Q1: Model

- ### Model

  mt5 為 encoder-decoder 架構，將 source 放入 encoder 並將結果存入 hidden state，再將 target 放入 decoder 並跟 hidden state 做 attention，將輸出結果重複跟 hidden state 做 attention，最終可得每個字出現的機率。

- ### Preprocessing

  mt5 使用的方式為 sentencepiece，因涵蓋很多語言，會先將每個字切成 subword，並選擇統計出較高機率出現的詞。

Q2: Training

- ### Hyperparameter

  a. model: google/mt5-small

  b. performance(f*100): {rouge-1: 23.3 , rouge-2: 8.7, rouge-L: 21.0}

  c. loss function: torch.nn.CrossEntropyLoss()

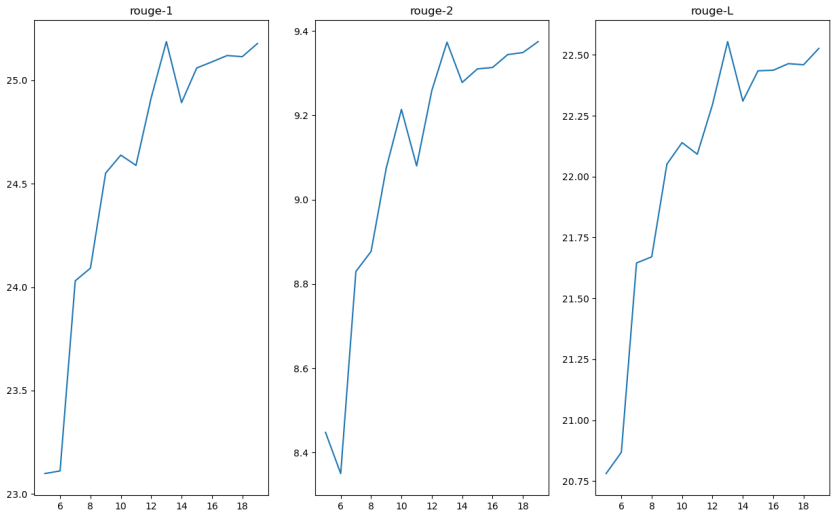
  d. optimization algorithm: torch.optim.AdamW()

  learning rate: 3e-5, batch size: 8, epoch: 20

  max_source_length:256, max_target_length:64

```
{
  "rouge-1": {
    "r": 0.21573511639058904,
    "p": 0.2774223415502372,
    "f": 0.2336016909556458
  },
  "rouge-2": {
    "r": 0.08252923851355029,
    "p": 0.1009435487119065,
    "f": 0.08736272507067948
  },
  "rouge-l": {
    "r": 0.19438947391357286,
    "p": 0.25026015995067485,
    "f": 0.21048205633417014
  }
}
```


- Learning Curves



Q3: Generation Strategies

- Strategies

1. Greedy

根據 $w_t = argmax_w P(w|w_{1:t-1})$ ，每個時間選擇機率最高的

詞，缺點是可能會造成先取到的機率較高的詞，但整體的機率不一定為最佳解。

2. Beam Search

會設定 beam 值，如果 beam 設成 3，則將每次選擇機率前 3 的詞紀錄下來，減少錯過最佳路徑的選擇，補足 greedy 的缺點，整體的機率會比 greedy 高。

3. Top-k Sampling

根據$w_t \sim P(w|w_{1:t-1})$，每個單詞從該機率，從中隨機選出下個出現的詞。而我們將前 k 高的機率挑出來，並根據機率重新分配機率給這 k 個值，每次結果就不會都相同。

4. Top-p Sampling

會設定 p 值，將最少個數組成並超過 p 的累積機率挑出來，因此子集會越變越小，相較 top-k 下一個詞比較容易預測。

5. Temperature

根據 temperature 並利用 softmax，可將原本就高機率出現的值更容易出現，而原本很少出現則變的更不容易出現。

- Hyperparameters

使用 Beam、Top_k、Top_p 跟原先的準確率做比較。發現使用 beam，其準確率有明顯提升；使用 top_k，會比原先的準確率低，但 top_k

的大小關係，對準確率的影響較小；使用 top_p，亦比原先的準確率低，但較小的 top_k 值，能產生較高的準確率。

	Rouge-1(f*100)	Rouge-2(f*100)	Rouge-L(f*100)
Origin	23.4	8.7	21.0
Beam = 5	24.7	9.9	22.3
Top_k = 20	18.2	5.6	16.1
Top_k = 50	18.1	5.7	16.1
Top_p = 0.66	18.4	6.3	16.5
Top_p = 0.95	14.8	4.7	13.3

Origin


```
{
  "rouge-1": {
    "r": 0.21573511639058904,
    "p": 0.2774223415502372,
    "f": 0.2336016909556458
  },
  "rouge-2": {
    "r": 0.08252923851355029,
    "p": 0.1009435487119065,
    "f": 0.08736272507067948
  },
  "rouge-l": {
    "r": 0.19438947391357286,
    "p": 0.25026015995067485,
    "f": 0.21048205633417014
  }
}
```


Beam = 5


```
{
  "rouge-1": {
    "r": 0.23031924387976538,
    "p": 0.2850968301507741,
    "f": 0.24679006570105394
  },
  "rouge-2": {
    "r": 0.09351326558685344,
    "p": 0.11381467479707866,
    "f": 0.09940584133010054
  },
  "rouge-l": {
    "r": 0.2079697886166981,
    "p": 0.25750786004686277,
    "f": 0.2227204532136968
  }
}
```


Top_k = 20


```
{
  "rouge-1": {
    "r": 0.1794017920077628,
    "p": 0.19900602037068615,
    "f": 0.18247576807178031
  },
  "rouge-2": {
    "r": 0.056699236094770906,
    "p": 0.06060974178494806,
    "f": 0.05646328369827098
  },
  "rouge-l": {
    "r": 0.15861706203043488,
    "p": 0.17577018499704108,
    "f": 0.1611201649507779
  }
}
```


Top_k = 20


```
{
  "rouge-1": {
    "r": 0.17790108259368972,
    "p": 0.197246348095154,
    "f": 0.18096904899955324
  },
  "rouge-2": {
    "r": 0.05647243204266725,
    "p": 0.061062836653327866,
    "f": 0.05660724669717872
  },
  "rouge-l": {
    "r": 0.15784640567952318,
    "p": 0.1751805788153068,
    "f": 0.1605380932968017
  }
}
```

Top_p = 0.66


```
{
  "rouge-1": {
    "r": 0.18024988603596132,
    "p": 0.20138542662212575,
    "f": 0.18425912774704364
  },
  "rouge-2": {
    "r": 0.06242165006131896,
    "p": 0.06796350476802865,
    "f": 0.0627249761784927
  },
  "rouge-l": {
    "r": 0.16095852959809026,
    "p": 0.18008116899524929,
    "f": 0.16457234115727637
  }
}
```


Top_p = 0.95


```
{
  "rouge-1": {
    "r": 0.14854196028210462,
    "p": 0.15742477574391694,
    "f": 0.14789322964267373
  },
  "rouge-2": {
    "r": 0.046957522083644794,
    "p": 0.049642374048130385,
    "f": 0.04652573801705475
  },
  "rouge-l": {
    "r": 0.13360758859442465,
    "p": 0.1417046378582896,
    "f": 0.132953363141229
  }
}
```