BARILLER Halvard
halvard.bariller@ens-paris-saclay.fr

Lab session # 4
ALTEGRAD 2023

11/14/23

1 Question 1

Compute the number of parameters of the model manually.

$RoBERT_a$ is based on the $BERT$ architecture [1], namely the Encoder part of the Transformer [3]. We will first remind the litteral expression of the encoder part, then proceed to the numerical computation.
For the base model in the Transformer, the trainable parameters (discarding the biases) are as follow:

- Embedding layer: $n_{tokens} \times n_{hid}$ (with n_{tokens} being the size of the vocabulary and n_{hid} the dimension of the embedding).
- Positional Encoding layer: In BERT the positional encoding is learned, resulting in $n_{hid} \times length\_seq_{max}$ trainable parameters.
- Self-Attention layer: $n_k \times n_{hid}$ for the Query, $n_k \times n_{hid}$ for the Keys and $n_v \times n_{hid}$ for the values (the weights are the same for each token). Here, $n_k = n_v = n_{hid}/n_{head}$.
- Multi-Head Attention: $n_{head} \times w_{sal} + n_{hid} \times n_{hid}$ (with w_{sal} being the number of parameters in one Self-Attention layer, the second term corresponding to a projection after concatenation of the self-heads).
- Feed-forward layer: $n_{in} \times n_{ff} + n_{ff} \times n_{out}$. Here, $n_{in} = n_{ff} = n_{out} = n_{hid}$.

The total number of trainable parameters is:

$$\begin{aligned} P &= n_{token} \times n_{hid} + n_{hid} \times length\_seq_{max} + n_{layers} \times \left(3 \times n_{hid}^2 + n_{hid}^2 + 2 \times n_{hid}^2\right) \\ &= n_{token} \times n_{hid} + n_{hid} \times length\_seq_{max} + 6 \times N \times n_{hid}^2. \end{aligned}$$

In our case, the numerical values are as follows:

- Vocabulary size: $n_{token} = 32000$
- Sequence max: $length\_seq_{max} = 256$
- Hidden dimension: $n_{hid} = 512$
- Encoder layers: $n_{layers} = 4$

$$P = 32000 \times 512 + 512 \times 256 + 6 \times 4 \times 512^2 = 22,806,528.$$

The number of parameters counted in the notebook is of the same order range $22,829,056$, the small difference probably being due to the fact that not all biases and normalization weights were removed in the notebook (resulting in a slightly higher number).

2 Question 2

Define each parameter used in LoraConfig.

LoRA (Low-Rank Adaptation) [2] is an approach presented to address challenges in adapting large-scale, pre-trained language models for various downstream tasks. The conventional fine-tuning process, called *full fine-tuning* since it updates all model parameters, becomes impractical as models grow in size, leading to storage and deployment challenges. LoRA introduces a novel fine-tuning method (*Parameter Efficient Fine-Tuning*) by freezing pre-trained model weights and incorporating trainable rank decomposition matrices into each layer of the Transformer architecture. LoRA exploits the empirical observation that weight alterations during fine-tuning exhibit a low "intrinsic rank." This insight leads to a unique low-rank adaptation approach where specific dense layers are trained indirectly via the optimization of rank decomposition matrices, all while maintaining the pre-trained weights as constants. In practical terms, this implies that a smaller set of task-specific parameters, represented by a low-rank matrix, suffices to capture the necessary adaptations. This results in a drastic reduction in trainable parameters for downstream tasks as well as GPU memory

requirements.
The method is task agnostic and requires an extremely light fine-tuning process as compared to full fine-tuning, allowing the creation of task-specific LoRA modules, reducing storage requirements, and enabling efficient task-switching.

