ADL Homework 3 Report

Name: Chou Tun Hsiang **Student Number:** B07902050 **Date:** June 2 2021

1 Model

The model used in this homework is Google's small multilingual T5. The structure of mT5 model is same as T5, except that mT5 is trained with 101 languages, and the pretraining objective includes T5's self-supervised training only, but not T5's supervised training.

According to the original T5 paper, "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer", the main structure of the model is similar to that of a encoder-decoder Transformer. First, an input sequence of tokens is mapped to a sequence of embeddings, which is then passed into the encoder. The encoder consists of a stack of "blocks", each of which comprises two subcomponents: a self-attention layer followed by a small feed-forward network. Layer normalization is applied to the input of each subcomponent, followed by a residual skip connection adds each subcomponent's input to its output.

The decoder is similar in structure to the encoder except that the self-attention mechanism in the decoder also uses a form of autoregressive or causal self-attention, which only allows the model to attend to past outputs. The output of the final decoder block is fed into a dense layer with a softmax output, whose weights are shared with the input embedding matrix.

To avail the same model for all the downstream tasks, a task-specific text prefix is added to the original input that is fed to the model. Then the model is trained to generate some target text. Specifically, in this news summarization task, the prefix 'summarize: ' is added to the articles.

2 Preprocessing

All articles are truncated to 256 words and tokenized before passed into the model. The tokenizer used by the mT5 model is based on SentencePiece, which is an unsupervised text tokenizer and uses BPE or unigram algorithm to construct the appropriate vocabulary.

3 Training Hyperparameter

First, I tried to use different learning rate to train the model and generate titles with 5-beams search. 2-grams penalties were added as restriction to avoid repetitions in the predicted text. The maximum of generated titles are 64 words. The results are shown in table 1. and 2. It seems that a larger learning rate helps training. Perhaps a value of 6e-5 or 8e-5 is worth a try. Next, the learning rate was fixed to be 4e-5 to observe the effect of weight decay. The results are illustrated in table 3. Compared to learning rate, the value of weight decay has a relative small impact on the performance. Finally, a learning rate 4e-5 and weight decay 1e-3 are chosen.

epoch	weight decay	batch size	optimizer
20	0.01	16	Adam

Table 1: Training configurations

learning rate	1e-5	2e-5	3e-5	4e-5
rouge-1	22.63	23.47	23.59	**24.65**
rouge-2	9.02	9.49	9.72	**9.86**
rouge-L	20.52	21.12	**22.28**	22.11

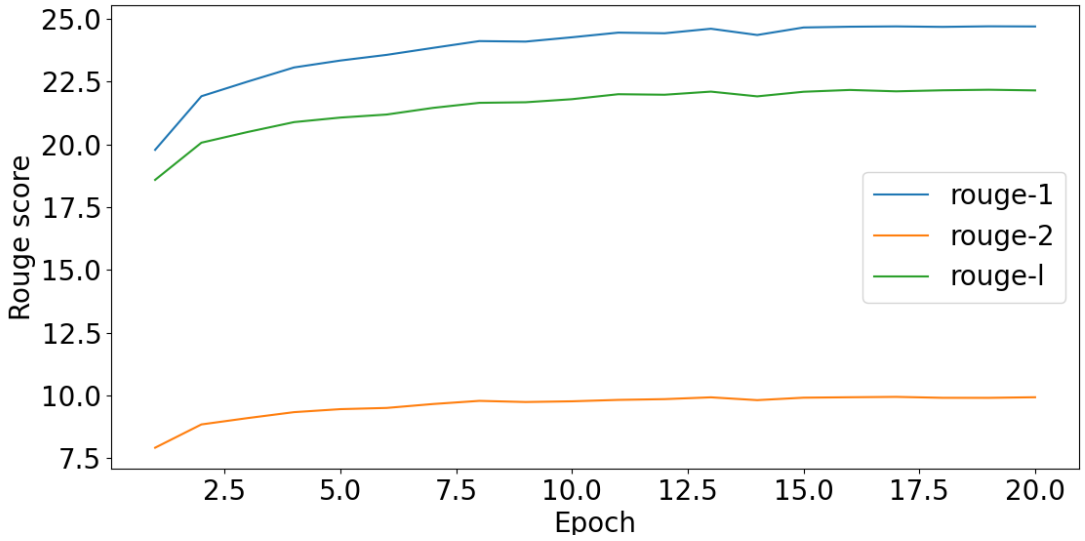
Table 2: Rouge score with different learning rate

weight decay	1e-3	5e-3	1e-2	5e-2
rouge-1	**24.70**	24.59	24.65	24.69
rouge-2	**9.92**	9.84	9.86	9.92
rouge-L	22.15	22.03	22.11	**22.15**

Table 3: Rouge score with different weight decay value

4 Learning Curves

The training configurations is the same as in table 1. with learning rate 4e-5. As shown by the line plot, the scores increase significantly before 2.5 epochs, then start to improve gently from 2.5 to 12.5 epochs. Then the training process hits a plateau after about 15 epochs.



5 Strategies

5.1 Greedy

Greedy search always selects the next word with highest probability.

5.2 Beam Search

To reduce the risk of missing hidden high probability word sequences, beam search keeps several hypotheses in hand at each time step and eventually chooses the hypothesis with the overall highest probability.

5.3 Top-k Sampling

The K most likely next words are filtered and the probability mass is redistributed among only those K next words. Then the next word is randomly picked according to its conditional probability distribution.

5.4 Top-p Sampling

Instead of sampling only from the most likely K words, Top-p sampling chooses from the smallest possible set of words whose cumulative probability exceeds the probability p. The probability mass is then redistributed among this set of words.

5.5 Temperature

Temperature is a hyperparameter applied to logits to affect the final probabilities from the softmax. With T approaching to 1, the output distribution will be softer and finally become normal softmax output.

6 Generation Hyperparameter

The training configurations of the base model are listed in table 4, and the rouge scores of different generation strategies are shown in table 5. Overall, beam search outperforms other strategies, followed by greedy generation. This kind of surprised me because top-k and top-p sampling seems to be more sophisticated methods, and they were proposed to solve the problem that, in greedy search and beam search, the resulting text tends to repeat itself. However, the issue can also be avoided if we add n-gram penalties to the beam search or greedy search model. As a result, I chose beam search with 5 beams as my final approach.

epoch	learning rate	weight decay	batch size	optimizer
20	2e-5	0.01	16	Adam

Table 4: Training configurations

		beam search		**temperature**		
	greedy	5	10	0.5	0.7	0.9
rouge-1	21.21	**23.47**	23.35	20.10	17.91	14.06
rouge-2	8.05	9.49	**9.55**	7.24	6.21	4.36
rouge-L	20.56	**21.12**	21.02	18.97	16.70	13.02
	top-k			**top-p**		
	25	50	100	0.80	0.92	0.95
rouge-1	17.67	16.40	15.51	14.52	13.12	12.50
rouge-2	5.71	5.00	4.80	4.65	4.00	3.74
rouge-L	16.48	15.20	14.34	13.43	12.10	11.60

Table 5: Rouge scores of different generation strategies