HW2

id: r12922192

name: 邱冠珅

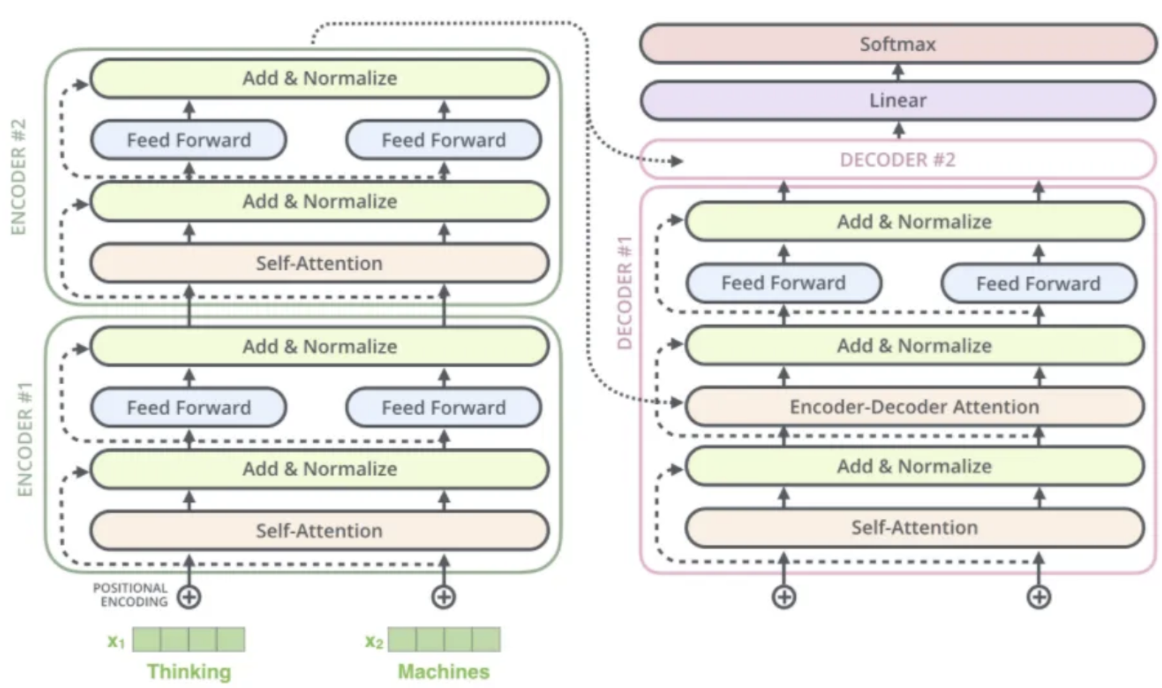
Q1: Model

Model:

Describe the model architecture and how it works on text summarization.

- Architecture:

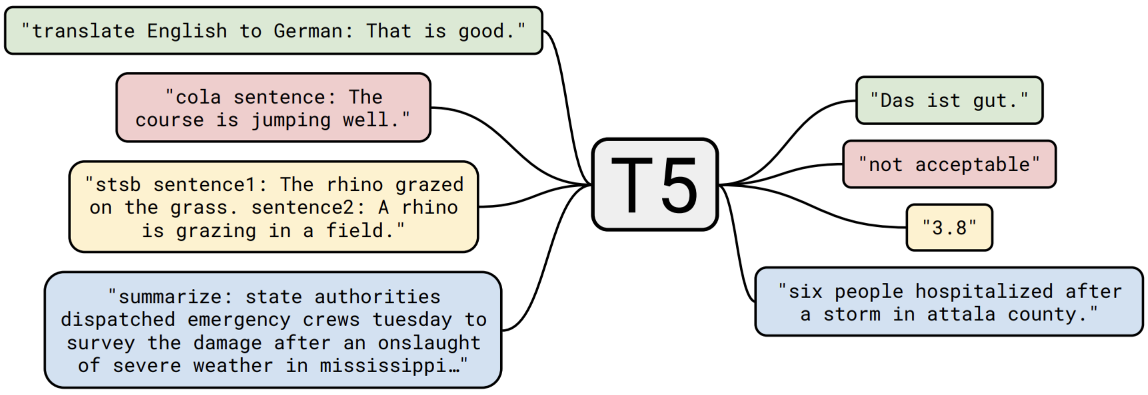
  I am using the multilingual T5-small model (MT5-small) with 77 million parameters. The architecture of MT5 is based on T5.1.1, which is a vanilla encoder-decoder transformer with 8 encoder layers and 8 decoder layers. It was pretrained on the C4 dataset and uses the Gated GELU (GEGLU) activation function instead of ReLU. Dropout was disabled during pretraining and is enabled when I am training the model.

