MSAI-337: Natural Language Processing
Group Project #2

Winter 2021

Description: For this project, your group will implement a LSTM language model in Python using either PyTorch (preferred) or Keras deep learning platforms. Your language model should conform to the following specifications:

a. use dropout at the input layer and within LSTM cells,
b. support gradient clipping,
c. support a configurable number of layers, time step sizes and batch sizes,
d. maintain the hidden state h_t between time steps,
e. explicitly calculate numerators and denominators for probability calculations (optional, but preferred),
f. provide the ability to save and recall the model,
g. report perplexity on training, validation and test sets,
h. train via stochastic gradient descent using a learning rate that decays at a constant geometric rate, and
i. support configurable hyper-parameters including: (i) embedding space dimensionality, (ii) vocabulary frequency threshold, (iii) a keep probability for dropout, (iv) an initialization range, (v) number of training epochs, (vi) an initial learning rate and decay schedule, (vii) number of layers, (viii) batch size, (ix) step size, (x) maximum gradient, (xi) embedding tying, (xii) a bias flag, and any others that you think may be necessary.

You are encouraged to follow the architecture presented in Recurrent Neural Network Regularization (https://arxiv.org/pdf/1409.2329.pdf) or the implementation provided in `zaremba.py` (attached), although you are free to use any architecture. Hyper-parameters can be hard-coded (please group this in a single location in the code) or obtained by parsing from the command line using `argparse` (or something similar). One of the most important parts of implementing an LSTM language model is being mindful of how the data is fed into the model. The corpus handling functions from Project #1 should be helpful.

Tasks: Please perform the following tasks using your LSTM language model:

1. (2.0 pts) Train you LSTM language model on the Wikitext-2 corpus (attached) for 20 epochs using the hyper-parameters parameters specified `zaremba.py`. This should take less than one hour on a commodity-grade GPU. Final test set perplexity should be roughly 130. You can use `zaremba.py` to benchmark your code (please note that this language model was developed in TensorFlow 1.0).

2. (5.0 pts) Train your LSTM language model on the corpus prepared for Project #1. You may want to experiment with hyper-parameter setting to obtain a model that performs reasonably well. Optimal hyper-parameter settings with be unique to each corpus. Note: Hyper-parameter tuning should not be exhaustive (maybe limited to 10 trials). Save your final model for future use. Note: It is a good practice to save the code and results along with the binary image of your model. Present learning curves for training and validation set perplexities in a single graph. Also report final test set perplexity.

3. (3.0 pts) Prepare a short write-up (about two pages, not counting result tables) including:
   a. a short description of you model,
   b. final hyper-parameters used to train your model and any observations made while selecting these,
   c. interpret results obtained on your corpus relative to the Wikitext-2 corpus, and
   d. interpret results obtain on your corpus using your LSTM language model relative to SRILM.

What to Turn In: Your submission should include (i) the Python code for you LSTM language model, preferably in a single file and (ii) a PDF of your write-up. All of these files should be submitted as a single zip file.