Q1: Data processing

1. tokenizer : 主要對詞做切割，讓分詞器能取得有意義的文字，但因為會出現一些沒看過的詞，不像英文一樣以單字作為單位，所以採選擇最大機率的詞。並會在先前定義每個單字的大小去做選擇，在我們建立LM時可以選擇最大機率的詞。

2. Answer Span :
    a. 使用huggingface的範例程式碼「run_qa_no_trainer.py」，透過offsets可以得到每個token的start、end的位置，然後找出與span start、span end相同的位置，即為start_postions、end_postions。

    b. 對每種start_postions、end_postions做機率統計，選出最大機率的詞，最後再用offset對應回去，即為最後選擇的結果。

```python
# Start/end character index of the answer in the text.
start_char = answers["start"]
end_char = start_char + len(answers["text"])

# Start token index of the current span in the text.
token_start_index = 0
while sequence_ids[token_start_index] != (1 if pad_on_right else 0):
    token_start_index += 1

# End token index of the current span in the text.

token_end_index = len(input_ids) - 1
while sequence_ids[token_end_index] != (1 if pad_on_right else 0):
    token_end_index -= 1

# Detect if the answer is out of the span (in which case this feature is labeled with the CLS index).
if not (offsets[token_start_index][0] <= start_char and offsets[token_end_index][1] >= end_char):
    tokenized_example["start_positions"].append(cls_index)
    tokenized_example["end_positions"].append(cls_index)
else:
    # Otherwise move the token_start_index and token_end_index to the two ends of the answer.
    # Note: we could go after the last offset if the answer is the last word (edge case).
    while token_start_index < len(offsets) and offsets[token_start_index][0] <= start_char:
        token_start_index += 1
    tokenized_example["start_positions"].append(token_start_index - 1)
    while offsets[token_end_index][1] >= end_char:
        token_end_index -= 1
    tokenized_example["end_positions"].append(token_end_index + 1)
```

Q2: Modeling with BERTs and their variants

1.
    a. model: bert-base-chinese
    b. performance: 0.75316

question_answering.csv Complete · 2d ago	**0.76513**	**0.75316**

    c. loss function: torch.nn.CrossEntropyLoss()
    d. optimization algorithm: torch.optim.AdamW()

learning rate: 3e-5, batch size: 1
epoch: mutiple_choice=1 qustion_answering=2

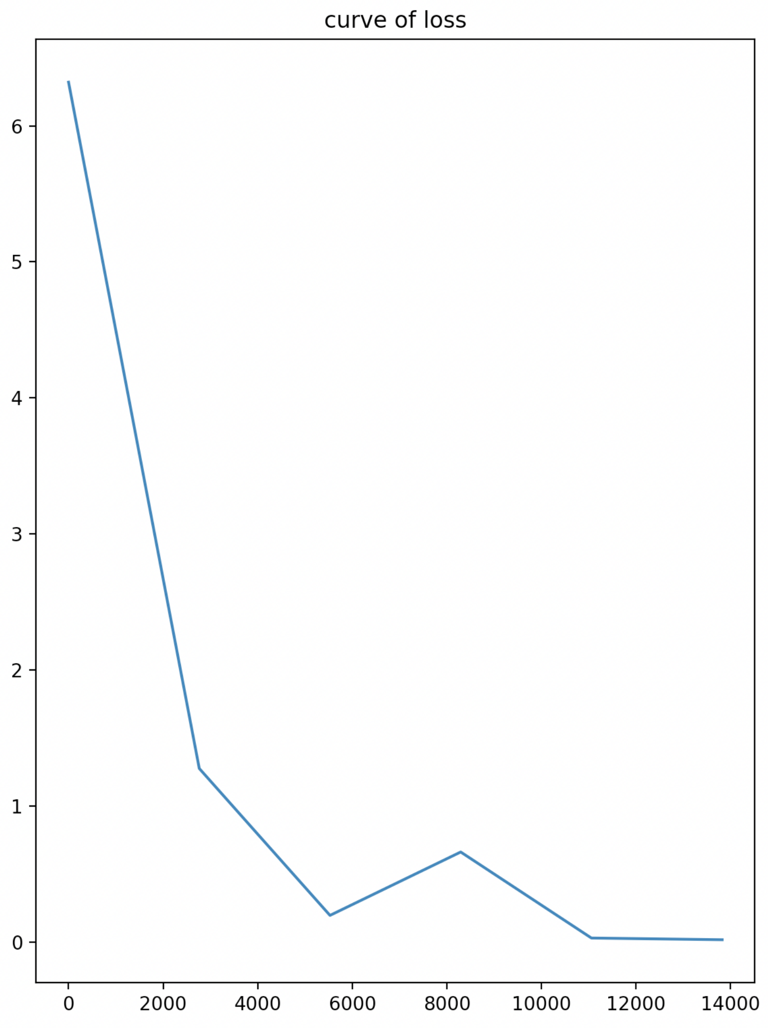
2.
   a. model: hfl/chinese-roberta-wwm-ext-large
   b. performance: 0.79385