  Architecture from: [The Illustrated Transformer – Jay Alammar – Visualizing machine learning one concept at a time. (jalammar.github.io)]

- Text summarization:

  The T5 paper mentions that they approach every task as a "text-to-text" mission and conduct pretraining on C4 datasets. Text summarization can also be viewed as a text-to-text mission, which is why mT5 is naturally capable of performing text summarization tasks.



from: [1910.10683] Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (arxiv.org)

Preprocessing:

Describe your preprocessing (e.g. tokenization, data cleaning and etc.)

- tokenization

  I use the T5TokenizerFast tokenizer, which utilizes SentencePiece to split the sentence using the Unigram algorithm and truncates the input to a length of 256 and the output to 64.

- data cleaning

  Before training, I replaced '\n' with a space and removed unknown symbols from the input, such as ∞, 👍, ☀️, 😷, 🌹, 😊, D,, ▀, ☕, 🟫, 🎄, 🚴, 🌞, 🎬, 🤪, 🙅, 〖, ➽, 🚬, but I did not remove unknown symbols that appeared in the training dataset's title references.

Q2: Training

Hyperparameter:

- Describe your hyperparameter you use and how you decide it.

```
--per_device_train_batch_size 8 \
--gradient_accumulation_steps 2 \
--max_source_length 256 \
--max_target_length 64 \
--lr_scheduler_type linear \
--learning_rate 1e-4 \
--optimizer torch.optim.AdamW \
--weight_decay 5e-5 \
--num_beams 2 \
--num_train_epochs 50 \

#To achieve the public baseline, use this hyperparameter set.
#Test on higher learning rates and lower learning rates,
#and performance is the best at 1e-4.

#Set the batch size to 8*2;
#using other settings such as 4*4 for training requires more time.
```

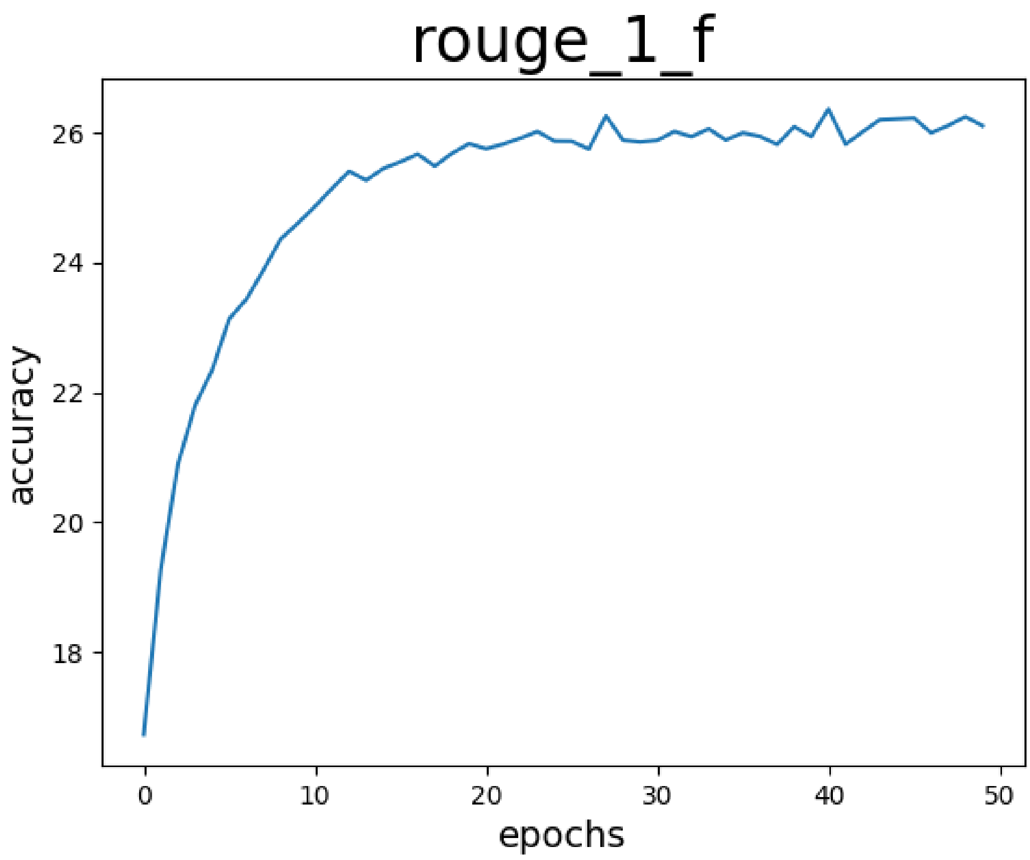
```
#For other hyperparameters, follow the TA's suggestions and use the default settings."
```

