Mohana Kumar Poongavanam
poongava@usc.edu

CSCI 544 - Spring 2019 - DL Assignment 2 Report

Purpose of the doc:

- This doc is written in connection with the final deep learning assignment of CSCI 544 Spring 2019 (Link). I've tried to document the structure of my deep learning model, the hyperparams used, the experiments that were run in arriving at those. I've also added some hyperlinks to the tensorflow functions used, for reference.

Problem Statement:

- POS tagger using deep learning.

Libraries:

- Tensorflow, Numpy

Datasets:

- Italian, Japanese for training and development.
- Tested on a secret language.

Dataset Pre-processing:

- Terms and tags are converted to distinct integers and matrices were formulated.
- Zero padding was incorporated to handle different sequence lengths.

Experiments and Model:

Phase 1:

- The pre-processed data was passed on to an embedding Layer.
    - tf.nn.embedding_lookup
- The obvious choice for this problem is to go ahead with some form of Recurrent Neural Network(RNN). As the POS tags of words are heavily dependent on the context in which they occur I chose to choose to use an LSTM cell with hidden units.
    - tf.contrib.rnn.LSTMCell
    - tf.nn.dynamic_rnn
- In order to generalize the model, dropout layers were added on to the output of the LSTM cells.
    - tf.contrib.rnn.DropoutWrapper

- A dense layer was added finally with the number of output units same as the number of tags to represent the each of the tag for each word.
    - tf.layers.dense
- CRF(Conditional Random Fields) log likelihood was used to compute the loss, as they were highly recommended for problems dealing with ordered data.
    - tf.contrib.crf.crf_log_likelihood
- AdamOptimizer was put in place.
    - tf.train.AdamOptimizer
- With a fixed learning rate over all epochs and decent hyper-params(batch size=32, lr = 1e-7 as with the default params given with the assignment) the above model gave around *~82% average accuracy for Italian and Japanese languages*.

Phase 2:

- Significant changes were made to the model layers and losses.
- LSTM was replaced with bidirectional LSTM so as to include both previous and future contexts in predicting the POS tags of the words.
    - tf.nn.bidirectional_dynamic_rnn
- Dropout wrappers were added to both input and output of the LSTM cells.
- CRF loss was replaced with sequence to sequence loss as suggested with the description of the assignment.
    - tf.contrib.seq2seq.sequence_loss
- Different hyper param for embedding dimensions(~128) was chosen specially for Japanese language. Italian / any secret language had much lower dimensions(~50) which had helped me run more epochs on Italian due to its size.
- Batch size was decreased and kept low to be around 25.
- Learning rate was fixed at ~0.007. Time based decay to learning rate was incorporated into the model which gave the best boost to the accuracies. i.e., lr = lr / (1 + decay_rate * epoch_cnt)
    - Related material
- The above changes to the models gave me *>95% accuracy.*

Other alternatives tried out, but didn't give better accuracies:

- Learning rate decay:
    - Exponential learning decay: lr = lr * e^(-k * epoch_cnt)
    - lr = lr / 2 (for each epoch) - This was used in DL assignment 1
- Losses:
    - Softmax cross entry loss: Tf.nn.softmax_cross_entropy_with_logits
- Size of data / network:
    - Batch size > ~100, Hidden units = ~256
- Gradient clipping
    - Unfortunately/Surprisingly this didn't help me improve the accuracy.