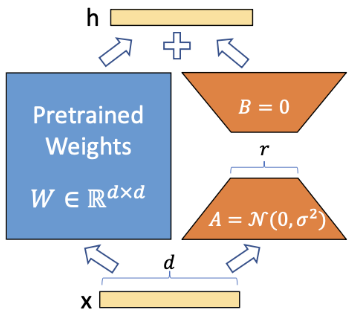

h
Pretrained Weights
$W \in \mathbb{R}^{d\times d}$
$B = 0$
r
$A = \mathcal{N}(0, \sigma^2)$
d
x

Figure 1: Representation of a dense layer with low-rank decomposition matrices (only A and B are trained) [2].

In our case, we used LoRA to fine-tune *BLOOM*-$560M$ on a set of customer Q&A's with the following configuration:

```
config = LoraConfig(
r=16,
lora_alpha=32,
target_modules=["query_key_value"],
lora_dropout=0.05,
bias="none",
task_type="CAUSAL_LM"
)
```

where we have that:

- `r` is the rank of the decomposition matrices (c.f. figure 1),
- `lora_alpha` is used to scale the decomposition matrices by a factor of $\frac{\alpha}{r}$ (at first considered as an hyperparameter, but the original paper concludes that when optimizing with Adam, tuning α is roughly the same as tuning the learning rate if the initialization is scaled appropriately),
- `target_module` are the weights' matrices that are adapted using LoRA (here the Attention weights),
- `lora_dropout` is the dropout probability of the LoRA layers,
- `bias` defines which biases are going to be updated during training (can be `none`, `all` or `lora_only`),
- `task_type` is the type of task to perform (can be one of `SEQ_CLS, SEQ_2_SEQ_LM, CAUSAL_LM, TOKEN_CLS, QUESTION_ANS, FEATURE_EXTRACTION`).