Learning Curves:

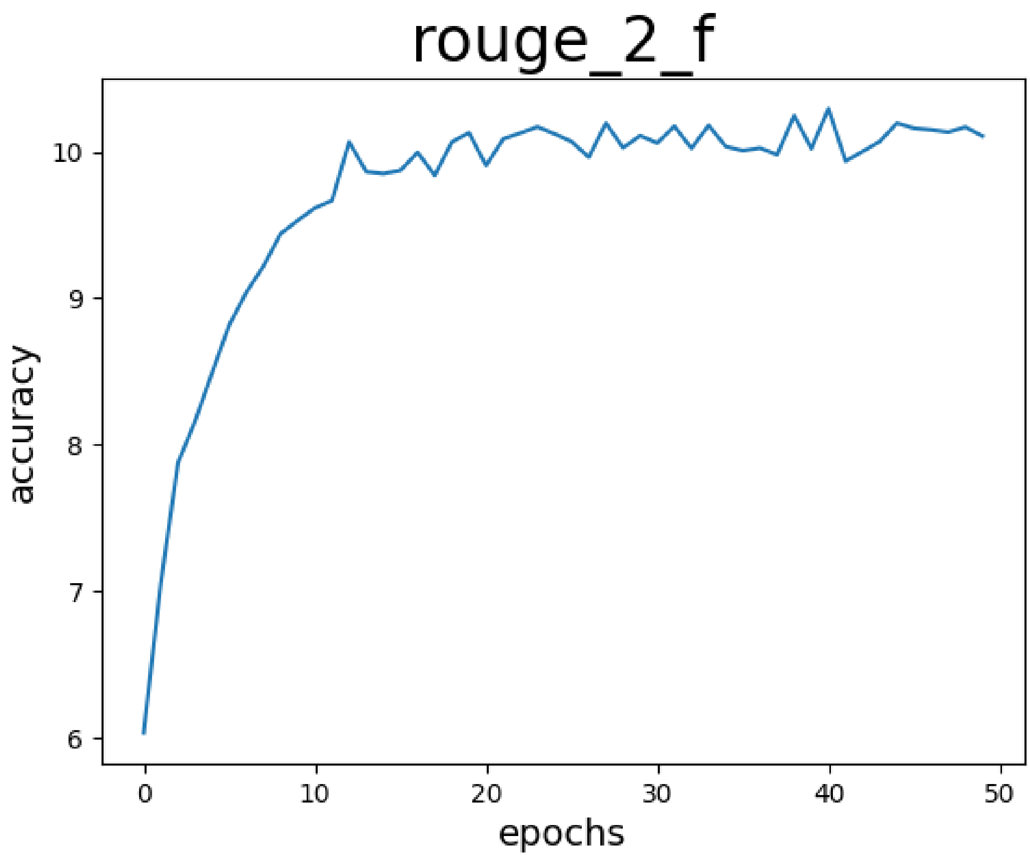
Plot the learning curves (ROUGE on public set versus training steps)

(all unit is f1 score*100)

