Machine Translation Sabrina Brändle, Eirini Valkana

**MT Exercise 4  
Layer Normalization for Transformer Models**

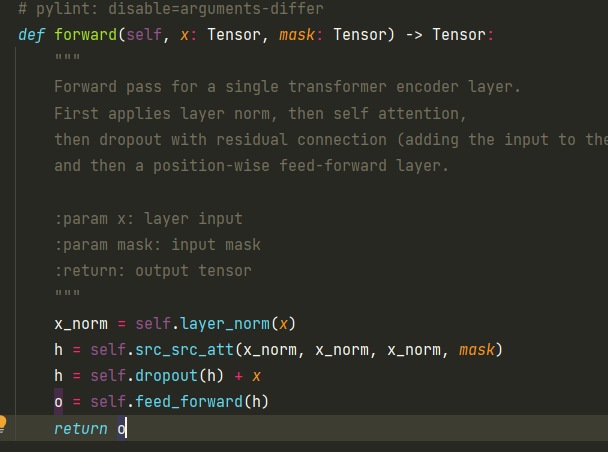
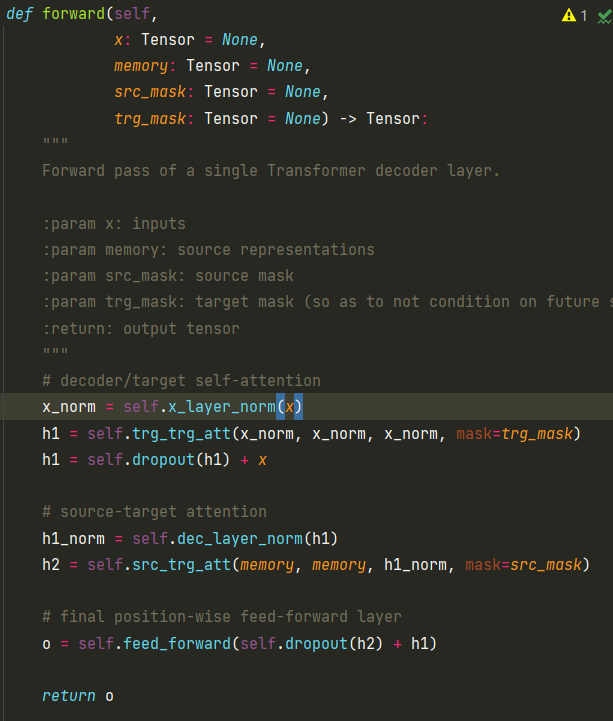
**Link to ‘ex4’ repository:** <https://github.com/sabrinabraendle/mt-exercise-4>

**Link to ‘joeynmt’ repository:** <https://github.com/sabrinabraendle/joeynmt>

**Instances of LayerNorm in JoeyNMT**

1. Python script: transformer\_layers.py

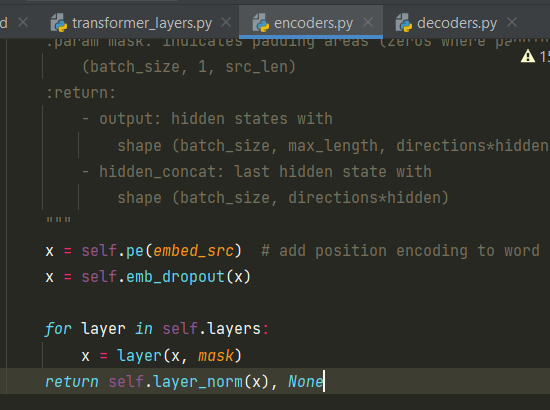
In this script, the layer normalization instance corresponds to pre-norm. The sublayers it modifies are the first one of the encoder and the first and second one of the decoder. The relevant code is the following:



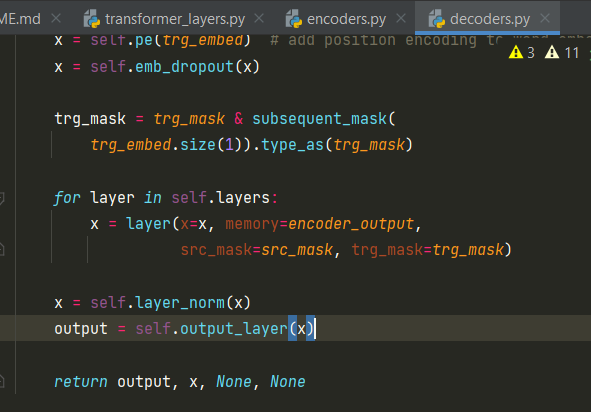
1. Python script: encoders.py

In this script, the layer normalization instance corresponds to post-norm. The sublayer it modifies is the last one of the encoder. The relevant code is the following:

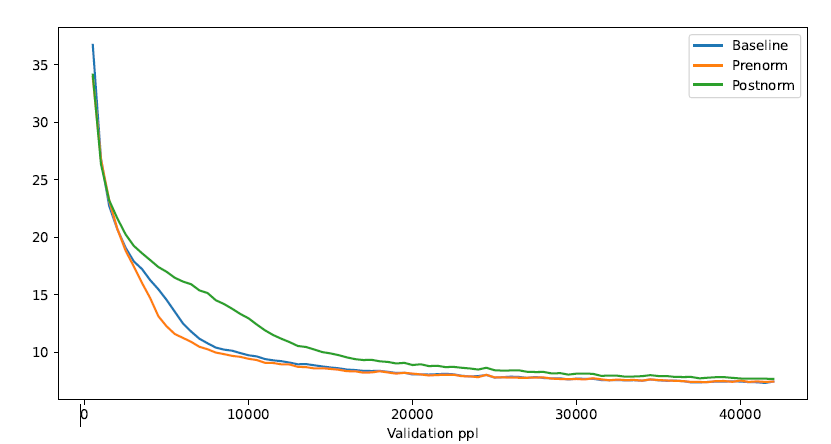


1. Python script: decoders.py

In this script, the layer normalization instance corresponds to post-norm as well. The sublayer it modifies is the last one of the decoder. The relevant code is the following:



**Discussion**



Discuss your results using the line chart. In your discussion, make sure to comment on the following points:

In the prenorm setting, the validation perplexity could be reduced slightly faster than the baseline around step 5000. But the perplexity can be reduced about more or less the same amount with further steps during the training. This makes sense as the pre-norm Transformer benefits more from the increase in encoder depth.

In the postnorm setting, a similarly better performance can be seen around step 500, while the perplexity values approach the ones of the baseline with further steps. It is possible that the performance is not significantly better because of the vanishing gradient problem.

Wang et al. also adapted some other hyperparameters according to the setting, i.e. prenorm or postnorm. It is therefore possible that the results in their experiments could be optimized as in our settings, the hyperparameters were kept the same. Additionally, the Transformer-Base model was updated for 100k steps, whereas our model was only trained during 42000 steps.

* Given that there is a difference in the training progress for the three models, can you think of a reason for it?
* In what way does our setup differ from Wang et. al. 2019? How could that have influenced our results?