

ZH-TW Reading Comprehension Test for LLMs

1. Create DataSets

因為使用 Alpaca_llama3 的 ipynb 做 finetune，因此先將原始數據集整理成 json 格式方便後續訓練時配合讀取成 prompt 格式。

DataSets json:

```
{
  "instruction": "明年元旦起，交通部新規定，汽車輪胎胎紋深度將納入定期檢驗項目之一，一旦深度未達1.6公厘，及1個月內沒換新胎者，將被吊銷牌照。",
  "question": "汽車胎紋未到達多少公厘將會被吊銷牌照?\nOptions\n1) 1.5公厘\n2) 1.4公厘\n3) 1.3公厘\n4) 1.6公厘",
  "output": "4) 1.6公厘"
},
{
  "instruction": "塑化劑是一種「環境荷爾蒙」，又稱內分泌干擾物，指會干擾生物體內分泌之外因性化學物質。若長時間高劑量使用，會造成生殖系統發育不良、女性性早熟，以及成年男性精蟲減少等問題。",
  "question": "若長時間使用哪一種塑化劑會使男嬰生殖系統發育不良、女性性早熟，以及成年男性精蟲減少等問題?\nOptions\n1) DEHP",
  "output": "1) DEHP"
},
{
  "instruction": "紐西蘭的海岸邊近日出現數百具「小藍企鵝」的屍體，專家指出，這些企鵝都是被活活餓死的，會有企鵝大量餓死的原因為何?",
  "question": "使此處海水變暖的原因為何?\nOptions\n1) 洋流\n2) 全球暖化\n3) 反聖嬰現象\n4) 核能電廠排放廢水",
  "output": "3) 反聖嬰現象"
},
{
  "instruction": "綜合報導，科學期刊《當代生物學》昨天刊登一份美澳聯合研究，研究指出，近年澳洲大堡礁北部的新出生的綠蠵龜數量減少，可能與海洋溫度上升有關。",
  "question": "為甚麼雄性海龜出生率降低，雌性海龜出生率則升高?\nOptions\n1) 機率問題\n2) 全球暖化\n3) 環境質荷爾蒙影響",
  "output": "2) 全球暖化"
},
}
```

Prompt(可以自行調整):

```
alpaca_prompt = """Below is an instruction that describes a task, \
paired with an input that provides further context. Write a response that appropriately completes the request.

### Instruction:
{}

### Input:
{}

### Response:
{}"""
```

2. Training

- 參考連結:

<https://colab.research.google.com/drive/135ced7oHytdxu3N2DNe1Z0kqjyYIkDXp?usp=sharing#scrollTo=e2pEuRb1r2Vg>

分別使用不同的模型進行訓練，分別有

1. "unsloth/mistral-7b-v0.3-bnb-4bit"
2. "yentinglin/Llama-3-Taiwan-8B-Instruct"
3. "yentinglin/Taiwan-LLM-7B-v2.0-base"

使用 unsloth library:

可以更高效率的讀取模型；並寫更高效進行模型訓練。

可以在維持不錯生成回答精確度的同時降低記憶體在訓練時的使用量，目前測試能夠限縮到使用 6GB

1. mistral-7b-v0.3-bnb-4bit

<pre>{ "epoch": 0.035424354243542434, "grad_norm": 1.0700453519821167, "learning_rate": 0.0, "loss": 1.851, "step": 60 }, "logging_steps": 5, "max_steps": 60, "num_input_tokens_seen": 0, "num_train_epochs": 1, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": true }, "attributes": {} } }, "total_flos": 1.1945492300611584e+16, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>	<pre>{ "epoch": 0.2952029520295203, "grad_norm": 1.2494179010391235, "learning_rate": 0.00015037593984962405, "loss": 1.339, "step": 500 }, "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 9.79097073744937e+16, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>	<pre>{ "epoch": 0.5904059040590406, "grad_norm": 1.4831438064575195, "learning_rate": 0.00010025062656641604, "loss": 0.9664, "step": 1000 }, "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 1.9606912176134554e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>
Checkpoint-60	Checkpoint-500	Checkpoint-1000
<pre>{ "epoch": 0.8856088560885609, "grad_norm": 1.0758816003799438, "learning_rate": 5.012531328320802e-05, "loss": 0.9456, "step": 1500 }, "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 2.939574477582336e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>	<pre>{ "epoch": 1.1812546125461254, "grad_norm": 1.6668428182601929, "learning_rate": 0.0, "loss": 0.5626, "step": 2000 }, "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": true }, "attributes": {} } }, "total_flos": 3.9198505325101056e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>	
Checkpoint-1500	Checkpoint-2000	

