SPEECH EMOTIONAL RECOGNITION

Assignment-3 Patter Recognition

Ahmed Abdelkabier Hassan	6669
Mohanned Bashar	6656

1-Download Data set and understand it:

```
import os
if not os.path.exists(r'/content/feature_space.pkl'):
  audio_signal = []
  samples_rate =  []
  files_path = pd.Series(filespath , name='.wav_path').astype(str)
  labels = pd.Series(emo_labels , name='labels')

  for audios in files_path:
      audio , sample_rate = lib.load(audios)
      audio_signal.append(audio)
      samples_rate.append(sample_rate)

  audio_features = pd.Series(audio_signal , name='signal')
  data = pd.concat([files_path,audio_features,labels] , axis=1)
else:
  data = None
```

```
[5] data
```

	.wav_path	signal	labels
0	/content/drive/MyDrive/Crema/1081_MTI_FEA_XX.wav	[-0.0013267642, -0.0011507067, -0.00074600475,...	FEA
1	/content/drive/MyDrive/Crema/1081_IOM_HAP_XX.wav	[0.00406869, 0.004215974, 0.0030919232, 0.0023...	HAP
2	/content/drive/MyDrive/Crema/1081_ITS_ANG_XX.wav	[-0.00031764526, -0.00031087518, 0.00014127654...	ANG
3	/content/drive/MyDrive/Crema/1079_TSI_DIS_XX.wav	[0.001069233, 0.0007935993, -0.00016964517, -0...	DIS
4	/content/drive/MyDrive/Crema/1081_IWL_HAP_XX.wav	[3.5962148e-05, 0.00024702173, 0.00045050596, ...	HAP
...	...	...	...
7446	/content/drive/MyDrive/Crema/1006_IOM_SAD_XX.wav	[0.0014635584, 0.0016795307, 0.0010614916, 0.0...	SAD
7447	/content/drive/MyDrive/Crema/1007_TSI_SAD_XX.wav	[0.0033311676, 0.004324746, 0.0038436854, 0.00...	SAD
7448	/content/drive/MyDrive/Crema/1006_TAI_SAD_XX.wav	[0.00387666, 0.0033433223, 0.002694597, 0.0031...	SAD
7449	/content/drive/MyDrive/Crema/1006_ITS_ANG_XX.wav	[-0.00012446963, -0.000649155, -0.0005525435, ...	ANG
7450	/content/drive/MyDrive/Crema/1008_IOM_SAD_XX.wav	[-0.010131634, -0.01164933, -0.010184769, -0.0...	SAD

7451 rows × 3 columns

Play and plot the audios:

Play audio and plot wave for each class

```
def play_and_plot():
 for i in set(emo_labels):
   plt.title(i + " AUDIO")
   sad_audios = data[data['labels'] == i]['.wav_path']
   sad_feature =  data[data['labels'] == i]['signal']

   librosa.display.waveplot(sad_feature.iloc[0])
   display(Audio(sad_audios.iloc[0]))
   plt.figure()
```

```
[8] play_and_plot()
```

