Sentiment Analysis with Neural Networks

REVIEW | HISTORY

Meets Specifications

Your project meets all the requirements, well done! 👍

You showed a good understanding of the concepts of this project and implemented all the functions perfectly. We are looking forward to your next submissions in the nanodegree! :-)

Importing Twits

✓ **Print the number of twits in the dataset.**

```
Number of twits:  1548010
```

Perfect!

Preprocessing the Data

✓ **The function `preprocess` correctly lowercases, removes URLs, removes ticker symbols, removes punctuation, tokenizes, and removes any single character tokens.**

The preprocess function is correctly implemented, good work!

✓ **Preprocess all the twits into the `tokenized` variable.**

A list comprehension does the job 👌

✓ **Create a bag of words using the tokenized data.**

You created a bag of words using Python's `Counter`, well done!

✓ **Remove most common and rare words by defining the following variables: `freqs`, `low_cotoff`, `high_cutoff`, `K_most_common`.**

The most and least common words were removed 👌

✓ **Defining the variables : 'vacab', 'id2vocab' and 'filtered' correctly.**

Perfectly implemented 👍

Neural Network

✓ **The init function correctly initializes the following parameters: `self.vocab_size`, `self.embed_size`, `self.lstm_size`, `self.lstm_layers`, `self.dropout`, `self.embedding`, `self.lstm`, and `self.fc`.**

The neural network is set up correctly!

✓ **The 'init_hidden' function generates a hidden state**

All weights are initialized to zero, well done!

✓ **The 'forward' function performs a forward pass of the model the parameter input using the hidden state.**

Perfect implementation of the forward pass!

Training

✓ **Correctly split the data into `train_features`, `valid_features`, `train_labels`, and `valid_labels`.**

You split the data into train and validation sets. 80%/20% is a good split ratio.

✓ **Train your model with dropout and clip the gradient. Print out the training progress with the loss and accuracy.**

The training loop is implemented according to the requirements, great work!

Making Predictions

✓ **The `predict` function correctly prints out the prediction vector from the trained model.**

```
tensor([[ 0.0015,  0.0279,  0.0096,  0.8286,  0.1324]])
```

The function returns the prediction vector 👌

✓ **Answer what the prediction of the model is and the uncertainty of the prediction.**

Excellent interpretation of the prediction of the model!