question_answering.csv Complete (after deadline) · 1d ago	**0.80307**	**0.79385**

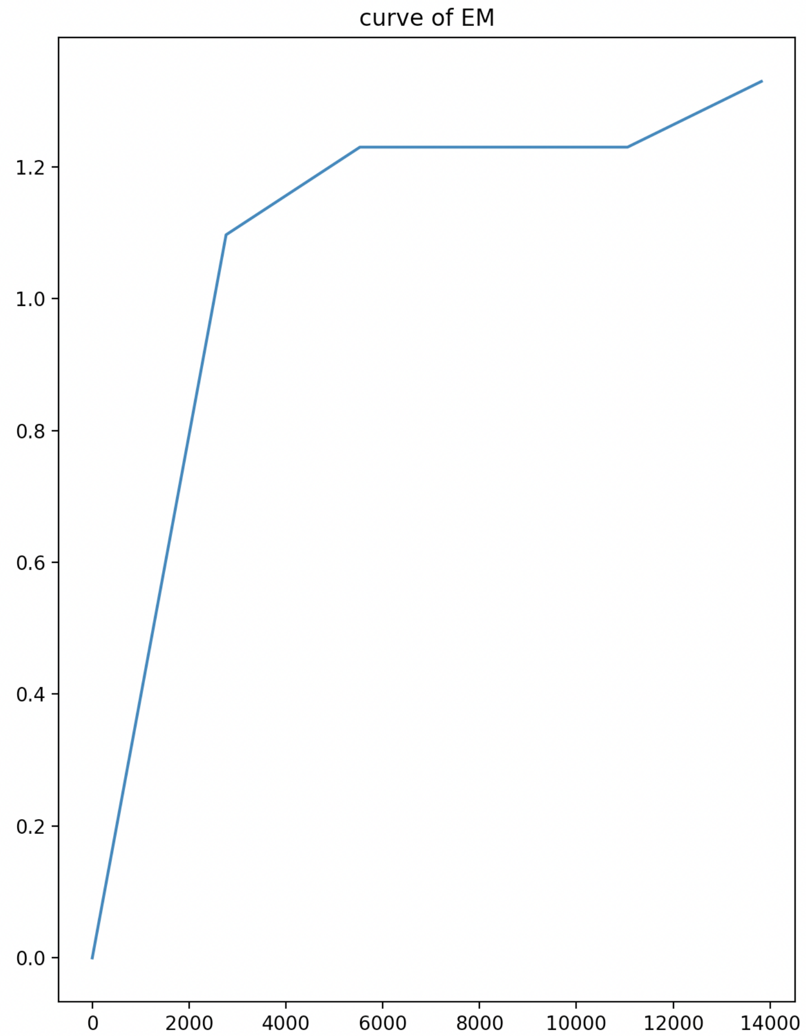
   c. loss function: torch.nn.CrossEntropyLoss()
   d. optimization algorithm: torch.optim.AdamW()
      learning rate: 3e-5, batch size: 1
      epoch: mutiple_choice=1, qustion_answering=2

Q3: Curves

a. Learning curve of loss (epoch : 1)



b. Learning curve of EM (epoch : 1)



Q4: Pretrained vs Not Pretrained

a. describe:變更question的訓練方式，基本上就是將其預訓練的權重去除掉，所以在僅僅訓練少少epoch時，沒辦法達到跟已經預訓練過的模型一樣，他的performance會極低。有可能訓練很多個epoch或是給予較多資料訓練，就會有比較好的performance。
b. model: bert-base-chinese
c. performance: 0.0018

question_answering-2.csv Complete (after deadline) · now	**0.0009**	**0.0018**

d. loss function: torch.nn.CrossEntropyLoss()
e. optimization algorithm: torch.optim.AdamW()
learning rate: 3e-5, batch size: 1
epoch: qustion_answering=2