ADL 2022 Fall - HW3 Report

Yang, Yu Chun (D11922022)

tags: 課程報告

Q1: Model (2%)

1. Model (1%)
   Describe the model architecture and how it works on text summarization.

   採用 `google/mt5-small` 這是一個透過涵蓋 101 種語言的 `mc4` 資料集訓練成的多語言 pre-trained model。它的結構是 encoder-decoder。在 text summarization 這個任務中，文章會被餵入 encoder，encoder 的 hidden layer 會跟 decoder 以 cross-attention 的方式同時考慮來自雙方的資訊，並以 auto-regressive 的方式逐一產生 decoder output 直到產生 `<EOS>` token 為止。
   模型 config：


```
{
  "_name_or_path": "ckpt/f2989158",
  "architectures": [
    "MT5ForConditionalGeneration"
  ],
  "d_ff": 1024,
  "d_kv": 64,
  "d_model": 512,
  "decoder_start_token_id": 0,
  "dense_act_fn": "gelu_new",
  "dropout_rate": 0.1,
  "eos_token_id": 1,
  "feed_forward_proj": "gated-gelu",
  "initializer_factor": 1.0,
  "is_encoder_decoder": true,
  "is_gated_act": true,
  "layer_norm_epsilon": 1e-06,
  "model_type": "mt5",
  "num_decoder_layers": 8,
  "num_heads": 6,
  "num_layers": 8,
  "pad_token_id": 0,
  "relative_attention_max_distance": 128,
  "relative_attention_num_buckets": 32,
  "tie_word_embeddings": false,
  "tokenizer_class": "T5Tokenizer",
  "torch_dtype": "float32",
  "transformers_version": "4.24.0",
  "use_cache": true,
  "vocab_size": 250100
}
```

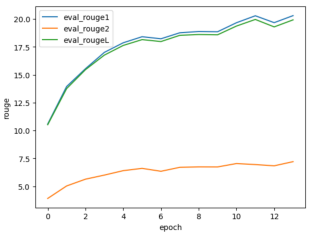

2. Preprocessing (1%)
   Describe your preprocessing (e.g. tokenization, data cleaning and etc.)
   使用基於 `SentencePiece` 的 `mt5` tokenizer，對新聞內文和標題做斷詞，如果超過長度，則會被 truncate。跟斷詞相關的設定如下：
   - `max_length`：256
   - `max_target_length`：64

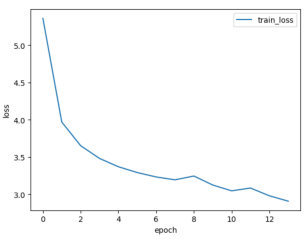
Q2: Training (2%)

1. Hyperparameter (1%)
   Describe your hyperparameter you use and how you decide it.
   hyperparameter 大多數是跟著 sample code 的配置。

超參數	變數名	設定值
pretrained model	`model_name_or_path`	`google/mt5-small`
optimizer	`optimizer`	`AdamW`
learning rate	`learning_rate`	`5e-5`
gradient accumulation steps	`gradient_accumulation_steps`	前 `8` epoch 為 `4` 後 `6` epoch 為 `1`
batch size	`total_batch_size`	`2`
fp16	`use_fp16`	`False`
num epoch	`num_train_epochs`	`14`

2. Learning Curves (1%)
   Plot the learning curves (ROUGE versus training steps)





Q3: Generation Strategies(6%)

1. Stratgies (2%)
   Describe the detail of the following generation strategies:
   - Greedy
     生成時，總是選擇機率最高的作為下一個字。
   - Beam Search
     是在每一 time step 中保持一定數量的輸出字作為候選名單，而在輸出最終結果時將列在候選名單上的序列輸出。
   - Top-k Sampling
     每次生成字時取前 K 機率值最高的為 candidate，經過 normalize 後，再按其機率抽樣。
   - Top-p Sampling
     每次生成字時取前面幾個機率值最高的為 candidate，且 candidate 機率值加總要超過 threshold p。經過 normalize 後，再按其機率抽樣。
   - Temperature
     類似於 softmax 的作用，能將機率值高的字，跟機率值低的字其間的機率落差拉得更開，也就是讓機率值高者的機率更高，讓機率值低者的機率更低。當 temperature > 1 則為 sharpen 的效果；當 temperature < 1 則為 smoothen 的效果。

2. Hyperparameters (4%)
   - Try at least 2 settings of each strategies and compare the result.
   - What is your final generation strategy? (you can combine any of them)

stretgy & setting	rouge1(`f1*100`)	rouge2(`f1*100`)	rougeL(`f1*100`)
greedy	25.580	9.654	22.751
beams 2	26.490	10.411	23.596
beams 4	26.693	10.854	23.919
temperature 0.3	25.0969	9.3507	22.3396
temperature 0.9	19.2667	9.3507	22.3396
top-k 5	23.837	8.317	21.103
top-k 20	21.863	7.407	19.364
top-p 0.75	23.128	8.170	20.419
top-p 0.95	21.064	7.0540	18.615

   根據 `rouge1` 最高分者，最終我選用 `beams 4` 作為 generation strategy

Bonus: Applied RL on Summarization (2%)

1. Algorithm (1%)
   Describe your RL algorithms, reward function, and hyperparameters.

2. Compare to Supervised Learning (1%)
   Observe the loss, ROUGE score and output texts, what differences can you find?