2. Llama-3-Taiwan-8B-Instruct

<pre> { "epoch": 0.03247232472324723, "grad_norm": 0.4276879131793976, "learning_rate": 1.81818181818182e-05, "loss": 1.8216, "step": 55 }, { "epoch": 0.035424354243542434, "grad_norm": 0.5364223122596741, "learning_rate": 0.0, "loss": 1.8193, "step": 60 }], "logging_steps": 5, "max_steps": 60, "num_input_tokens_seen": 0, "num_train_epochs": 1, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": true }, "attributes": {} } }, "total_flos": 8988057514131456.0, "train_batch_size": 2, "trial_name": null, "trial_params": null } </pre>	<pre> { "epoch": 0.2952029520295203, "grad_norm": 0.7382127046585083, "learning_rate": 0.00015037593984962405, "loss": 1.4738, "step": 500 }], "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 7.361690663067648e+16, "train_batch_size": 2, "trial_name": null, "trial_params": null } </pre>	<pre> { "epoch": 0.5904059040590406, "grad_norm": 1.1535353660583496, "learning_rate": 0.00010025062656641604, "loss": 1.1433, "step": 1000 }], "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 1.4737400402814566e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null } </pre>
Checkpoint-60	Checkpoint-500	Checkpoint-1000
<pre> { "epoch": 0.8856088560885609, "grad_norm": 0.7359607815742493, "learning_rate": 5.012531328320802e-05, "loss": 1.1024, "step": 1500 }], "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 2.2100114422254797e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null } </pre>	<pre> { "epoch": 1.1812546125461254, "grad_norm": 1.2499291896820068, "learning_rate": 0.0, "loss": 0.735, "step": 2000 }], "logging_steps": 5, "max_steps": 2000, "num_input_tokens_seen": 0, "num_train_epochs": 2, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": true }, "attributes": {} } }, "total_flos": 2.9462049604101734e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null } </pre>	
Checkpoint-1500	Checkpoint-2000	

3. Taiwan-LLM-7B-v2.0-base

<pre>}, { "epoch": 0.2952029520295203, "grad_norm": 0.32989731431007385, "learning_rate": 0.00010050251256281407, "loss": 0.9999, "step": 500 }], "logging_steps": 5, "max_steps": 1000, "num_input_tokens_seen": 0, "num_train_epochs": 1, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": false }, "attributes": {} } }, "total_flos": 1.114409468139602e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>	<pre>{ "epoch": 0.5904059040590406, "grad_norm": 0.43240800499916077, "learning_rate": 0.0, "loss": 0.9144, "step": 1000 }], "logging_steps": 5, "max_steps": 1000, "num_input_tokens_seen": 0, "num_train_epochs": 1, "save_steps": 500, "stateful_callbacks": { "TrainerControl": { "args": { "should_epoch_stop": false, "should_evaluate": false, "should_log": false, "should_save": true, "should_training_stop": true }, "attributes": {} } }, "total_flos": 2.2329588875339366e+17, "train_batch_size": 2, "trial_name": null, "trial_params": null }</pre>
Checkpoint-500	Checkpoint-1000

3. Inference

使用 usloth 載入模型：

```
# 加載最佳模型和 Tokenizer
model, tokenizer = FastLanguageModel.from_pretrained(
    model_name = model_dir,
    max_seq_length = max_seq_length,
    dtype = dtype,
    load_in_4bit = load_in_4bit,
    # token = "hf_...", # use one if using gated models like m
)

# 调用 for_inference
FastLanguageModel.for_inference(model)
```

用與 train 相同的 prompt 進行 test 的讀取。

```
def formatting_prompts_test(examples):
    instructions = examples["instruction"]
    inputs       = examples["question"]
    outputs      = examples["output"]
    texts = []
    for instruction, input, output in zip(instructions, inputs, outputs):
        text = alpaca_prompt.format(instruction, input, output)
        texts.append(text)
    return { "text" : texts }
```

最後按照平常的格式進行文字生成

```
with torch.no_grad():
    print("Start inference.")
    results = []
    for test_dataset in tqdm(test_datasets, desc="inference: "):

        id = int(test_dataset['id'])
        input_text = str(test_dataset['text'])

        # 生成回答
        inputs = tokenizer(input_text, return_tensors="pt").to("cuda")
        generation_output = model.generate(**inputs, max_new_tokens = 64, use_cache = True)

        s = generation_output[0]

        output = tokenizer.decode(s, skip_special_tokens=True)
        response = output.split("### Response:")[1].strip()

        answer = response.split(' ')[0]

        # print(f"====={id}=====")
        # print(f"Input: {input_text}\n")
        # print(f"Response: {response}\n")

        results.append({"id":id, "answer":answer})

    print("End inference.")
```

P. S

機器學習：

【分類問題】：需要 label 正確答案，作為參照。

【文字生成】：不需要 label。需要將所有文字文章、問題、選擇 1~4、答案，利用自行設定的 prompt 串接成一個 text，作為 input。Encode/decode 都使用 tokenizer，不過 Encode 部分可以包還在 SFTTrainer、Trainer 訓練器做轉換，不須在資料前處理的時候就先手動轉換 Encode。避免發生不必要的轉換錯誤。