Deep Learning Report 3

Anup Patel (Sr.No. - 15474)
M.tech CSA

May 15, 2020

Objective and Task

The aim of the project is to implement a deep learning network for the task of Natural Language Inference. Given two sentences, we will have to predict if the sentence pair constitutes entailment/contradiction/neutral. Following are the tasks to be performed.

1. Build a simple Logistic regression classifier using TF-IDF features.
2. Build a Deep model specific for text like RNN/GRU/LSTM or any of the new approaches like Transformers, BERT etc for NLI.

SNLI Dataset

The dataset contains 550,152 training pairs, 10,988 validation pair and 10,000 testing pairs. Few training pairs and validation pairs do not have gold_label.

tfidf and Logistic Regression

For preprocessing the text data, i had used nltk to remove stop words. I had used *spacy* as a tokenizer. For tfidf, i had used *tfidfvectorizer* from Sklearn library. I had cancatenated premises and hypothesis and then computed tfidf. tfidf value depend on number of document so it is not a good idea to compute idf on validation set and test set. I had use *fit_and_transform* on train set whereas *transform* on validation and test set.
After preprocessing, i used logistic regression from sklearn library. Tried with different values of C. max iter set to 500. Since sentence lengths are not too long, there might be chance that stop words can convey good information. Without removing stop word there is increase of accuracy of around 2 percent.

Results

Train Accuracy (With removing stopword):: 57 %
Test Accuracy(With removing stopword):: 56 %

Train Accuracy (Without removing stopword):: 60 %
Test Accuracy(Without removing stopword):: 59 %

Deep Model

Second problem is to build a Deep model specific for text like RNN/GRU/LSTM or any of the new approaches like Transformers, BERT etc for NLI.

Preprocessing step is similar to logistic regression part but i had not removed stop word in this part based on good result in logistic regression. I had used Bi direction LSTM model to solve this problem using *pytorch* library. My LSTM model has an embedding layer, then a layer LSTM and then sequential layer.
In forward pass, i had computed embedding of premise and hypothesis, pass it to LSTM layer and then cancatenate them.

Loss function :: Cross Entropy (because it was a classification task)

I had set batch size to 64 for all train, test and validation set.

I had tried various learning rate but got best result with learning rate = 0.001.
I had also tried with various embedding dimension i.e 128,256,300,512. I got best result with embedding dimension of 300.

Model starts getting overfit after 26 iteration.

I had also tried various dropout ratio : 0, 0.1,0.3,0.01 and got best result with dropout ration = 0.01

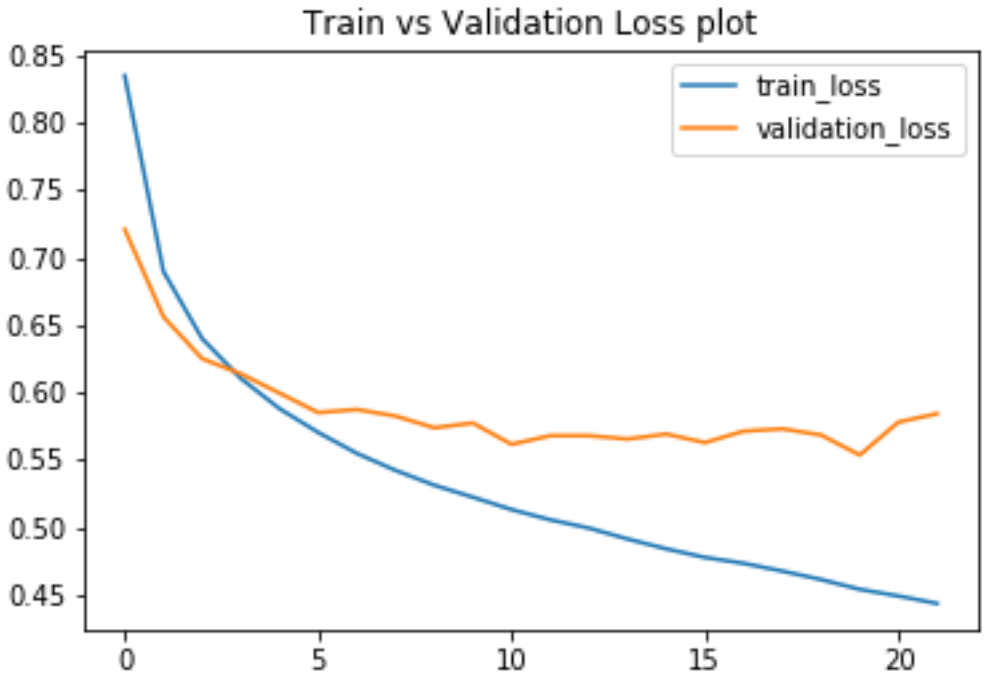

Figure 1: Train, Test loss vs Epoch plot for LSTM Model

Results

Train Accuracy :: 85 %
Validation Accuracy:: 83 %
Test Accuracy:: 78.9 %
Test loss:: 0.55