0:00 / 0:02

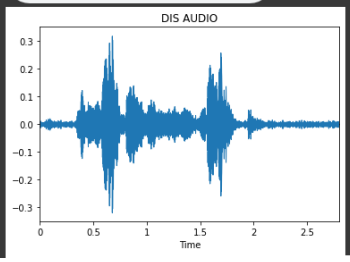
0:00 / 0:03

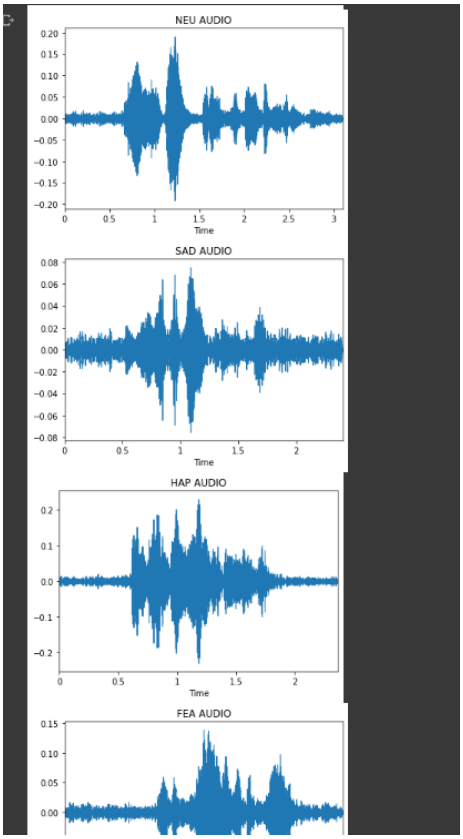
0:00 / 0:02

0:00 / 0:02

0:00 / 0:03

0:00 / 0:02





Some work on data

PADDING DATA

```
[9] def padding(data_features):
      max_audio_len = max([len(i) for i in  data_features['signal']])
      for index , audio in enumerate(data_features['signal']):
        diff = max_audio_len - len(audio)
        data_features['signal'][index] = np.pad(audio , pad_width= (0,diff) , mode= 'constant')
```

```
[10] padding(features)
     length = [len(i) for i in features['signal']]
     print(length)
     features
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#return
  """
[110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 110361, 1
```

	.wav_path	signal
0	/content/drive/MyDrive/Crema/1081_MTI_FEA_XX.wav	[-0.0013267642, -0.0011507067, -0.00074600475,...
1	/content/drive/MyDrive/Crema/1081_IOM_HAP_XX.wav	[0.00406869, 0.004215974, 0.0030919232, 0.0023...
2	/content/drive/MyDrive/Crema/1081_ITS_ANG_XX.wav	[-0.00031764526, -0.00031087518, 0.00014127654...
3	/content/drive/MyDrive/Crema/1079_TSI_DIS_XX.wav	[0.001069233, 0.0007935993, -0.00016964517, -0...
4	/content/drive/MyDrive/Crema/1081_IWL_HAP_XX.wav	[3.5962148e-05, 0.00024702173, 0.00045050596, ...
...	...	...
7446	/content/drive/MyDrive/Crema/1006_IOM_SAD_XX.wav	[0.0014635584, 0.0016795307, 0.0010614916, 0.0...
7447	/content/drive/MyDrive/Crema/1007_TSI_SAD_XX.wav	[0.0033311676, 0.004324746, 0.0038436854, 0.00...
7448	/content/drive/MyDrive/Crema/1006_TAI_SAD_XX.wav	[0.00387666, 0.0033433223, 0.002694597, 0.0031...
7449	/content/drive/MyDrive/Crema/1006_ITS_ANG_XX.wav	[-0.00012446963, -0.000649155, -0.0005525435, ...
7450	/content/drive/MyDrive/Crema/1008_IOM_SAD_XX.wav	[-0.010131634, -0.01164933, -0.010184769, -0.0...

7451 rows × 2 columns

2-Creating the feature spaces:

```
def create_feature_space(_audio_data_frame , sampling_rate):
  zero_crossing_rate_Time = []
  singal_Energy = []
  mel_spectrogram = []

  for audio  in _audio_data_frame['signal']:
    zero_crossing_rate_Time.append(lib.feature.zero_crossing_rate(y = audio))
    singal_Energy.append(lib.feature.rms(y = audio))
    mel_spectrogram.append(lib.feature.melspectrogram(y= audio , sr= sampling_rate , n_mels= 64))

  zero_crossing_rate_Time = pd.Series(zero_crossing_rate_Time , name= 'zero_crossing_rate_Time')
  singal_Energy = pd.Series(singal_Energy , name= 'Energy')
  mel_spectrogram = pd.Series(mel_spectrogram , name= 'mel_spectrogram')
  _audio_data_frame = pd.concat([_audio_data_frame , zero_crossing_rate_Time , singal_Energy , mel_spectrogram] , axis=1)

  return _audio_data_frame
```

```
[70] _new_fs_data.to_pickle('/content/drive/MyDrive/feature_space.pkl')
```

```
if not os.path.exists(r'/content/feature_space.pkl'):
  _new_fs_data = create_feature_space(features , samples_rate[0])
  _new_fs_data.to_pickle('/content/feature_space.pkl')
else:
  _new_fs_data = pd.read_pickle('/content/feature_space.pkl')
```

