ROOT2AI Technology Private Limited

Artificial Neural Networks internship

Introduction

- I have used Recurrent Neural Networks which is capable of processing sequential data such as text or characters.
- Since the given dataset contains many words that follow in a very specific and meaningful order we need to be able to keep track of each word and when it occurs in the data.This is why I treated text as a sequence and processed one word at a time.
- I kept track of where each of these words appear and used that information to try to understand the meaning of pieces of text.
- cleaning text is performed using stemming
- Created bag of words.
- To make sure that different lengths of data is not passed into neural network, I have padded my sequences.

Model Interpretation:

- **Count Vectorizing**:To convert a collection of text documents to a matrix of token counts.
- **train_test_split:**The dataset was splitted into train and test parts.
- I have used keras.datasets.imdb.load_data () to load the dataset in a format that is ready for use in neural network and deep learning models. The words have been replaced by integers that indicate the absolute popularity of the word in the dataset.

- **creating a model and training a model:**

. I have used word embedding layer as the first layer in our model and added a LSTM layer afterwards that feeds into a dense node to get our predicted target.(32 stands for output dimensions of the vectors generated by the embedding layer).

Later I compiled and trained the model(where epochs=10,batch_size=64).

Predictions:

- I used the above model to obtain the predications for y_test.
- And I used Confusion matrix and accuracy score to obtain the accuracy of my model.

Train & test accuracy score:

- The final accuracy after 10 epochs was 86.732
- the confusion matrix was: array([[10668, 1485],
  [1832, 11015]], dtype=int64)

Limitation of the model:

- Difficult to preprocess.
- Vanishing gradient might be one of the problem if we have used more layers, so it will be best to add Dropout after each layer.

(Initially I tried with naive Bayes algorithm but the accuracy I got was 68%, which was not what I was looking for, so I decided to go with the above approach Although many a times I faced with overfitting problem(my accuracy went from 0.90 to 0.98 in just 2 epochs) ,I was able to resolve it and now my accuracy is around 0.86).

THANK YOU!