Deep Learning

Assignment 5

Word Embedding

Aim

To develop a neural network model to perform sentiment classification on the IMDB dataset with both trainable and loaded embedding layer and to test the accuracy of each model. Finally an LSTM layer is added to the model with good accuracy to make it a further better model.

Problem Statement

The IMDB dataset consists of the reviews of movies. These reviews are to be classified based on the terms used in the reviews by a neural network model.

Proposed Solution

Dataset

The IMDB dataset has 50000 reviews on movies. It includes all sorts of reviews that can be used for Natural Language Processing. This dataset is loaded through keras. 25000 reviews are loaded into the training module and the rest into the testing model. The training model is further split into training and validation module for better accuracies.

Embedding layer

An embedding layer is created to which the dataset is passed. The dimensions of the embedding layer matrices are specified. This layer creates a unique 2D vector for every word in the given input. By this way it makes the process of language processing easier.

Dense Layer

Finally dense layers are attached to the model with relu activation. Since the output of the embedding layer has to be passed to the dense layer, the outputs of the embedding layer is flattened. Finally the model is built with the optimizer RMSProp and binary cross entropy loss and compiled.

Data preprocessing

The data was preprocessed by splitting it into test, train and validation datasets. Then the labels were converted to one hot encodings. Since the input datasets could be of various string lengths they were padded with zeroes for a definite length.

Glove embedding

The Glove embedding layer was used for the loaded embedding. GloVe stands for global vectors for word representation. It is an unsupervised learning algorithm for generating word embedding by aggregating global word-word co-occurrence matrix from a corpus. The resulting embedding show interesting linear substructures of the word in vector space.

LSTM layer

It stands for Long Short Term Memory. It consists of feedback connections that help it to remember the sequences of input. Thus it can give a better accuracy than other deep learning models.

Evaluation

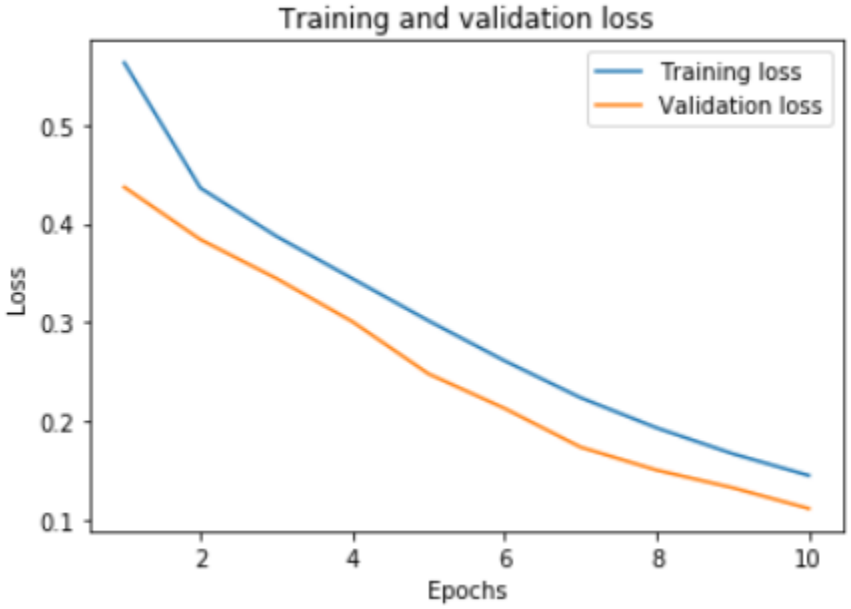
Trainable Embedding

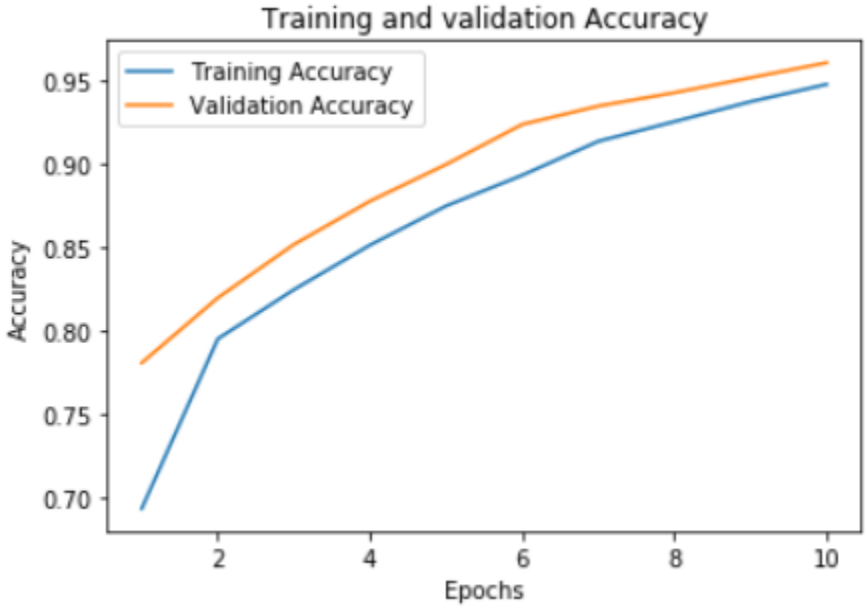
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 20, 8)             80000
_________________________________________________________________
flatten_1 (Flatten)          (None, 160)               0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               82432
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 513
=================================================================
Total params: 162,945
Trainable params: 162,945
Non-trainable params: 0
_________________________________________________________________
```

Model Fitting