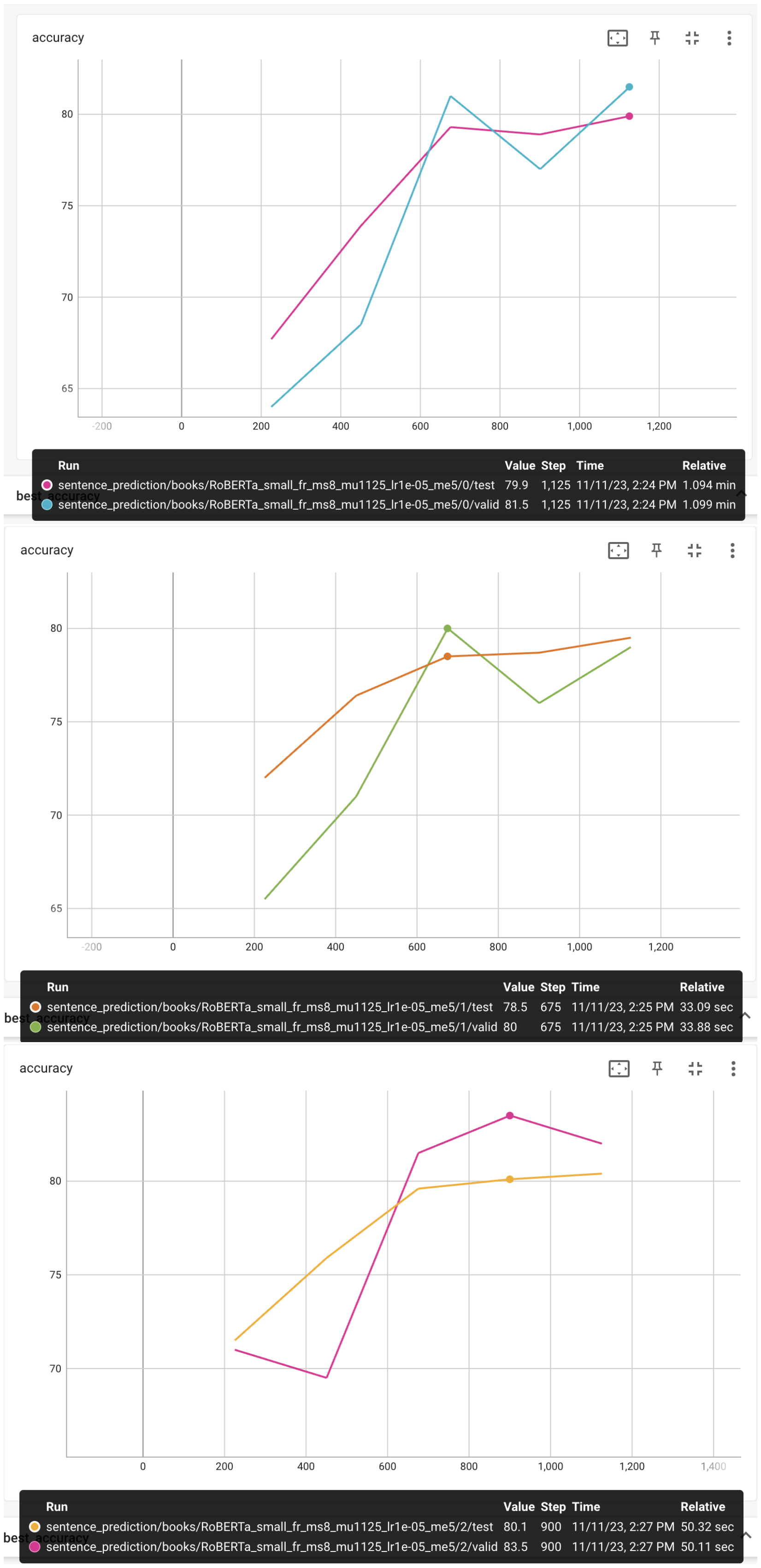
3 Model's comparison

Table 1: Comparison of the accuracies over the test set for different versions of $RoBERT_a$

		Initializations		
Model	Accuracy	Seed 1	Seed 2	Seed 3
Pretrained $RoBERT_a$ (Fairseq)	79.5 ± 0.71	79.9	78.5	80.1
Random checkpoint $RoBERT_a$ (Fairseq)	66.5 ± 0.83	65.4	66.7	67.4
Pretrained $RoBERT_a$ (HuggingFace)	80.22 ± 0.12	80.05	80.3	80.3

For each initialisation, the checkpoint yielding the best validation accuracy was chosen. The considered accuracies are then obtained on the test set.
See the Tasks 3, 4 and 5 in the next pages.

