AI6127: Assignment 2

Report Submission Due: 23 Apr, 2024 (before 11:59 pm)

Submission: Submit to NTULearn (Assignments → Assignment 2 → Report) with subject "AI6127-Ass2-YourStudentID".

Topic: Seq2Seq model for machine translation

In this assignment, we will implement a machine translation model using seq2seq architecture.

Example code base https://colab.research.google.com/drive/1rfBv2y-Zz016KhVeOo15IHjx9B5JR4RK?usp=sharing

In the code base, you are given a sample seq2seq model where Encoder and Decoder are simple GRUs.

Tasks:

1. Warm up: Read, understand, and reimplement the examples in the code base
2. Run the example code base and record the Rouge scores for test set (Rouge 1 and Rouge 2)
3. Change the GRU in Encoder and Decoder in the code base with LSTM, run the code, and record the Rouge scores for test set
4. Change the GRU in Encoder (not Decoder) in the code base with bi-LSTM, run the code, and record the Rouge scores for test set
5. Add the attention mechanism between Encoder and Decoder in the original code base (you can refer to Lecture 8 for attention mechanism), run the code, and record the Rouge scores for test set
6. Change the GRU in Encoder (not Decoder) in the original code base with Transformer Encoder, run the code, and record the Rouge scores for test set.

   You can refer to the next page for the documents and tutorials of using Transformer and how to use it as Encoder

Report:

- Summarize the results of experiments (better in tables)
- You should add the analysis, comparison, and explanation about and findings in the results
- The format is free style. Try to be concise and must not more than 4 pages (excluding the cover page if have)
- The deadline is firm at **11:59pm, 23 Apr 2024**

Documents and tutorial of using Transformers

Document for Transformer Pytorch:

TransformerEncoder — PyTorch 1.13 documentation

Example of using a Transformer Pytorch:

https://pytorch.org/tutorials/beginner/transformer_tutorial.html

https://towardsdatascience.com/a-detailed-guide-to-pytorchs-nn-transformer-module-c80afbc9ffb1

Note: You should take the mean of all hidden representation output from the transformer encoder of each token to be a sentence representation of the encoder.

When using a transformer encoder, the dimensions may differ for RNN (or LSTM). Also, for the transformer encoder, you must input the whole sentence instead of feeding word by word to get the next token representation.

```
def train(input_tensor, target_tensor, encoder, decoder,
encoder_optimizer, decoder_optimizer, criterion,
max_length=MAX_LENGTH):
    ...

    for ei in range(input_length):
        encoder_output, encoder_hidden = encoder(
            input_tensor[ei], encoder_hidden)
        encoder_outputs[ei] = encoder_output[0, 0]
    ...
```