ADL HW2

R11922196 林佑鑫

Q1: Data processing

1. Tokenizer:

a. Describe in detail about the tokenization algorithm you use. You need to explain what it does in your own ways.

用 wordpiece, 把 word 切成 subword，能簡化同一單字的各種不同後綴，如時態、被動等等，減少詞表數量。

Step1: 將訓練資料的所有 word 切成最小單位，並確定所需詞表大小

Step2: 用 1.的資料建立詞表

Step3: 選擇詞表中最相鄰的兩個 word 合併後加入詞表

Step4: 重複 Step3 直到詞表大小到達需求。

2. Answer Span:

a. How did you convert the answer span start/end position on characters to position on tokens after BERT tokenization?

Tokenizer 用 return_offset_mapping 會回傳 (char start,char end)，iterate 找出 span start, char start; span end, char end 相同的位置就是 start, end position。

b. After your model predicts the probability of answer span start/end position, what rules did you apply to determine the final start/end position?

先做 postprocessing 去除不可能的答案(e.g. end position > start position 或是 subsentence 比 sentence 長，之後 iterate 所有機率，選出最高的(start, end)就是 final start/end position。

Q2: Modeling with BERTs and their variants

*這邊是我寫報告當下做的 model，可能與 kaggle 上結果最好的 model 不同。

1. Describe

a. your model (configuration of the transformer model)

bert-base-chinese (左：MultipleChoice，右：QuestionAnswering)

```
{
  "_name_or_path": "bert-base-chinese",
  "architectures": [
    "BertForMultipleChoice"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "directionality": "bidi",
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 21128
}
```

```
{
  "_name_or_path": "bert-base-chinese",
  "architectures": [
    "BertForQuestionAnswering"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "directionality": "bidi",
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 21128
}
```

b. performance of your model.

MultipleChoice_eval: **0.9594549684280492**

QuestionAnswering_eval_EM: **79.26221335992024**

QuestionAnswering_eval_f1: **79.26221335992024**

Public result: **0.75316**

c. the loss function you used.

都是 CrossEntropyLoss

d. The optimization algorithm (e.g. Adam), learning rate and batch size.

MultipleChoice:

- Optimizer: AdamW
- Learning rate: 3e-5
- Batch size: 1
- Gradient accumulation: 2

QuestionAnswering:

Optimizer: AdamW

Learning rate: 3e-5

Batch size: 1

Gradient accumulation: 2

2.

Try another type of pretrained model and describe

a. your model

hfl/chinese-roberta-wwm-ext (左：MultipleChoice，右：QuestionAnswering)

```
{
  "_name_or_path": "hfl/chinese-roberta-wwm-ext",
  "architectures": [
    "BertForMultipleChoice"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "directionality": "bidi",
  "eos_token_id": 2,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "output_past": true,
  "pad_token_id": 0,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 21128
}
```

```
{
  "_name_or_path": "hfl/chinese-roberta-wwm-ext",
  "architectures": [
    "BertForQuestionAnswering"
  ],
  "attention_probs_dropout_prob": 0.1,
  "bos_token_id": 0,
  "classifier_dropout": null,
  "directionality": "bidi",
  "eos_token_id": 2,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "output_past": true,
  "pad_token_id": 0,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 21128
}
```

b. performance of your model

MultipleChoice_eval: **0. 9601196410767697**

QuestionAnswering_eval_EM: **82. 31970754403456**

QuestionAnswering_eval_f1: **82. 31970754403456**

Public result: **0.78481**

c. the difference between pretrained model (architecture, pretraining loss, etc.)