4 Task 3 - Finetuning pretrained $RoBERT_a$ with Fairseq



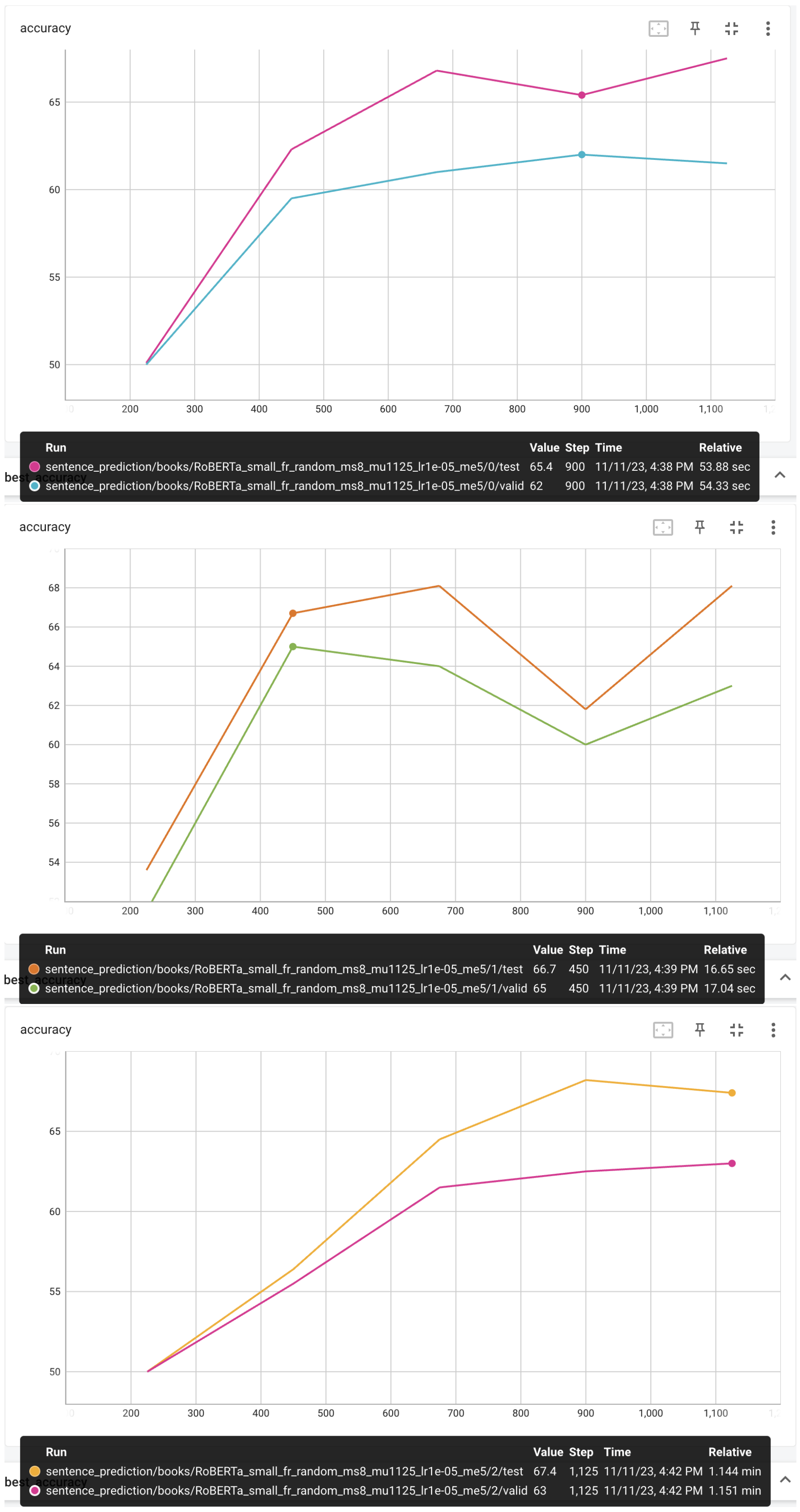
	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/0/test	79.9	1,125	11/11/23, 2:24 PM	1.094 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/0/valid	81.5	1,125	11/11/23, 2:24 PM	1.099 min

	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/1/test	78.5	675	11/11/23, 2:25 PM	33.09 sec
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/1/valid	80	675	11/11/23, 2:25 PM	33.88 sec

	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/2/test	80.1	900	11/11/23, 2:27 PM	50.32 sec
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/2/valid	83.5	900	11/11/23, 2:27 PM	50.11 sec

Figure 2: Accuracy scores after finetuning a pretrained $RoBERT_a$ with different initialisations

