ADL2022 HW3 report

R09922126 黃宇瑍

Q1: Model

- mt5-small is a transformer encoder-decoder created by Google.

The input of it is the t5-tokenized text sequence. And after go through the entire transformer , it will produce a logits distribution. Then, they will pass through the softmax layer and produce a probability distribution(or confidence scores). We can use any sample method to construct a sentence based on the distribution.

Q1: Model

- Preprocessing:
  - Tokenization: t5 tokenizer is sentencepiece based, which is compose of byte-pair-encoding and unigram-language-model. It will cut sentence into subwords and use Viterbi-algorithm to find the highest probability path to tokenize.

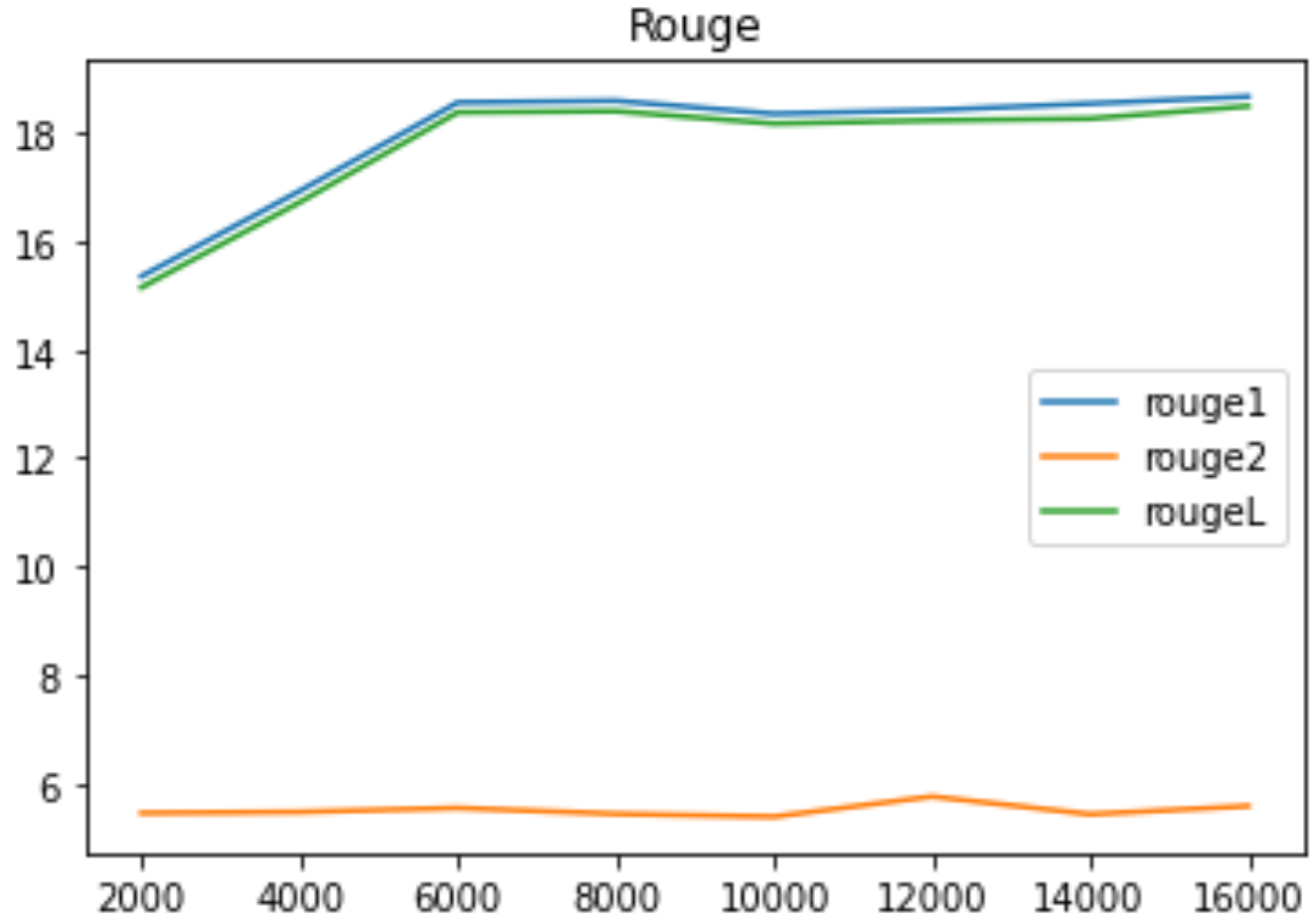
Q2: Training

- Hyperparameter:
  - Epochs: 12
  - batch size: 16
  - Warmup ratio: 0.1
  - Weight decay: 0.1

The epochs and batch size are random choosed. Warmup ratio can prevent fast convergence to a local minimum. Weight decay can reduce overfitting situation.

Q2: Training

- Learning Curves(caculate by load_metric(“rouge”)):



Q3: Generation Strategies

- Stratgies:
  - Greedy: This method will choose the token(word) with highest probability at every time step.
  - Beam Search: With a beam size, for example , 3, in every time step, it will choose the top 3 tokens(words) as candidates. Then, it will multiplies the probability of it with all previous tokens as a score and stores them. So, in the first time step, it will have 3 candidates, after that, it will have 9(3x3) candidates and preserve the top 3 candidates(base on the score) until the last time step. In the last time step, it will choose the highest score one as the final result.