```
Epoch 6/10
25000/25000 [==============================] - 5s 202us/step - loss: 0.261
0 - acc: 0.8935 - val_loss: 0.2128 - val_acc: 0.9240
Epoch 7/10
```

```
25000/25000 [==============================] - 5s 199us/step - loss: 0.223
6 - acc: 0.9139 - val_loss: 0.1734 - val_acc: 0.9350
Epoch 8/10
25000/25000 [==============================] - 5s 196us/step - loss: 0.193
0 - acc: 0.9257 - val_loss: 0.1502 - val_acc: 0.9430
Epoch 9/10
25000/25000 [==============================] - 5s 213us/step - loss: 0.166
8 - acc: 0.9376 - val_loss: 0.1323 - val_acc: 0.9520
Epoch 10/10
25000/25000 [==============================] - 5s 208us/step - loss: 0.144
7 - acc: 0.9480 - val_loss: 0.1112 - val_acc: 0.9610
```





Test accuracy: 71%

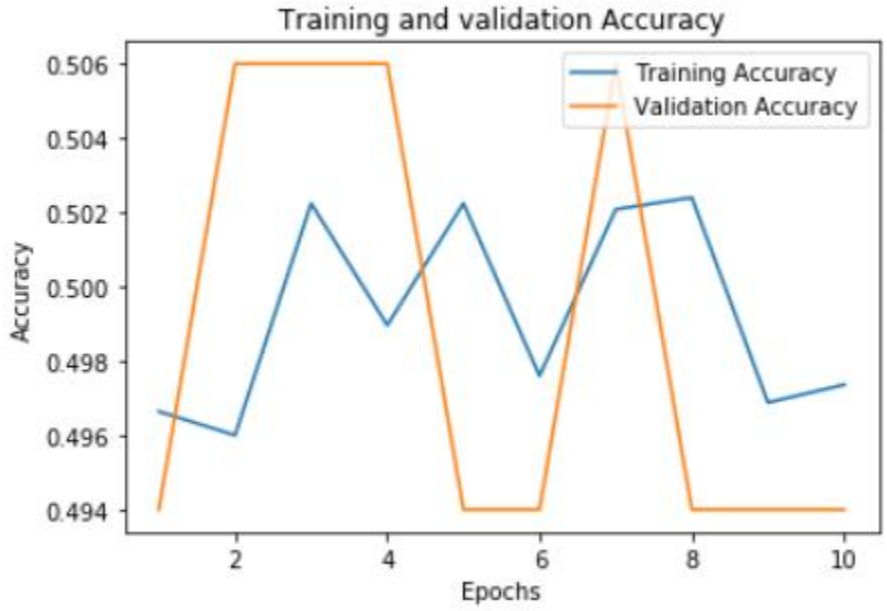
```
25000/25000 [==============================] - 2s 65us/step
[0.9007700750923157, 0.7129600048065186]
```

Glove model

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, 20, 100)           1000000
_________________________________________________________________
flatten_1 (Flatten)          (None, 2000)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               1024512
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 513
=================================================================
Total params: 2,025,025
Trainable params: 2,025,025
Non-trainable params: 0
_________________________________________________________________
```

Fitting the model

```
Epoch 5/10
25000/25000 [==============================] - 7s 288us/step - loss: 0.693
2 - acc: 0.5022 - val_loss: 0.6932 - val_acc: 0.4940
Epoch 6/10
25000/25000 [==============================] - 6s 240us/step - loss: 0.693
2 - acc: 0.4976 - val_loss: 0.6932 - val_acc: 0.4940
Epoch 7/10
25000/25000 [==============================] - 6s 230us/step - loss: 0.693
2 - acc: 0.5021 - val_loss: 0.6931 - val_acc: 0.5060
Epoch 8/10
25000/25000 [==============================] - 6s 238us/step - loss: 0.693
2 - acc: 0.5024 - val_loss: 0.6932 - val_acc: 0.4940
Epoch 9/10
25000/25000 [==============================] - 6s 242us/step - loss: 0.693
2 - acc: 0.4969 - val_loss: 0.6932 - val_acc: 0.4940
Epoch 10/10
25000/25000 [==============================] - 6s 248us/step - loss: 0.693
2 - acc: 0.4974 - val_loss: 0.6932 - val_acc: 0.4940
```



Test accuracy – 50%

```
25000/25000 [==============================] - 1s 35us/step
[0.6931483315849304, 0.5]
```