5 Task 3 - Finetuning $RoBERT_a$ from a random checkpoint with Fairseq



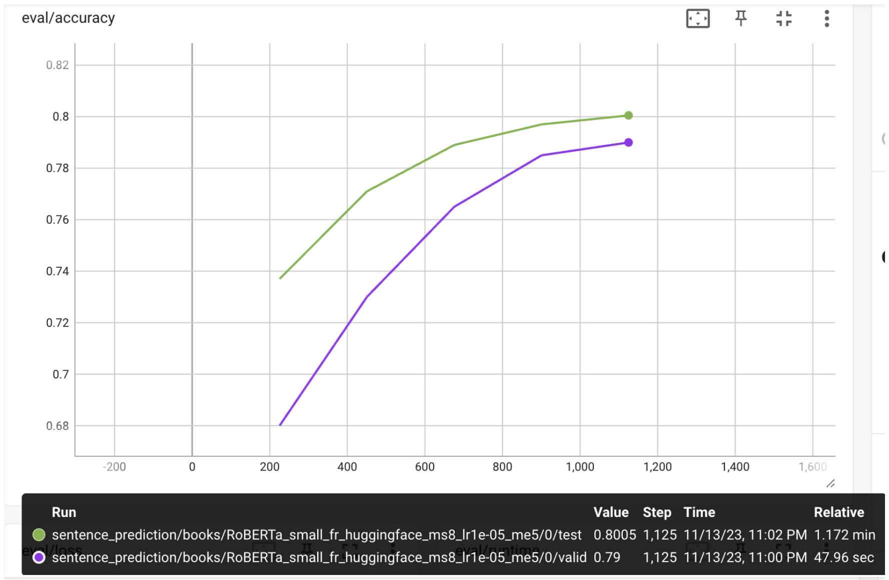
	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/0/test	65.4	900	11/11/23, 4:38 PM	53.88 sec
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/0/valid	62	900	11/11/23, 4:38 PM	54.33 sec

	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/1/test	66.7	450	11/11/23, 4:39 PM	16.65 sec
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/1/valid	65	450	11/11/23, 4:39 PM	17.04 sec

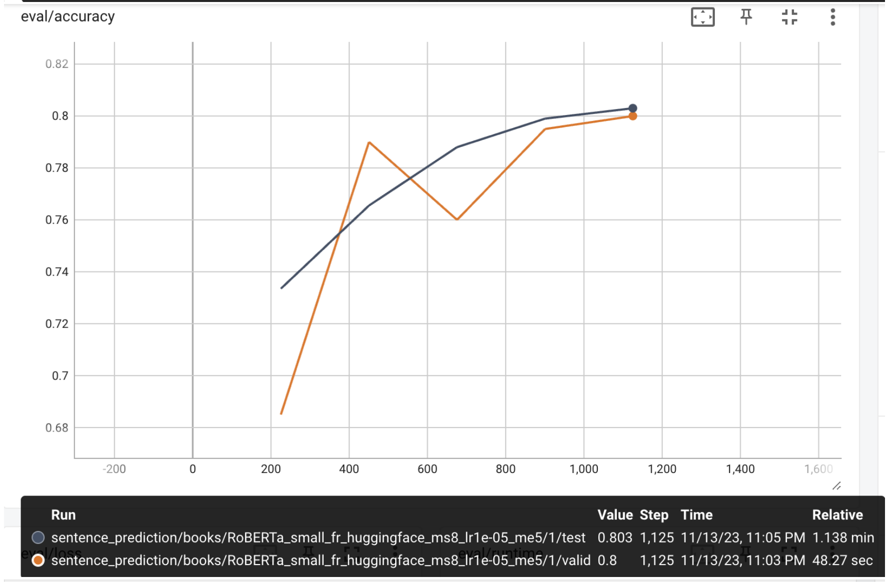
	Run	Value	Step	Time	Relative
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/2/test	67.4	1,125	11/11/23, 4:42 PM	1.144 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/2/valid	63	1,125	11/11/23, 4:42 PM	1.151 min

Figure 3: Accuracy scores after finetuning a random checkpoint of $RoBERT_a$ with different initialisations