Q3: Generation Strategies

- Stratgies:
  - Top-k Sampling: In every time step, it only preserves the top k probability tokens as candidates, and sample a token(word) from them.
  - Top-p Sampling: In every time step, it only preserves the tokens(words) with probability higher than p, and sample a token from them.
  - Temperature: Before the softmax layer, the output logits will be divide by T(Temperature). Then it will go through the softmax layer and output a probability. After that, we do greedy sampling.

Q3: Generation Strategies

- Hyperparameters:
  - I tried several methods.

  We can see, beam seach is performed all other methods.

  And greedy is very similar to top-p with p = 0.9.

  Finnaly, I choose beam search with size 15 as final result.

	greedy	Beam3	Beam5	beam15	Top-p 0.9
Rouge1-r	0.2875	0.3058	0.3085	-	0.2875
Rouge1-p	0.1913	0.2030	0.2086	0.2141	0.1913
Rouge1-f	0.2237	0.2373	0.2421	**0.2462**	0.2237
Rouge2-r	0.1016	0.1124	0.1149	0.1163	0.1016
Rouge2-p	0.0588	0.0666	0.0697	0.0724	0.0588
Rouge2-f	0.0725	0.0813	0.0844	**0.0869**	0.0725
RougeL-r	0.2511	0.2670	0.2693	0.2705	0.2511
RougeL-p	0.1661	0.1762	0.1812	0.1864	0.1661
RougeL-f	0.1946	0.2064	0.2106	**0.2145**	0.1947

Bonus: Applied RL on Summarization

- Algorithm:
  - I use classical policy gradient as the RL algorithm. For every generated sentence, I assign the same reward to every single token(word). The reward functions I tried are rouge-1 and rouge-L. Each reward function make the model more robust for the model.
- Compare to Supervised Learning:
  - The most significant difference is that, in the RL methods bellow, they all need only one epoch to pass the baseline. But supervised learning method needs 5 or more epochs to pass it.

Bonus: Applied RL on Summarization

As we can see, with RL algo. the scores can improved dramatically.

Most importantly, RL methods only needs 1 epoch to pass the base line.

And rouge-1 performs more robust than rouge-L.

(RL methods are all producing sentence in greedy way.)

	Beam search 15 (12 epochs)	Reward: rouge-1 (1 epoch)	Reward: rouge-L (1 epoch)
Rouge1-r	-	**0.2569**	0.2434
Rouge1-p	0.2141	**0.3002**	0.2881
Rouge1-f	0.2462	**0.2747**	0.2618
Rouge2-r	**0.1163**	0.0948	0.0866
Rouge2-p	0.0724	**0.1038**	0.0959
Rouge2-f	0.0869	**0.0983**	0.0903
RougeL-r	**0.2705**	0.2464	0.2331
RougeL-p	0.1864	**0.2875**	0.2754
RougeL-f	0.2145	**0.2633**	0.2506

Bonus: Applied RL on Summarization

- Text generation comparation:
  - Beam search size 15

```
beam15.head()
```

	id	title
0	21710	Anker推出真無線藍牙耳機Liberty Air 2 Pro 加入主動式降噪、收音、收音與...
1	21711	彷彿置身鐵路藝術極品!苗栗「三義舊山線鐵道自行車」精選,客家美食+客家美食+客家美食
2	21712	華碩推出換上Intel第11代Core處理器的Chromebook Flip C5 加入支援...
3	21713	掌握疫後智慧商機/理財我最大 帶領讀者掌握疫後產業發展重點 產科國際所長帶領讀者掌握產業新重點
4	21714	全球僅有15億台!Windows 7 已經退出消費市場 但仍有多少裝置仍有2億台上看2億台後...

Bonus: Applied RL on Summarization

- Text generation comparation:
  - Rouge1 greedy

```
rouge1.head()
```

	id	title
0	21710	Anker新款真無線藍牙耳機LibertyAir2Pro預進台灣市場
1	21711	鐵綠鐵綠家美食!美食山線鐵車!三栗鐵遊」氣鐵景
2	21712	華碩推出換應15.6規安全護認安全in1使用Chrome吋機bookFlip
3	21713	疫情發展重革蘇產業產業勢產業會
4	21714	微僅7置的暴超過15億台但佔率達定這

Bonus: Applied RL on Summarization

- Text generation comparation:
  - RougeL greedy

```
rougeL.head()
```

	id	title
0	21710	Anker新款真無線藍牙耳機Air2Pro預進台灣市場
1	21711	苗綠鐵綠家必推薦景山線鐵車!舊栗鐵遊」氣路線推薦景
2	21712	華碩換Chrome應16規護防in1使用Chrome吋機book
3	21713	疫情發展大革二產業產業勢挑會
4	21714	微僅7置的暴1115億台但佔率恐定這

Bonus: Applied RL on Summarization

- Text generation comparation: As we can see in the previous examples. The fluency of supervised learning is greater than rouge RL based methods. But still, RL methods gets the higher scores.
- So we can tell, RL methods may lead to overfit on reward function. Which makes sense that it gets the higher score but outputs a lot of trashes. So we may need to fine tune the weight of reward to prevent overfitting, or even introduce dynamic weighted reward.

e.g. In the beginning , the weight is very low. And it keeps going up until the end of the training.