ADL HW3 Report　R12944062 李泓賢

前言：

這次的作業我沒有過 baseline，而且最後繳交的那份 model 效果還是最差的，接下來會詳細說明。在 11/29 之前我在 train 的時候不知道需要將 prompt 加進 training data 做訓練，我以為 prompt 加進模型中會使整個 data 的維度增加，且每個 batch 中 duplicate data 會太多而導致模型效果變差，故我當時認為 get_prompt 函數只在 inference 作使用即可。然而在 11/29 之前我調了非常多次參數組合，ppl 最好的某一次有 4.3，然而直到 Deadline 之時，我都還沒有找到一個可以過 baseline 的參數組合。於是我問了朋友我有什麼地方可以改進，他剛好跟我提到他有將 prompt 加進 training data 中，此外，Taiwan LLaMa 在 pretrain 之時是 instruction-based 的訓練方式，因此 Taiwan LLaMa 是看得懂 prompt 的。於是我在 11/30 這天才嘗試將 prompt 加入到 training data 之中，然而最後 ppl 才 4.81，考量到每遲交一天會打折一次，且我已經花非常多時間在調參上了，因此我最終決定將這個結果(4.81)繳交。

Q1：

(A)Describe

我使用了 3000 筆的 training data

```
#load dataset
data_file = {}
if args.train_file is not None:
    data_file = args.train_file
extension = args.train_file.split(".")[-1]
dataset = load_dataset(extension, data_files=data_file)
dataset['train'] = dataset['train'].select(range(3000))
```

使用 QLora 作 fine-tuning，QLora 是一種 PEFT 的方式，不需要去 fine-tune 原本 Taiwan LLaMa 所有的 weights，而是 fine-tune 一個維度更小的 weight matrix。此外，QLora 與 Lora 的最大差別是，他是使用 4-bit 的 weights 而不是 8-bit，因此可以達到更好的 memory-efficient 效果。

QLora config 如下頁的圖:

```
{
  "alpha_pattern": {},
  "auto_mapping": null,
  "base_model_name_or_path": "./Taiwan-LLM-7B-v2.0-chat",
  "bias": "none",
  "fan_in_fan_out": false,
  "inference_mode": true,
  "init_lora_weights": true,
  "layers_pattern": null,
  "layers_to_transform": null,
  "lora_alpha": 16,
  "lora_dropout": 0.05,
  "modules_to_save": null,
  "peft_type": "LORA",
  "r": 4,
  "rank_pattern": {},
  "revision": null,
  "target_modules": [
    "q_proj",
    "v_proj"
  ],
  "task_type": "CAUSAL_LM"
}
```


此外，BitsAndBytesConfig 如下圖:

```python
def get_bnb_config() -> BitsAndBytesConfig:
    '''Get the BitsAndBytesConfig.'''
    config = BitsAndBytesConfig(
                load_in_4bit=True,
                bnb_4bit_quant_type="nf4",
                bnb_4bit_compute_dtype=torch.bfloat16,
                bnb_4bit_use_double_quant=True,
            )
    return config
```

Model 相關的參數如下圖以及文字說明:

Model:Taiwan-LLaMa

Loss Function: CrossEntropyLoss()

Optimization: AdamW()