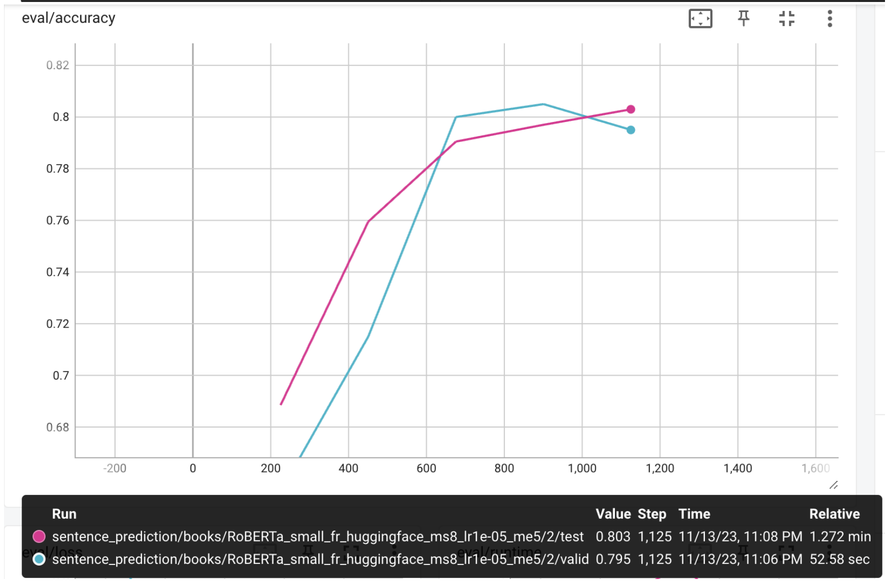
6 Task 5 - Finetuning pretrained $RoBERT_a$ with HuggingFace



Run	Value	Step	Time	Relative
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/0/test	0.8005	1,125	11/13/23, 11:02 PM	1.172 min
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/0/valid	0.79	1,125	11/13/23, 11:00 PM	47.96 sec



Run	Value	Step	Time	Relative
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/1/test	0.803	1,125	11/13/23, 11:05 PM	1.138 min
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/1/valid	0.8	1,125	11/13/23, 11:03 PM	48.27 sec

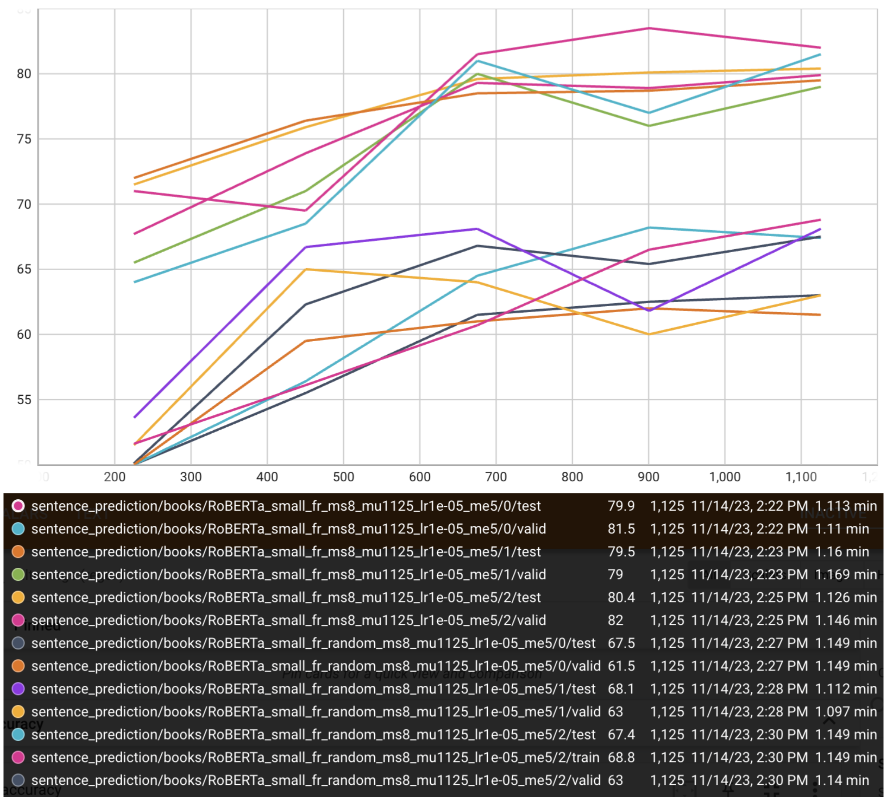


Run	Value	Step	Time	Relative
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/2/test	0.803	1,125	11/13/23, 11:08 PM	1.272 min
● sentence_prediction/books/RoBERTa_small_fr_huggingface_ms8_lr1e-05_me5/2/valid	0.795	1,125	11/13/23, 11:06 PM	52.58 sec

