Yunhao CHEN
yunhao.chen@polytechnique.edu

Lab session # 3
ALTEGRAD 2022

11/22/22

Question 1

We denote by $n_t = 30000$ the number of tokens; by $n_h = 512$ the hidden size of attention layer (the embedding dimension). The dimension of key, query and value is $n_k = n_v = n_q = 64$. Look at Fig. 1

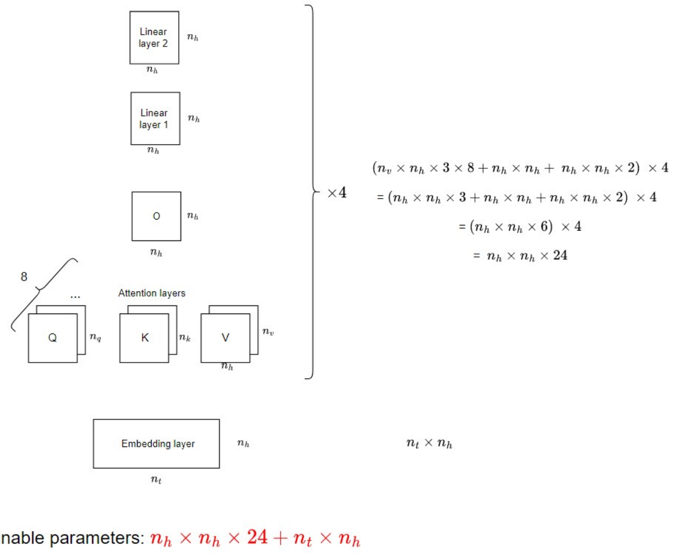
Linear layer 2
n_h
n_h
Linear layer 1
n_h
n_h
O
n_h
n_h
8
...
Attention layers
Q
n_q
K
n_k
V
n_v
n_h
$\times 4$
$(n_v \times n_h \times 3 \times 8 + n_h \times n_h + n_h \times n_h \times 2) \times 4$
$= (n_h \times n_h \times 3 + n_h \times n_h + n_h \times n_h \times 2) \times 4$
$= (n_h \times n_h \times 6) \times 4$
$= n_h \times n_h \times 24$
Embedding layer
n_h
n_t
$n_t \times n_h$
Trainable parameters: $n_h \times n_h \times 24 + n_t \times n_h$

Figure 1: Parameters in RoBERTa

If we omit the bias and normalization layers, the total trainable parameters in RoBERTa model equals to **22675456**; If we count the parameters in positional embedding layer, the total parameters is **22807552**. This result is verified by checking the model architecture in Pytorch.

Task 3 & 4

Finetune the pretrained model: for each seed, the checkpoint with best validation accuracy give respectively **82.0, 80.5, 83.5** as validation accuracy. On the test set, the accuracy is respectively **80.2, 79.5, 80.4**. The average test accuracy is **80.03**; and its standard deviation is **0.47**.

Finetune a random checkpoint: the result becomes much worse. For each seed, the checkpoint with best validation accuracy give respectively **62.0, 65.0, 63.0** as validation accuracy. On the test set, the accuracy is respectively **67.4, 68.1, 67.5**. The average test accuracy is **67.67**; and its standard deviation is **0.38**.

Question 2

Fairseq: need to perform tokenization and binarization.
Hugging face framework: need to convert your data to json format.
Personally, I prefer hugging face framework. Because it is better documented. Compared to HF, fairseq is poorly documented and sometimes we could not find the right parameters (for example, in 'train.py', there is no parameter 'warmup ratio', but only 'warmup-updates' indicating the number of warmup steps. However, the explanation about this point is unclear in the documentation.)