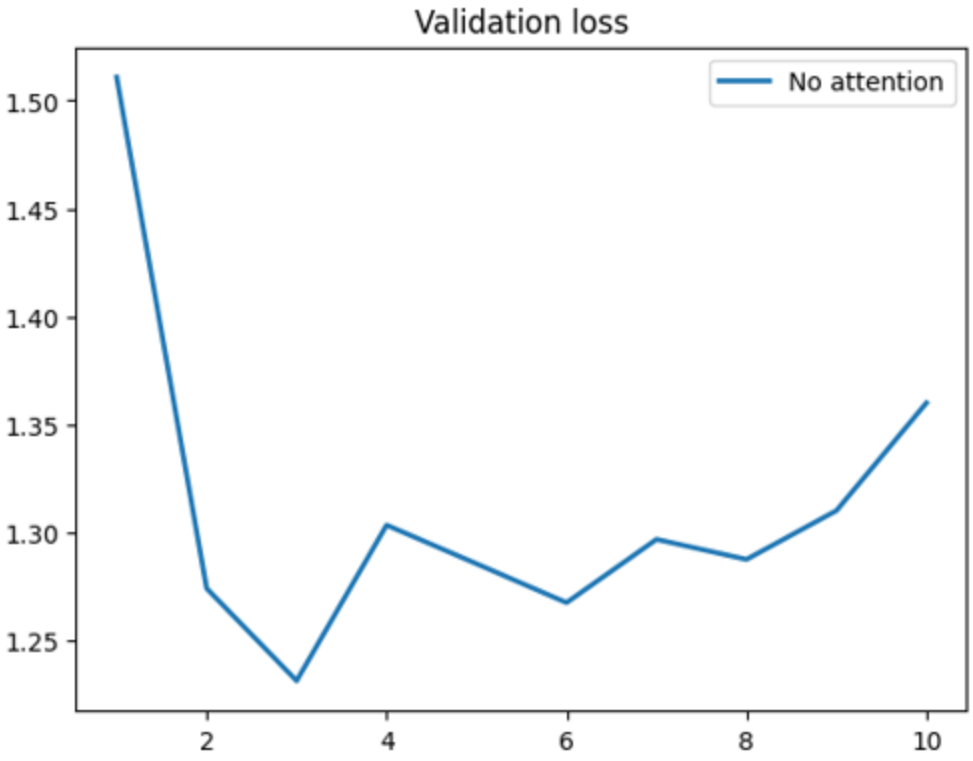


Okay, looks like this way we will not go below 1.5 validation loss.

By looking at the output it is easy to see that our model just do not know enough words and contexts in where to use them.

And as a general rule to avoid overfitting, we should add more data.

Now we increase our dataset to 100000 sentences. We do 10 epochs.



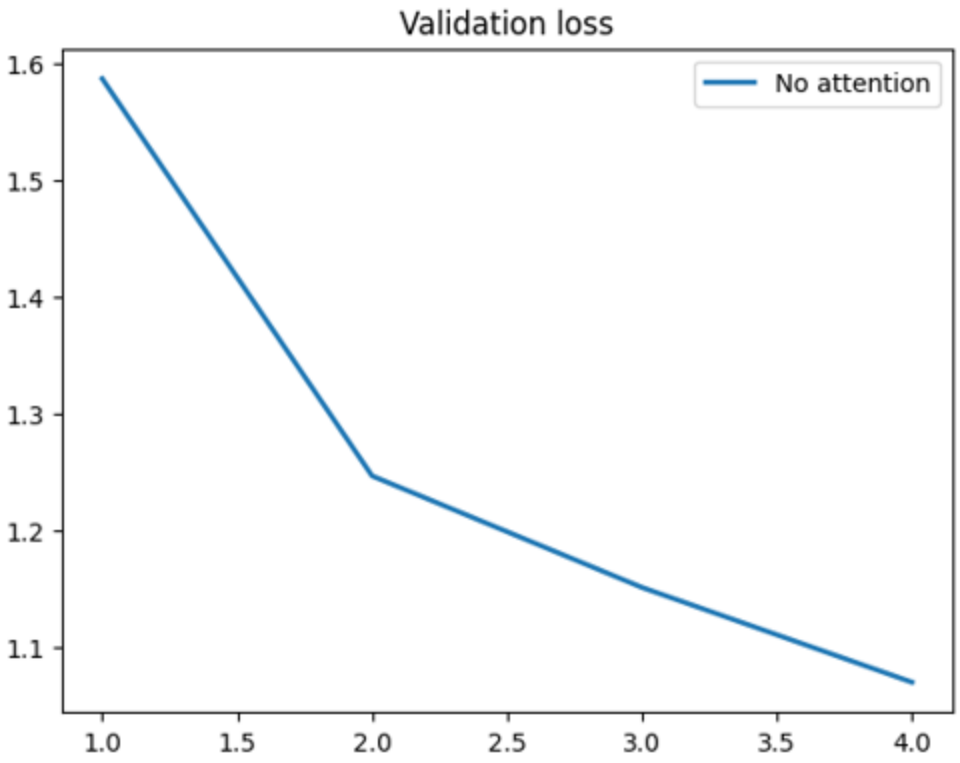
We could further increase our dataset but it become to take quite some time to wait it to train, instead let's try to improve the quality of the data.

3. **Improving quality of the data**

   We could notice that some of the sentences have tox score not close to 0 or 1. So we can filter out those pairs to leave only those that show difference more clearly.

   Also, we have similarity score. Sentences that are more similar to each other show difference more clearly. That might also help the model to understand what differs toxic sentence from not toxic one.

   Result:



Subjective quality of the result improved along with the validation loss:

```
    Stream
        Input: <start> get the hell outta here <end>
        Predicted translation: get out of here . <end>

Кодовая ячейка <-IPQYqUxwOEa>
# %% [code]
result, sentence = translate("This house is fucking creepy", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))
Дата и время получения выходных данных: 5 нояб. 2023 г. 16:44.
0KB
    Stream
        Input: <start> this house is fucking creepy <end>
        Predicted translation: this house is terrible . <end>

Кодовая ячейка <RJww0CGdwYrN>
# %% [code]
result, sentence = translate("I am mad at you", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))
Дата и время получения выходных данных: 5 нояб. 2023 г. 16:44.
0KB
    Stream
        Input: <start> i am mad at you <end>
        Predicted translation: i'm mad at you . <end>

Кодовая ячейка <R_yXbRpXweSg>
# %% [code]
result, sentence = translate("Don't be so fucking rude", encoder_dp, decoder_dp)
print('Input: %s' % (sentence))
print('Predicted translation: {}'.format(result))
Дата и время получения выходных данных: 5 нояб. 2023 г. 16:44.
0KB
    Stream
        Input: <start> don't be so fucking rude <end>
        Predicted translation: don't be so rude . <end>
```

4. **Increasing dataset even more**

   Now as we improved quality of the data we can improve quantity.

   As you could see by the graph above, we still didn't reach minimum loss on 100k sentences which means we add more epochs.

   But we still have spare 470k if we want to increase dataset.

   So we increase number of epochs and of sentences to 300k for the final solution described in report 2.