

NEURAL NETWORK & DEEP LEARNING

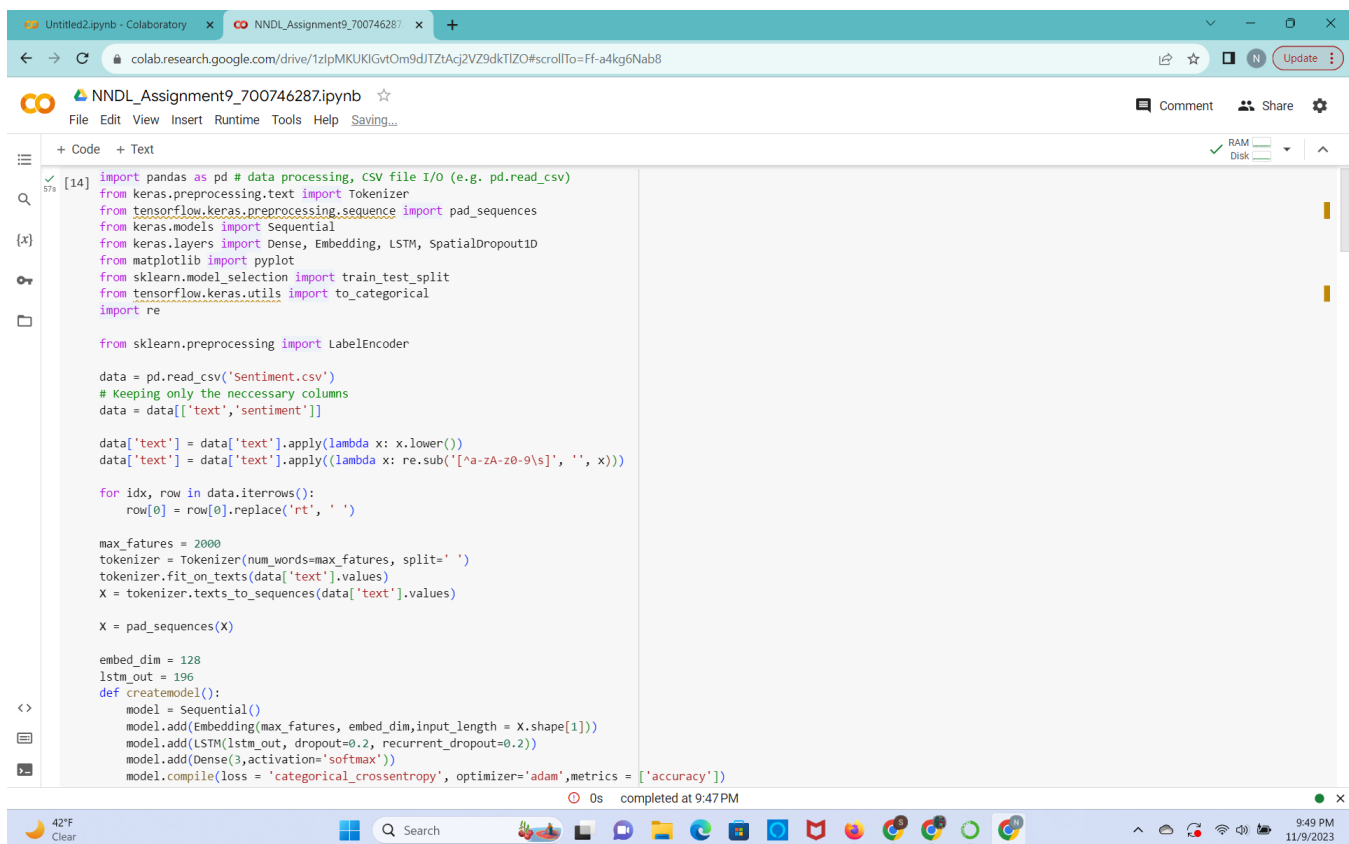
ASSIGNMENT 9

Name : SAI SNUSHA NAKKA

Student ID : 700746287

Git hub Link: https://github.com/NSnusha/NNDL_Assignment9

Video link: https://drive.google.com/file/d/1E-dWl3fXiBBdrr8L7x9bWuABILK52GVg/view?usp=share_link



```
[14]: import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
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 tensorflow.keras.utils import to_categorical
import re

from sklearn.preprocessing import LabelEncoder

data = pd.read_csv('Sentiment.csv')
# Keeping only the necessary 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))

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)

X = pad_sequences(X)

embed_dim = 128
lstm_out = 196
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'])
```

Untitled2.ipynb - Colaboratory x NNDL_Assignment9_700746287 x +

colab.research.google.com/drive/1zlpMKUKIGvOm9dJTzAcj2VZ9dkTIZO#scrollTo=FF-a4kg6Nab8

NNDL_Assignment9_700746287.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

[14]

```
embed_dim = 128
lstm_out = 196
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
# print(model.summary())

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()
model.fit(X_train, Y_train, epochs = 1, batch_size=batch_size, verbose = 2)
score,acc = model.evaluate(X_test,Y_test,verbose=2,batch_size=batch_size)
print(score)
print(acc)
print(model.metrics_names)
```

