Link : https://github.com/jxc84430/NeuralNetworks/tree/main/ICP_10

Spring 2023 : CS5720

Neural Networks & Deep Learning ICP_10 : Jahnavi Chadalavada (700728443)

```
[35] from google.colab import drive
     drive.mount('/NN')
```

```
Mounted at /NN
```

```
[36] path_to_csv = '/NN/MyDrive/Colab Notebooks/NN/Sentiment.csv'
```

```
[37] import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
     from keras.preprocessing.text import Tokenizer
     from keras.models import Sequential
     from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
     from matplotlib import pyplot
     from sklearn.model_selection import train_test_split
     from keras.utils.np_utils import to_categorical
     import re
     from sklearn.preprocessing import LabelEncoder
     import tensorflow as tf
```

```
[38] data = pd.read_csv(path_to_csv)
```

```
[40] # Keeping only the neccessary columns
     data = data[['text','sentiment']]
     data['text'] = data['text'].apply(lambda x: x.lower())
     data['text'] = data['text'].apply((lambda x: re.sub('[^a-zA-z0-9\s]', '', x)))
```

```
[41] for idx, row in data.iterrows():
         row[0] = row[0].replace('rt', ' ')

     max_fatures = 2000
     tokenizer = Tokenizer(num_words=max_fatures, split=' ')
     tokenizer.fit_on_texts(data['text'].values)
     X = tokenizer.texts_to_sequences(data['text'].values)
```

```
[42] X = tf.keras.utils.pad_sequences(X)
     X.shape
```

```
(13871, 28)
```

```
[43] embed_dim = 128
     lstm_out = 196
```

```
[44] def createmodel():
         model = Sequential()
         model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1]))
         model.add(LSTM(lstm_out, dropout=0.2, recurrent_dropout=0.2))
         model.add(Dense(3,activation='softmax'))
         model.compile(loss = 'categorical_crossentropy', optimizer='adam',metrics = ['accuracy'])
         return model
```

```
labelencoder = LabelEncoder()
integer_encoded = labelencoder.fit_transform(data['sentiment'])
y = to_categorical(integer_encoded)
X_train, X_test, Y_train, Y_test = train_test_split(X,y, test_size = 0.33, random_state = 42)

batch_size = 32
model = createmodel()
```

```
WARNING:tensorflow:Layer lstm_1 will not use cuDNN kernels since it doesn't meet the criteria. It will use a generic GPU
```

```
model.fit(X_train, Y_train, epochs = 1, batch_size=batch_size, verbose = 2)
```

```
291/291 - 58s - loss: 0.8234 - accuracy: 0.6449 - 58s/epoch - 199ms/step
<keras.callbacks.History at 0x7f95e443d340>
```

```
score,acc = model.evaluate(X_test,Y_test,verbose=2,batch_size=batch_size)
```

```
144/144 - 3s - loss: 0.7486 - accuracy: 0.6737 - 3s/epoch - 22ms/step
```

```
print(score)
print(acc)
print(model.metrics_names)
```

```
0.7486432194709778
0.6736566424369812
['loss', 'accuracy']
```

```
model.save("/NN/MyDrive/Colab Notebooks/NN/LSTM_Twitter.h5")
```

```
[46] model = tf.keras.models.load_model('/NN/MyDrive/Colab Notebooks/NN/LSTM_Twitter.h5')
```

Prediction on new text data

```
sentence = ["A lot of good things are happening. We are respected again throughout the world, and that's a great thing.@
s = list(map(lambda x: x.lower(),sentence))
s = list(map((lambda x: re.sub('[^a-zA-z0-9\s]', '', x)),s))
s = list(map(lambda x: x.replace('rt', ' '),s))

max_fatures = 2000
tokenizer = Tokenizer(num_words=max_fatures, split=' ')
tokenizer.fit_on_texts(s)
input = tokenizer.texts_to_sequences(s)
input = tf.keras.utils.pad_sequences(input)
```

```
prediction = model.predict(input)
```

```
1/1 [==============================] - 0s 27ms/step
```

```
print(prediction)
```

```
[[0.7791218  0.13369924 0.08717896]]
```

```
[ ] import numpy as np
    pred_labels = []
    for i in prediction:
        print(i,np.max(i))
        if np.max(i) >= 0.5:
            pred_labels.append(1)
        else:
            pred_labels.append(0)
    for i in range(len(sentence)):
        print(sentence[i])
        if pred_labels[i] == 1:
            s = 'Positive'
        else:
            s = 'Negative'
        print("Predicted sentiment : ",s)
```

```
[0.7791218  0.13369924 0.08717896] 0.7791218
A lot of good things are happening. We are respected again throughout the world, and that's a great thing.@realDonaldTrum
Predicted sentiment :  Positive
```

GridSearchCV

```
✓   [93] from scikeras.wrappers import KerasClassifier
0s       model = KerasClassifier(build_fn=model,verbose=0)
         batch_size = [10,20, 40]
         epochs = [1, 2, 3]
         param_grid = dict(batch_size=batch_size, epochs=epochs)
```

```
✓   [94] from sklearn.model_selection import GridSearchCV
5m       grid = GridSearchCV(estimator=model, param_grid=param_grid,cv=2)
         grid_result = grid.fit(X_train, Y_train)
```

```
WARNING:absl:Found untraced functions such as _update_step_xla while saving (showing 1 of 1). These functions will not be
WARNING:tensorflow:Detecting that an object or model or tf.train.Checkpoint is being deleted with unrestored values. See
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).keras_api.metrics.0.total
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).keras_api.metrics.0.count
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).keras_api.metrics.1.total
```

```
✓   [95] print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
0s
```

```
Best: 0.651111 using {'batch_size': 20, 'epochs': 2}
```