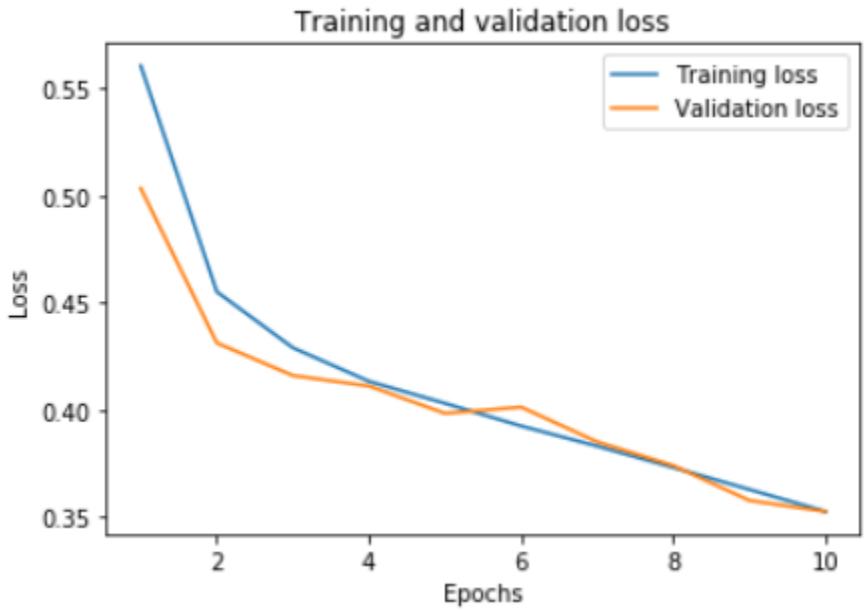
LSTM model

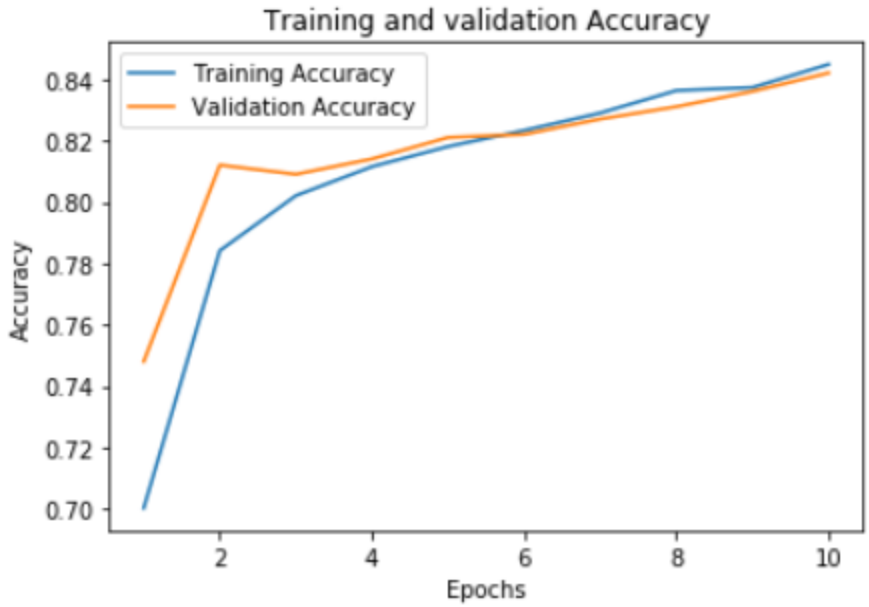
```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
```

```
embedding_1 (Embedding)      (None, None, 8)           80000
_________________________________________________________________
lstm_1 (LSTM)                (None, 128)               70144
_________________________________________________________________
dropout_1 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 129
=================================================================
Total params: 150,273
Trainable params: 150,273
Non-trainable params: 0
_________________________________________________________________
```

Model fitting

```
Epoch 6/10
25000/25000 [==============================] - 32s 1ms/step - loss: 0.3925
- acc: 0.8233 - val_loss: 0.4013 - val_acc: 0.8220
Epoch 7/10
25000/25000 [==============================] - 32s 1ms/step - loss: 0.3831
- acc: 0.8290 - val_loss: 0.3851 - val_acc: 0.8270
Epoch 8/10
25000/25000 [==============================] - 32s 1ms/step - loss: 0.3731
- acc: 0.8363 - val_loss: 0.3740 - val_acc: 0.8310
Epoch 9/10
25000/25000 [==============================] - 32s 1ms/step - loss: 0.3629
- acc: 0.8373 - val_loss: 0.3578 - val_acc: 0.8360
Epoch 10/10
25000/25000 [==============================] - 34s 1ms/step - loss: 0.3526
- acc: 0.8448 - val_loss: 0.3526 - val_acc: 0.8420
```





Test accuracy = 77%

```
25000/25000 [==============================] - 10s 418us/step
[0.4813276582145691, 0.7717599868774414]
```

Conclusion

The LSTM model seemed to give better accuracy due to its capability to remember sequences.

References

Code given by Professor

Keras page for IMDB dataset.