291/291 - 51s - loss: 0.8309 - accuracy: 0.6439 - 51s/epoch - 176ms/step
144/144 - 3s - loss: 0.7467 - accuracy: 0.6739 - 3s/epoch - 22ms/step
0.7466925382614136
0.6738750338554382
['loss', 'accuracy']

[15] model.save('sentiment_model.h5')

/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered leg
saving_api.save_model()

42°F Clear

Search

0s completed at 9:47 PM

11/9/2023

Untitled2.ipynb - Colaboratory x NNDL_Assignment9_700746287 x +

colab.research.google.com/drive/1zlpMKUKIGvOm9dJTzAcj2VZ9dkTIZO#scrollTo=FF-a4kg6Nab8

NNDL_Assignment9_700746287.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

[17]

```
from keras.models import load_model
import numpy as np

loaded_model = load_model('sentiment_model.h5')

new_text = ["A lot of good things are happening. We are respected again throughout the world, and that's a great thing.@realDonaldTrump"]
new_text = tokenizer.texts_to_sequences(new_text)
new_text = pad_sequences(new_text, maxlen=X.shape[1], dtype='int32', value=0)
sentiment_prob = loaded_model.predict(new_text, batch_size=1, verbose=2)[0]

sentiment_classes = ['Negative', 'Neutral', 'Positive']
sentiment_pred = sentiment_classes[np.argmax(sentiment_prob)]

print("Predicted sentiment: ", sentiment_pred)
print("Predicted probabilities: ", sentiment_prob)
```

1/1 - 0s - 274ms/epoch - 274ms/step
Predicted sentiment: Positive
Predicted probabilities: [0.43323177 0.12365673 0.44311148]

```
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV
from keras.optimizers import Adam

def create_model(units=196, dropout=0.2, learning_rate=0.001):
    model = Sequential()
    model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1]))
    model.add(LSTM(units, dropout=dropout, recurrent_dropout=dropout))
    model.add(Dense(3, activation='softmax'))
    optimizer = Adam(lr=learning_rate)
    model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
    return model

model = KerasClassifier(build_fn=create_model, verbose=2)
```

Untitled2.ipynb - Colaboratory

colab.research.google.com/drive/1zlpMKUKIGvOm9dJTzAcj2VZ9dkTIZO#scrollTo=FF-a4kg6Nab8

Update

Comment Share

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

```
from keras.wrappers.scikit_learn import KerasClassifier
from sklearn.model_selection import GridSearchCV
from keras.optimizers import Adam

def create_model(units=196, dropout=0.2, learning_rate=0.001):
    model = Sequential()
    model.add(Embedding(max_fatures, embed_dim,input_length = X.shape[1]))
    model.add(LSTM(units, dropout=dropout, recurrent_dropout=dropout))
    model.add(Dense(3, activation='softmax'))
    optimizer = Adam(lr=learning_rate)
    model.compile(loss='categorical_crossentropy', optimizer=optimizer, metrics=['accuracy'])
    return model

model = KerasClassifier(build_fn=create_model, verbose=2)

units = [64, 128, 196]
dropout = [0.1, 0.2, 0.3]
learning_rate = [0.001, 0.01, 0.1]
epochs = [1]
batch_size = [32]

param_grid = dict(units=units, dropout=dropout, learning_rate=learning_rate, epochs=epochs, batch_size=batch_size)
grid = GridSearchCV(estimator=model, param_grid=param_grid, cv=3, verbose=2)
grid_result = grid.fit(X_train, Y_train)

print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.01, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.1, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.1, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.1, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.1, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.3, epochs=1, learning_rate=0.1, units=128; total time= 0.0s

0s completed at 9:47 PM

42°F Clear

colab.research.google.com/drive/1zlpMKUKIGvOm9dJTzAcj2VZ9dkTIZO#scrollTo=FF-a4kg6Nab8

Update

Comment Share

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

RAM Disk

```
param_grid = dict(units=units, dropout=dropout, learning_rate=learning_rate, epochs=epochs, batch_size=batch_size)
grid = GridSearchCV(estimator=model, param_grid=param_grid, cv=3, verbose=2)
grid_result = grid.fit(X_train, Y_train)

print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

<ipython-input-34-6bec1c9b5d5d>:14: DeprecationWarning: KerasClassifier is deprecated, use Sci-Keras (https://github.com/adriangb/scikeras) instead. See https://www.adriangb.com/sci/
model = KerasClassifier(build_fn=create_model, verbose=2)
Fitting 3 folds for each of 27 Candidates, totalling 81 fits
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.001, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.01, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=128; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.1, epochs=1, learning_rate=0.1, units=196; total time= 0.0s
[CV] END batch_size=32, dropout=0.2, epochs=1, learning_rate=0.001, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.2, epochs=1, learning_rate=0.001, units=64; total time= 0.0s
[CV] END batch_size=32, dropout=0.2, epochs=1, learning_rate=0.001, units=64; total time= 0.0s

0s completed at 9:47 PM

