MSAI-437, Winter 2024
Homework #4: RNN Language Model
Due Date: Friday, March 8th @ 11:59PM
Total Points: 9.0 (plus optional 2.0 bonus point)

In this assignment, you will work with your group to implement and train a RNN language model on the Wikitext-2 corpus **using PyTorch.** You may discuss the homework with other groups, but do not take any written record from the discussions. Also, do not copy any source code from the Web.

Guidelines

- You should recall that the language modeling task is to predict the probability of the next token based upon the historical context, such that $P(w_t \mid w_{0:t-1})$ and that multi-class cross entropy is the most commonly used objective function.
- You should recall that perplexity is the most common evaluation metric for language models.
- You should use white-space tokenization (*e.g.,* tokens are separated by a blank space -- ASCII 32) to parse your corpus into tokens. Note: `tokens = text.split(' ')` is a common approach.
- You may reduce the vocabulary size to match computational resources. This is often accomplished by replacing tokens with frequency of less than *threshold* with the token `<unk>`. Wikitext-2 has a vocabulary size of approximately ~33K, and it is not unreasonable to reduce this to ~10K.
- You should create an integer representation of the corpus that serve as indices for the embedding look-up tables. Note: Python dictionaries are helpful for mapping string tokens to an integer value.
- You should use `nn.Embedding()` to create a look-up table of embeddings.
- You are encouraged to the `torch.nn.RNN()` module.
- Your initial RNN should be only one layer deep.
- You should use 100 as the initial dimensionality of your embedding space.
- Your RNN should use dropout, treating probability of dropout as a hyper-parameter.
- You should treat the number of unrolled time steps as a hyper-parameter in the range [10,30].
- You must remember to "recycle" your hidden state over time steps, such that you h_t for the start of the current time step is the final h_t from the prior time step.
- You are encouraged to experiment with batch size to match your computational resources.
- You should train your RNN language model for 20 epochs.
- You are encouraged to experiment with learning rates and optimization algorithms (e.g., `torch.optim.Adam()`) such that you model is close to convergence at 20 epochs.
- You should expect a final test set perplexity at or below 250.
- You may find it helpful to use a random number seed for reproducibility when debugging.
- You may want save your model weights.
- You can complete the assignment entirely as a Jupyter Notebook, or you can turn in a single Python file with an accompanying PDF of the results.

Steps to complete the homework

1. (7.0 points) Implement and train your RNN language model on Wikitext-2 corpus using the guidelines specified above.
    a. provide a description (or illustration) of you architecture and discuss design choices,
    b. list hyper-parameters used by you model an discuss how you selected these values,
    c. provide learning curves of perplexity vs. epoch on the training and validation sets, and
    d. provide final test set perplexity.

2. (2.0 points) Discuss how you might improve this "vanilla" RNN language modeling architecture.

3. (2.0 Bonus Points) Implement one (or more) of the improvements mentioned above, and provide a new set of learning curves and final test perplexity.

Submission Instructions

Turn in your homework as a single zip file, in Canvas. Specifically:

1. Create a single Jupyter Notebook or a single PDF file with the answers to the questions above, and your graphs.

2a. Create a single ZIP file containing:

- `homework2.pdf`, and
- Your `.py` code file or Juptyer Notebook

3. Turn the zip or ipynb file in under Homework #2 in Canvas.

Good luck, and have fun!