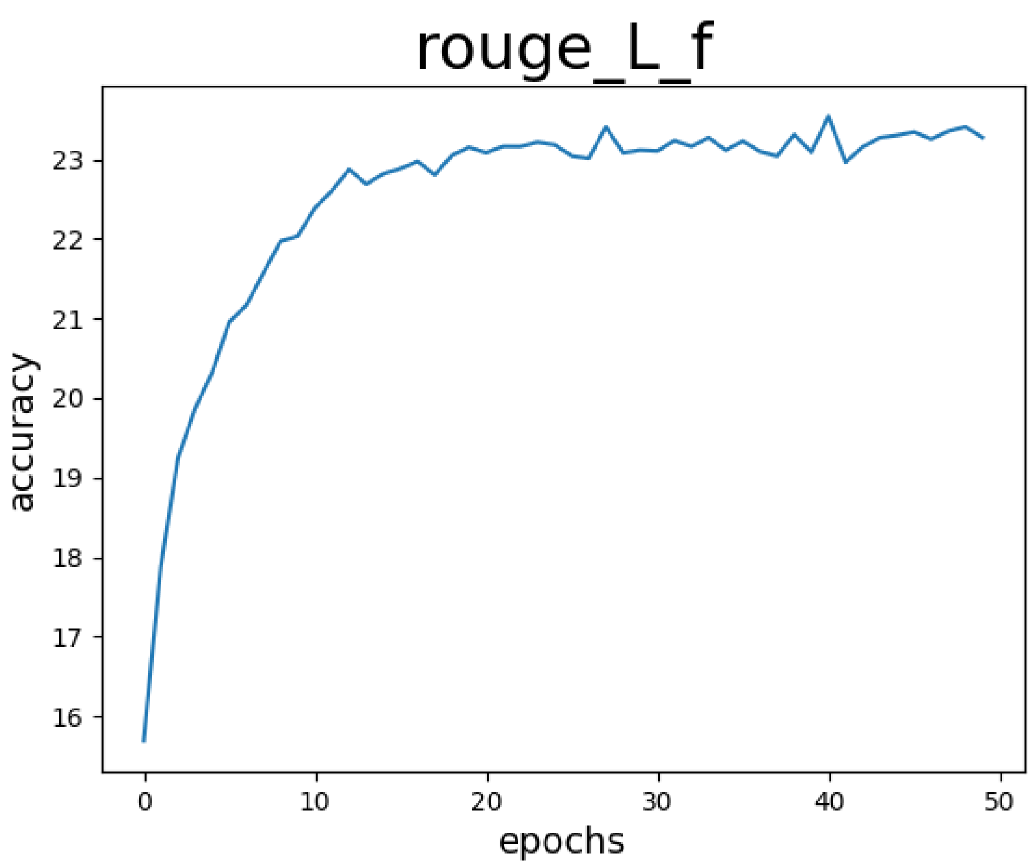
- **rouge-1**



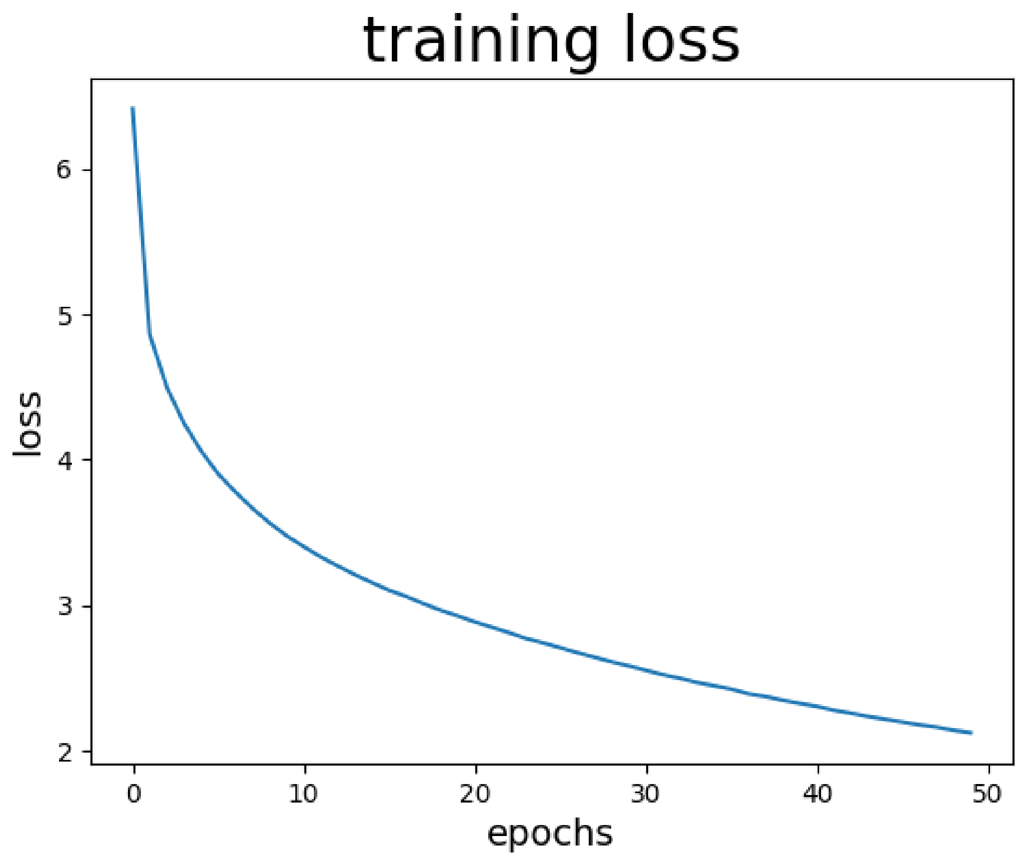
- **rouge-2**



- **rouge-L**



- **training loss**



Q3: Generation Strategies

Strategies:

Describe the detail of the following generation strategies.

- Greedy

  Greedy search refers to the strategy of selecting the next word at each time step based on probability, choosing the word with the highest probability.

- Beam Search

  The most significant issue with Greedy Search is that it only considers the current high-probability word, ignoring high-probability words that might come after low-probability words. Beam Search solves this issue by not only choosing one word with the highest probability but also remembering the next n-1 highest probability words at every step. It then selects the sentence with the highest probability among these n sentences.

- Top-k Sampling

  Selecting the top-k words with the highest probabilities, forming a subset of these words, and then normalizing the probabilities within this subset. Subsequently, sampling from the renormalized probability distribution within this subset of words.

- Top-p Sampling

  Unlike the approach of top-k, which directly discards low probability vocabulary, top-p sampling utilizes cumulative probabilities. In other words, its samples from the vocabulary where the cumulative probability exceeds a certain threshold p. In simple terms, by adjusting the parameter p (0<=p<=1), Top-P Sampling increases the likelihood of generating words with lower occurrence probabilities.

- Temperature

