Assignment #8 Write-Up
John Moderwell
MSDS 422

Summary

This assignment looks at building language models with pre-trained word vectors. The objective was to evaluate movie sentiment by studying the effects of word vector size, vocabulary size, and neural network structure (hyperparameters) on classification performance. Students were provided with jumpstart code and asked to revise it by incorporating different word embeddings and language sizes. The goal was to to choose the best model based on classification accuracy and make recommendations to management based on the results.

Research Design

Like previous assignments, one of the objectives was to study the effect of different hyperparameters on each model. Models 1 and 2 looked at the difference between the default single layered RNN and a triple layered RNN. Models 3,4 and 5 studied the effect of adding a dropout layer to the model and increasing the learning rate. A unique objective in this assignment was to test these models on different word vectors as well as vocabulary sizes. The word vectors in this assignment utilized GloVe (global vectors) . According to the Canvas assignment page, “These numeric vectors or neural network embeddings capture the meaning of words as well as their common usage as parts of speech.” Each word is replaced by a vector of numbers to represent that word. For this assignment, two different vectors were used: 50 dimensional vector and a 100 dimensional vector. For models 1 and 2, the 50 dimensional vector was used with a 10,000 vocabulary size. For models 3, 4 and 5, the 100 dimensional vector was used with a vocabulary size of 40,000.

Assignment #8 Write-Up
John Moderwell
MSDS 422

Programming Overview

The entire project was done in Python using the Jupyter Notebook environment. Packages that were used included pandas and numpy for data handling and manipulation. Tensorflow was used to handle the machine learning aspects of the assignment. Other packages that were included in the jumpstart code and aided in the ingestion of data included: chakin, nltk, collections, re as well as various parts of _future_ (absolute import, division and print function).

Recommendations

All models shared a few of the same basic hyperparameters: batch size = 100, optimizer = Adam, number of epochs = 50, number of neurons = 20 and number of outputs = 2. Based on the results, it is apparent that Model 4 performed best. It averaged 100% training accuracy and 75.5% testing accuracy. Model 4 had 3 layers and a dropout layer. It was trained using the 100 dimensional embedding and a vocabulary size of 40,000. The worst performer was Model 1 which resulted in an average of 80% training accuracy and 68% testing accuracy. It was trained on the 50 dimensional embedding and vocabulary size of 10,000. There are a few conclusions that can be drawn from these results. It is apparent that adding a dropout layer and increasing the number of layers has a positive effect on model performance. It was also apparent that increasing the learning rate to a certain extent also has a positive effect.

As far as these results concern management, I would say that the the type of word vector and the size of the vocabulary included in the model have a large impact on the performance. Movie reviews can be similar to customer messages and so being able to identify critical words is very important. Because language is so complex and can take many forms, data scientists can

improve their models by constantly updating word vector types as well as increasing vocabulary size.