Praneel Trivedy

Professor Yin

Foundations of Machine Learning

06 December 2022

System Description

I began this project by converting the 3 data files from text to csv. I opened the data into Microsoft Excel and labeled the columns so I could apply preprocessing steps in a more familiar manner.

The first step of my research was to understand how to structure my data so that it could be preprocessed and inputted to the neural network. The relevant packages used included: *Numpy, Pandas, Matplotlib, Re, String, NLTK, SKLearn, and CountVectorizer*. I initiated these at the beginning of my code so I could always reference back to them. After these libraries were initiated, I loaded the training and development sets into jupyter notebook. I glanced at the data to ensure it loaded correctly using head().

After a series of trial and error, I cleaned the code to create a function for preprocessing the sentences. The key factors I addressed was the removal of punctuation and stopwords. These two features made it significantly easier to find clarity within the instances. I appended two new columns to the original train and development data frames with the preprocessed sentences.

The next step was to implement CountVectorizer() to prepare a bag of words representation for the data. I experimented with unigrams, bigrams, and trigrams. I observed the dimensionality of my resulting dataframe increased significantly as n increased. The resulting data frame for unigrams consisted of X rows/instances from the original dataset and the number of unique n-grams created using the CountVectorizer() function.

For both the training and test data, I merged the two data frames resulting from the bag of words representation. The rows represented the number of sentences. The columns represented the number of unique words in the both sentence 1 and 2. These were large and sparse matrices because of the high dimensionality.

In terms of the architecture for the neural network, we used MLP with an input layer, 2 hidden layers, and an output layer. The hidden layers transformed from 784 perceptrons to 10.