2018FA_MSDS_422-DL_SEC58

Practical Machine Learning Assignment 8

Prabhat Thakur

1. Problem Definition

This assignment involves creating **recurrent neural networks** (RNN) model using TensorFlow package for Language Modeling to predict movie review sentiment (thumbs-up versus thumbs-down). We will study effects of word vector size, vocabulary size, and neural network structure (hyperparameters) on classification performance.

2. Research Design and methods

For this assignment , we will be using pre-trained word vectors GloVe (global vectors) and subsets of pre-trained word vectors to speed up the training process. We will train language models using two pre-trained word vectors and two vocabulary sizes representing the cells of a completely-crossed 2-by-2 experimental design, defining four distinct language models and compute their classification accuracy in the test set.

There are 500 text files with positive movie reviews and 500 text files with negative movie reviews. We use sentences (sequences of words) to train language models for predicting movie review sentiment (thumbs-up versus thumbs-down).

3. Implementation and Programming:

The first step is to install chakin package and download GloVe **GloVe.6B** embeddings with 50,100,200, and 300 dimensions. For this experiment we used glove.6B.50d.txt and glove.6B.100d.txt embeddings of dimension 50 and 100 respectively. The embeddings are loaded to create dictionary data structures with vocabulary of 10 thousand and 50 thousand words for the language models to reduce the size of the vocabulary to the n most frequently used words.

Next we read data for negative movie reviews and positive movie reviews files. Data is then stored in a list of lists where the each list represents a document and document is a list of words. Then we convert positive/negative documents into numpy array note and only first 20 and last 20 words of each review is used as our word sequences for analysis which construct list of 1000 lists with 40 words in each list. Then we run these words through the vocabulary list to create list of lists of lists for embeddings. Next we make embeddings a numpy array for use in an RNN. Also created training and test sets with Scikit Learn package. Now we build and train out models one by one.

Model 1: Vocabulary size 10000 and glove.6B.50d.txt

Model 2: Vocabulary size 10000 and glove.6B.100d.tx

Model 3: Vocabulary size 50000 and glove.6B.50d.txt

Model 4: Vocabulary size 50000 and glove.6B.100d.txt

For all above models we used basic cell RNN (BasicRNNCell). One more model is built and trained Model 5: Vocabulary size 50000 and glove.6B.100d.txt using GRU cell (GRUCell).

4. Findings and recommendations

Out of 5 models we built, model 5 performs best with classification accuracy on test dataset about 80%. It is observed that large vocabulary size and embedding dimension **improvs the prediction accuracy but at the cost of memory and processing time**. Also compared to basic cell, GRU cell perform much better for RNNs.

Recommendation : The management should capture past data on written customer reviews, call, and complaint logs and their classification to build and train RNN language models one for each category. A model with moderate embedding dimension (100 to 200) and vocabulary size (50k to 100K) and GRU cells would be a good choice for the predictive models.

Once the models are trained and deployed, any new customer reviews, call, and complaint logs can be feed to the respective model to predict its importance automatically and can also be used to assign priority for customer service agents to start working on them.