architecture: 多了 dynamic masking，在 training 時會改變 mask 的位置。

pretraining: 比 BERT 用更大量資料訓練。

loss: RoBERTa loss 比 BERT 更小，表現更好。

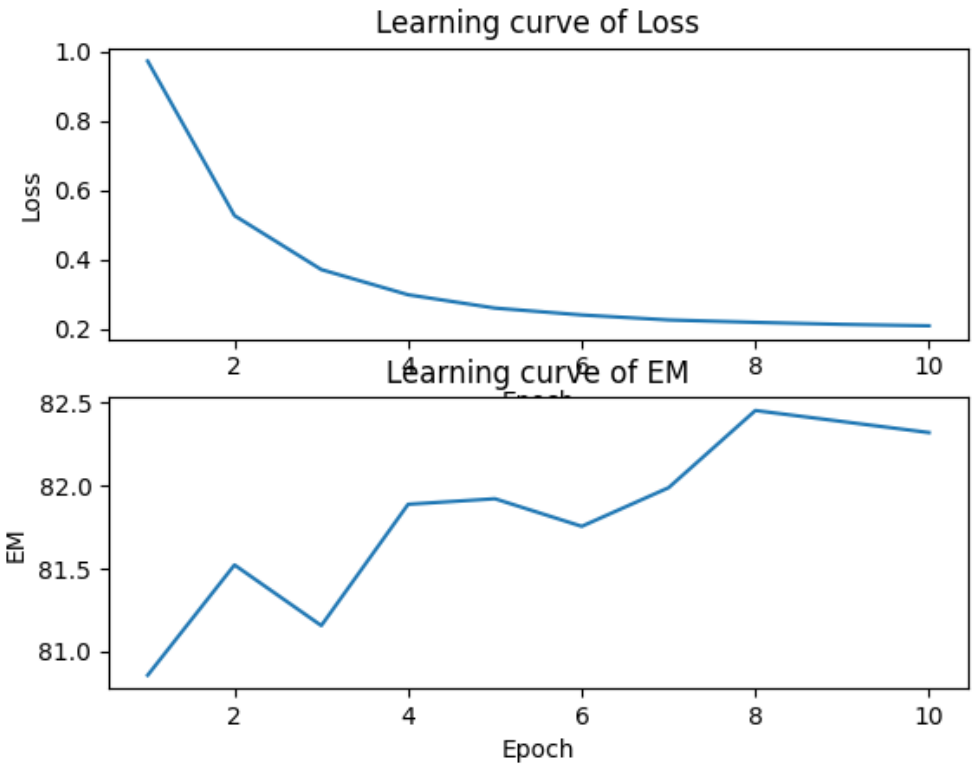
Q3: Curves

1. Plot the learning curve of your QA model

a. Learning curve of loss

b. Learning curve of EM

註：這是 Q2: hfl/chinese-roberta-wwm-ext 的 Learning curve



可以看到理想的 epoch 數應該是 8 個，再加大的話 EM 反而會變小，故我後面有參考此現象來做 training。

Q4: Pretrained vs Not Pretrained

1. The configuration of the model and how do you train this model

我將 Q1 的 bert-base-chinese 的 Question Answering 改成 not pretrained model，Multiple Question 部分與 Q1 相同（下圖爲 QA 的 config）


```
{
  "_name_or_path": "bert-base-chinese",
  "architectures": [
    "BertForQuestionAnswering"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "directionality": "bidi",
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "pooler_fc_size": 768,
  "pooler_num_attention_heads": 12,
  "pooler_num_fc_layers": 3,
  "pooler_size_per_head": 128,
  "pooler_type": "first_token_transform",
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 21128
}
```


2. the performance of this model v.s. BERT

MultipleChoice_eval: **0. 9594549684280492**

QuestionAnswering_eval_EM: **6.613492854769026**

QuestionAnswering_eval_f1: **6.613492854769026**

Public result: **0.07414**

Question Answering 部分完全不能做爲正常的 model 使用，只有 7.4%準確率，對比 bert-base-chinese 的 pretrained model 近 75%的準確率，可得知 pretrain model 的訓練過資料量很龐大，已經相對完整。若從零開始 train 我們的資料、model 並沒辦法訓練完全，只能產生很低的準確率。

Q5: Bonus: HW1 with BERTs

a. your model

bert-base-uncased

(左：Intent Classification，我用 Sequence Classification)

(右：Slot tagging，我用 Token Classification)


```
{
  "_name_or_path": "bert-base-uncased",
  "architectures": [
    "BertForSequenceClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "accept_reservations",
    "1": "account_blocked",
    "2": "alarm",
    "3": "application_status",
    "4": "apr",
    "5": "are_you_a_bot",
    "6": "balance",
    "7": "bill_balance",
    "8": "bill_due",
    "9": "book_flight",
    "10": "book_hotel",
    "11": "calculator",
    "12": "calendar",
    "13": "calendar_update",
    "14": "calories",
    "15": "cancel",
    "140": "weather",
    "141": "what_are_your_hobbies",
    "142": "what_can_i_ask_you",
    "143": "what_is_your_name",
    "144": "what_song",
    "145": "where_are_you_from",
    "146": "whisper_mode",
    "147": "who_do_you_work_for",
    "148": "who_made_you",
    "149": "yes"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "problem_type": "single_label_classification",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}
```

```
{
  "_name_or_path": "bert-base-uncased",
  "architectures": [
    "BertForTokenClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "B-date",
    "1": "B-first_name",
    "2": "B-last_name",
    "3": "B-people",
    "4": "B-time",
    "5": "I-date",
    "6": "I-people",
    "7": "I-time",
    "8": "O"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "B-date": 0,
    "B-first_name": 1,
    "B-last_name": 2,
    "B-people": 3,
    "B-time": 4,
    "I-date": 5,
    "I-people": 6,
    "I-time": 7,
    "O": 8
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "torch_dtype": "float32",
  "transformers_version": "4.22.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}
```


註： 第一題 label2id 很長、且放上來沒有意義，所以我只截圖有用部分。

b. performance of your model.
   Intent Classification: Public: **0.948**, Private: **0.95155**
   Slot tagging: Public: **0.80536**, Private: **0.81939**
   都 train 8 個 epoch 就遠超過 HW1 時的成績(原本是 Intent: 0.92, Slot: 0.78)可見 BERT 是較強大的架構，達到的結果能比傳統的 RNN 更加精準。
c. the loss function you used.
   都是預設的 CrossEntropyLoss
d. The optimization algorithm (e.g. Adam), learning rate and batch size.
   Intent Classification:
      optimizer: AdamW
      lr: 3e-5
      weight decay: 1e-2
      batch size: 32
      gradient accumulation: 2
      effective batch size: 64

   Slot tagging:
      optimizer: AdamW
      lr: 3e-5
      weight decay: 1e-2
      batch size: 32
      gradient accumulation: 2
      effective batch size: 64