  Temperature sampling calculate the probability distribution for each candidate word, then divide these probability values by a parameter called "temperature." Then, input them into the SoftMax to compute a new probability distribution, and select a word from it.

Hyperparameters:

- Try at least 2 settings of each strategies and compare the result.

  all scores are f1 standard.

  Top_k Sampling: num_beams = 1, top_p = 1.0, temperature = 1.0

  Top_p Sampling: num_beams = 1, top_k = 50, temperature = 1.0

  Temperature Sampling: num_beams = 1, top_k = 50, top_p = 1.0

strategy	rouge-1	rouge-2	rouge-L
greedy(beams = 1)	25.508	09.498	22.685
beam search(beams = 2)	26.364	10.294	23.545
beam search(beams = 3)	26.636	10.609	23.809
beam search(beams = 4)	26.681	10.677	23.864
beam search(beams = 5)	26.747	10.734	23.942
beam search(beams = 7)	26.838	10.826	23.975
beam search(beams = 10)	26.877	10.847	23.920
top k (top_k = 5)	23.898	08.263	21.022
top k (top_k = 10)	22.980	07.806	20.244
top k (top_k = 15)	22.395	07.691	19.721
top k (top_k = 25)	22.341	07.607	19.659
top k (top_k = 50)	21.047	07.005	18.624
top k (top_k = 75)	20.915	06.932	18.465
top k (top_k = 100)	20.722	06.891	18.180
top p (top_p = 0.1)	25.560	09.529	22.721
top p (top_p = 0.5)	24.687	09.055	21.894
top p (top_p = 0.7)	23.837	08.508	21.114
top p (top_p = 1.0)	21.419	07.074	18.835
top p (top_p = 5.0)	21.193	07.043	18.671
temperature (temperature = 0.5)	24.693	08.955	21.908
temperature (temperature = 1.0)	21.067	06.961	18.597
temperature (temperature = 2.0)	12.449	02.582	10.861
temperature (temperature =5.0)	05.558	00.226	04.783

- What is your final generation strategy? (you can combine any of them)

  Choose beam sampling and num_beams = 7.