Assignment 3: Language Modeling

Homework assignments will be done individually: Each student must hand in their own answers. Use of partial or entire solutions obtained from others or online is strictly prohibited. Electronic submission on Canvas is mandatory.

You will need to use the Penn Treebank corpus for this assignment. Four data files are provided: train.txt, train.5k.txt, valid.txt, and input.txt. You can use train.txt to train your models and use valid.txt for testing. File input.txt can be used for a sanity check on whether the model produces coherent sequences of words for unseen data with no next word.

1. **N-gram** (55 points)
   (a) (10 pts) Preprocess the train and validation data, build the vocabulary, tokenize, etc.
   (b) (10 pts) Implement an N-gram model (bigram or trigram) for language modeling.
   (c) (10 pts) Implement Good Turing smoothing.
   (d) (10 pts) Implement Kneser-Ney Smoothing using:

$$P_{\text{KN}}(w_i|w_{i-1}) = \frac{\max(c(w_{i-1}, w_i) - d, 0)}{c(w_{i-1})} + \lambda(w_{i-1}) P_{\text{CONTINUATION}}(w_i)$$

   where

$$\lambda(w_{i-1}) = \frac{d}{c(w_{i-1})} |\{w : c(w_{i-1}, w) > 0\}|$$

$$P_{\text{CONTINUATION}}(w) = \frac{|w_{i-1} : c(w_{i-1}, w) > 0|}{\sum_{w'} |\{w'_{i-1} : c(w'_{i-1}, w') > 0\}|}$$

   (e) (5 pts) Predict the next word in the valid set using a sliding window. Report the perplexity scores of N-gram, Good Turing, and Kneser-Ney on the test set.
   (f) (10 pts) There are 3124 examples in input.txt. Choose the first 30 lines and print the predictions of next words using your N-gram model.

2. **RNN** (45 points)
   (a) (5 pts) Initialize parameters for the model.
   (b) (10 pts) Implement the forward pass for the model. Use an embedding layer as the first layer of your network (e.g. `tf.nn.embedding_lookup`). Use a recurrent neural network cell (GRU or LSTM) as the next layer. Given a sequence of words, predict the next word.
   (c) (5 pts) Calculate the loss of the model (sequence cross-entropy loss is suggested) e.g. `tf.contrib.seq2seq.sequence_loss`.
   (d) (5 pts) Set up the training step: use a learning rate of $1e-3$ and an Adam optimizer. Set window size to be 20 and batch size to be about 50.
   (e) (10 pts) Train your RNN model. Calcuate the model's perplexity on the test set. Prove that perplexity is $\exp\left(\frac{\text{total loss}}{\text{number of predictions}}\right)$.

(f) (10 pts) Print the predictions of next words in the same 30 lines of input.txt as in N-gram.

Submission Instructions You shall submit a zip file named Assignment3_LastName_FirstName.zip which contains:

- python files (.ipynb or .py) including all the code, plots and results. You need to provide detailed comments in English.