Learning Rate: 2e-4

Total Training Batch Size: 4*4=16

Epoch: 2

(B) Show your performance

Mean perplexity 為 4.81。

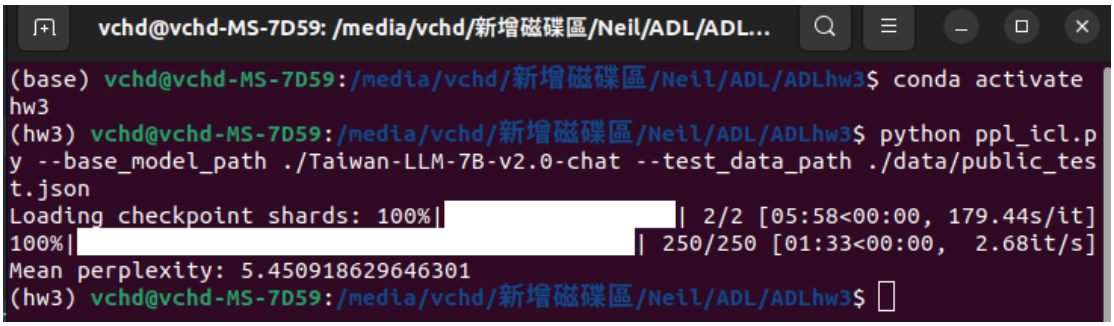
Learning Curve 我這次沒有畫，因為時間都花在調參上了，原本打算找到可以過 baseline 的參數再來寫 Learning Curve 的 code。

Q2：

(A) Zero-Shot

```
def get_prompt_zero_shot(instruction: str) -> str:
    '''Format the instruction as a prompt for LLM.'''
    return f"你是人工智慧助理，以下是用戶和人工智能助理之間的對話。\
    你要對用戶的問題提供有用、安全、詳細和禮貌的回答。USER: {instruction} ASSISTANT:"
```

我沒有改變助教一開始提供的 prompt，總之 zero-shot 的精神就是沒有給予任何的範例，因此我沒有給任何的範例。結果如下圖:



(B) Few-Shot (In-context Learning)

```
def get_prompt_few_shot(instruction: str) -> str:
    '''Format the instruction as a prompt for LLM.'''
    return f"請根據以下兩個例子進行翻譯工作。\n \
    例子一：翻譯成文言文：\n因此忠貞的臣子，並非不想竭盡忠誠，竭盡忠誠實在太難瞭。\n \
    答案：故忠貞之臣，非不欲竭誠。竭誠者，乃是極難。\n \
    例子二：翻譯成現代文：\n秦王惡之，後戒左右贊來不得通，贊亦不往，月一至府而已，退則杜門不交人事。\n \
    答案：秦王討厭他，後來告誡手下人劉贊來瞭不得通報，劉贊也不去，每月去王府一次罷瞭，迴來後就閉門不齣，不和人交往。\n \
    USER: {instruction} ASSISTANT:"
```

我提供了兩個例子給 LLM 作 ICL，我分別選了一個翻譯成文言文、翻譯成現代文的例子。希望能夠增強 model 對於不同語言的轉換能力。結果如下圖:

(C) Comparison

可以看到效果是 Few-shot > QLora > Zero-shot (儘管這三個應該還可以更好)，不過 Few-shot > Zero-shot 是非常合理的，畢竟提供了兩個例子給 model 學習。此外讓我驚訝到的是，ICL 真的可以不需要對下游任務進行 fine-tuning 就可以使得 model 有還不錯的效果，這也難怪近期 ICL 越來越盛行，畢竟可以省去不少計算資源。

後話：

前言有提到我一開始沒有將 prompt 加入 training data 中，取得的效果更好，且我自己做 human-evaluation 的結果也認為好很多，prediction.json 中不太會有如下圖的句子出現。下圖是我 model 的最終結果，也就是將 prompt 加入 training data 中的其中一些輸出:


```
{
  "id": "3d971fcf-f415-4ca6-8322-f84a93fe194c",
  "output": "翻譯成文言文："
},
{
  "id": "b5e24a2c-e8fc-41f8-93d4-a0640992b4cb",
  "output": "翻譯成現代文： ASSISTANT: 將這句話翻譯成現代文： ASSISTANT: 秦始皇時
},
```


圖中可以看到，我的 model generate 出了”翻譯成 xx 文”這種字眼，此外，甚至還出現了”ASSISTANT:”這個只出現在 prompt 的 word，所以我認為將 prompt 加入到 training data 中真的會帶來一部分的壞處，只是這個壞處有沒有勝過”讓 model 知道現在要做禮貌的回答”的好處我就不清楚了。希望改我這份作業的助教能幫我解答這個疑惑(可以寄信給我 r12944062@g.ntu.edu.tw)，到底該不該把 prompt 加入到 training data 中?