Assignment (Optional)

CS-583: Deep Learning

Fall 2023

In this assignment you will implement beam search algorithm and apply them to a real-world dataset. You would need to install the `sacrebleu` package.

1 Neural Machine Translation

Details

The task is to implement beam search for Neural Machine Translation (NMT). This NMT model is already trained on the French to English translation task and the code is provided to run for few examples. The data files have already been pre-processed including tokenization and normalization.
The following files already been provided:

- *models/encoder* & *models/decoder*: trained model files
- *data-bin/fra.data* & *data-bin/eng.data*: vocabulary files created using the training data
- *data/valid.fra* & *data/valid.eng*: normalized and tokenized validation files
- *data/test.fra* & *data/test.eng*: normalized and tokenized test files
- *data_utils.py* & *rnn.py*: supporting files for data processing and model initialization
- *beam_search.py*: Main file to preprocess, train, translate input sentences. **This is the file you will be updating.**

Part 1. Compute BLEU score for *test.fra* file **[5 pts]**

Setup all the files and data in Google Colab and run the file *beam_search.py* to read and translate the *test.fra* file and save the output in a new file named *test_beam_1.out*. You can update the `translate_test` function in the *beam_search.py* file on `Line 92`. A `translate_example` function is given as an example. Finally, use sacrebleu package to compute the BLEU Score[1] performance of the model. You should use the following command and report the BLEU scores.

```
sacrebleu data/test.eng < test_beam_1.out
```

Part 2. Implement beam search **[20 pts]**

Implement the beam search algorithm. You can update the `beam_search` function in the *beam_search.py* file on `Line 98`. The function should be generic to use any beam size.

Part 3. Tune beam size on *valid.fra* & plot BLEU scores **[10 pts]**

Apply beam search on the *valid.fra* file and get output files for beam size $K = 1, 2, \cdots, 20$. Use `sacrebleu` command to compute the BLEU scores for each of the output files. Plot the validation data BLEU scores for corresponding beam sizes.

Part 4. Compute BLEU score for *test.fra* file using optimal beam size **[5 pts]**

Apply the beam search on the *test.fra* file using the K which gives the highest BLEU score in part 3. Use `sacrebleu` command to compute the BLEU score for the test translations.

Note:

1. You should select GPU in Google colab from Runtime > Change Runtime Type > Hardware Accelerator
2. You can install sacrebleu using the command `pip install sacrebleu`
3. Use the '!' for all the commands in Google Colab, e.g. `!pip install ...` or `!python ...`
4. You should not change the order of the files. The correct test translation (*test.eng*) should always be before the output translations.
   `sacrebleu correct_data < output_data`

Submission

This is an optional individual assignment. Each person should submit as a single zip file named with 'HW_' followed by the username (e.g. *HW_akhan4.zip*). The zip file should contain the following:

- The updated `beam\_search.py` code file
- BLEU Score plot on validation data as `png`
- A Readme file containing:
  - (Part 1) Test data BLEU score without beam search implementation
  - (Part 4) Test data BLEU score on the optimal beam size

Remember that after general discussions with others, you are required to work out the problems by yourself. All submitted work must be your own, though you can get help with others, so long as you cite the help. Please refer to the Stevens Honor System for clarifications.

References

[1] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.