TTIC 31230 Fundamentals of Deep Learning
Problem set 4
Due Thursday 11:59 pm, February 2

- Zip all your ipynb&pdf&model file with name PS4_yourfullname to: ttic.dl.win.2017@gmail.com.
- Late Submission: submitting late work will be penalized 10% per day, maximum three days delay allowed, no submission allowed after that.

This problem set involves writing an LSTM character language model in EDF.

The model will be trained on the Penn Tree Bank (PTB). Each training instance is a sentence where sentences come in different length. The provided code (in the notebook) divides the training into minibatches where each minibatch containes sentences of similar length. A new model is to be built for each minibatch where the length of the model (the number of time steps) is different for different minibatches. You are to write the prodecure `BuildModel()`.

The data for a given minibatch will be contained in computation node (instance) `inp` where `inp.value` has shape (B, T) where B is the minibatch size and T is the length of the sentence data in the minibatch. `inp.value`$[b, t]$ is an integer number from 0 to 50 inclusive where 1∼50 is the index of characters and where 0 is padding. for every sentence, we add { and } to represent the beginning of sentences and termination of sentences, and { and } are treated like normal characters, you will have to predict sentence termination symbol }.

The integer 0 is used for padding the end shorter sentences to make them match the sentence length of the batch. However , it is convenient to ignore this and treat zero as just another character. The training and testing code does not score predictions for padded characters.

Your procedure `BuildModel()` should construct the computation graph for an LSTM character language model and return two variables, one is the loss and another one is a list named "Prob" with length T, where `Prob[t]` is a EDF instance and its value has shape `(B,51)` and where $Prob[t].value[b, k]$ is the probability that `inp.value`$[b, t] = k$ based on the input sequence up to time $t - 1$.

Note that you must produce a probability distribution over the first character based on no information (that's why we artificially add { before sentences). Your parameters should include initial carry and hidden state vectors from which to predict the first character.

EDF for this problem set include a class `ConCat` for concatenating feature vectors and also several others such as `ArgMax`, `Embed`, `MeanwithMask`, `GradClip` etc.

You should one-hot character vectors and then project your one-hot representation to a vector with 200 dimension. The carry and hidden state vectors of your model should also have dimension 200. You should initialize all weight vectors with Xavier initialization. Of course the parameters should be initialized once and then used in all your models and in each time step of each model. You can add bias in LSTM or not.

You should achieve a character-level log loss around 1.2 in 20 epochs. This takes about $3 \sim 6$ hours to train, depends on your implementation and your hardware. The given code provides the ability to store a model after partial training and restarting the training from a stored model.

Please fill the corresponding functions and your code should be runnable, and please also submit your trained model, thus we can check it easily.

The following link is a good reference: http://karpathy.github.io/2015/05/21/rnn-effectiveness/