Figure 4: Accuracy scores after finetuning a pretrained $RoBERT_a$ with different initialisations

7 Comparison of Roberta's initialization using fairseq

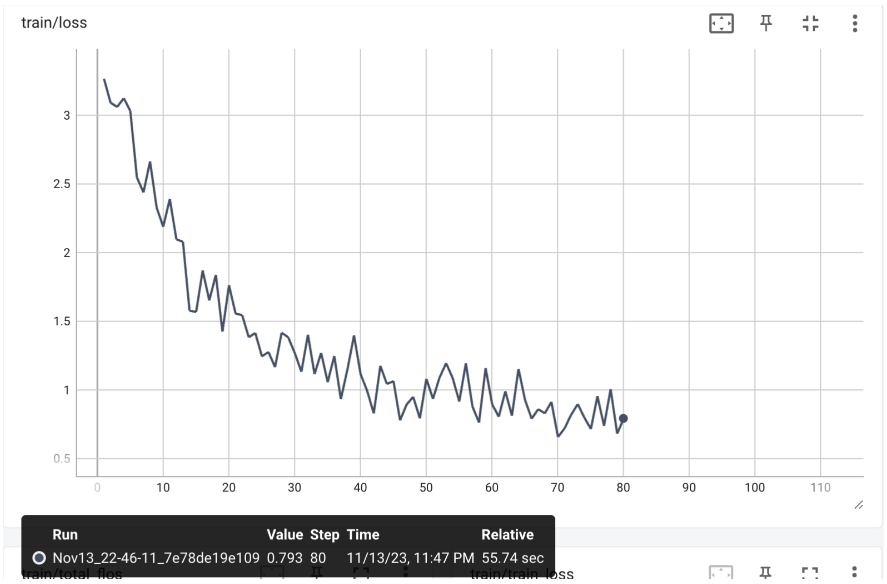
A clear representation of the results gathered in table 1 is available in figure 4, showing distinctly the gap in performance resulting from a pretrained model.



●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/0/test	79.9	1,125	11/14/23, 2:22 PM	1.113 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/0/valid	81.5	1,125	11/14/23, 2:22 PM	1.11 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/1/test	79.5	1,125	11/14/23, 2:23 PM	1.16 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/1/valid	79	1,125	11/14/23, 2:23 PM	1.169 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/2/test	80.4	1,125	11/14/23, 2:25 PM	1.126 min
●	sentence_prediction/books/RoBERTa_small_fr_ms8_mu1125_lr1e-05_me5/2/valid	82	1,125	11/14/23, 2:25 PM	1.146 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/0/test	67.5	1,125	11/14/23, 2:27 PM	1.149 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/0/valid	61.5	1,125	11/14/23, 2:27 PM	1.149 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/1/test	68.1	1,125	11/14/23, 2:28 PM	1.112 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/1/valid	63	1,125	11/14/23, 2:28 PM	1.097 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/2/test	67.4	1,125	11/14/23, 2:30 PM	1.149 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/2/train	68.8	1,125	11/14/23, 2:30 PM	1.149 min
●	sentence_prediction/books/RoBERTa_small_fr_random_ms8_mu1125_lr1e-05_me5/2/valid	63	1,125	11/14/23, 2:30 PM	1.14 min

Figure 5: Accuracy scores after finetuning $RoBERT_a$ with different initialisations

8 BLOOM's fine-tuning



	Run	Value	Step	Time	Relative
○	Nov13_22-46-11_7e78de19e109	0.793	80	11/13/23, 11:47 PM	55.74 sec

Figure 6: Loss during training of BLOOM

References

[1] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. arxiv:1810.04805.

[2] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.