	.wav_path	signal	zero_crossing_rate_Time	Energy	mel_spectrogram
0	/content/drive/MyDrive/Crema/1081_MTI_FEA_XX.wav	[-0.0013267642, -0.0011507067, -0.00074600475,...	[[0.01171875, 0.0166015625, 0.0234375, 0.03076...	[[0.003751567, 0.00499682, 0.005775987, 0.0059...	[[0.041217044, 0.036117308, 0.041154295, 0.050...
1	/content/drive/MyDrive/Crema/1081_IOM_HAP_XX.wav	[0.00406869, 0.004215974, 0.0030919232, 0.0023...	[[0.01220703125, 0.021484375, 0.0322265625, 0....	[[0.005943279, 0.0060860105, 0.0055265073, 0.0...	[[0.045877956, 0.018270325, 0.0338492, 0.0553...
2	/content/drive/MyDrive/Crema/1081_ITS_ANG_XX.wav	[-0.00031764526, -0.00031087518, 0.00014127654...	[[0.009765625, 0.0224609375, 0.03076171875, 0....	[[0.006384029, 0.0061387853, 0.005591352, 0.00...	[[0.07248108, 0.064830616, 0.04831033, 0.06275...
3	/content/drive/MyDrive/Crema/1079_TSI_DIS_XX.wav	[0.001069233, 0.0007935993, -0.00016964517, -0...	[[0.0166015625, 0.02294921875, 0.02783203125, ...	[[0.0044570197, 0.0051914803, 0.006885936, 0.0...	[[0.07827319, 0.029487826, 0.035867628, 0.0555...
4	/content/drive/MyDrive/Crema/1081_IWL_HAP_XX.wav	[3.5962148e-05, 0.00024702173, 0.00045050596, ...	[[0.02294921875, 0.03466796875, 0.03955078125,...	[[0.0047711073, 0.004686294, 0.005223743, 0.0...	[[0.041727513, 0.07174666, 0.06327322, 0.06390...
...	...	...	...	...	...
7446	/content/drive/MyDrive/Crema/1006_IOM_SAD_XX.wav	[0.0014635584, 0.0016795307, 0.0010614916, 0.0...	[[0.01513671875, 0.02685546875, 0.04541015625,...	[[0.005884879, 0.00562598, 0.00491079, 0.0050...	[[0.018325038, 0.04317272, 0.06460701, 0.04703...
7447	/content/drive/MyDrive/Crema/1007_TSI_SAD_XX.wav	[0.0033311676, 0.004324746, 0.0038436854, 0.00...	[[0.01220703125, 0.01708984375, 0.0244140625, ...	[[0.0071519525, 0.0076949564, 0.007697036, 0....	[[0.006068286, 0.010732218, 0.014269074, 0.025...

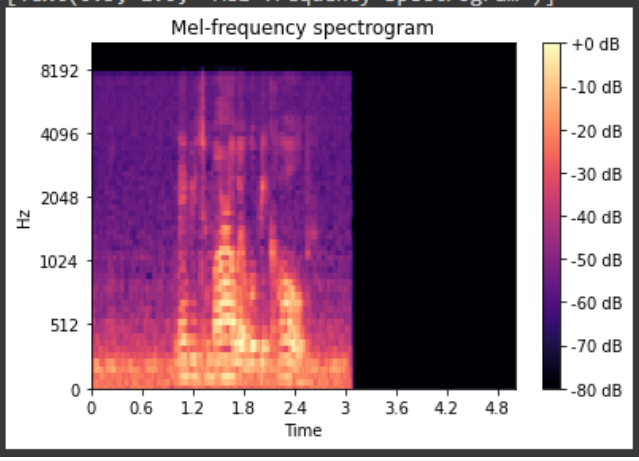
Executing (2h 9m 47s) Cell > error_handler() > fit() > error_handler() > __call__() > _call() > __call__() > _call_flat() > call() > quick_execute()

Mel spectrogram example:

plot spectrogram

```
fig, ax = plt.subplots()
S_dB = librosa.power_to_db(_new_fs_data['mel_spectrogram'][0], ref=np.max)
img = librosa.display.specshow(S_dB, x_axis='time',
                               y_axis='mel', sr= samples_rate[0],
                               fmax=samples_rate[0] / 2, ax=ax)
fig.colorbar(img, ax=ax, format='%+2.0f dB')
ax.set(title='Mel-frequency spectrogram')
```

```
[Text(0.5, 1.0, 'Mel-frequency spectrogram')]
```



3-Bulding models 1D-CNN , 2D-CNN:

- Data split 30% test and 70% train
- 5% of train validation test

3)DATA SPLITING

70% trainig set and 30% testing set 5% of trainig set will be validation set

```
[39] from sklearn.model_selection import train_test_split

     freq_features = pd.DataFrame(_new_fs_data[['mel_spectrogram']])


     #SPLIT DATA INTO 30% TEST AND 70% TRAIN
     train_data_time , test_data_time , train_labels_time , test_labels_time = train_test_split(time_feature_space , labels , test_size= 0.3 , stratify= labels)
     train_data_freq , test_data_freq , train_labels_freq , test_labels_freq = train_test_split(freq_features , labels , test_size= 0.3 , stratify= labels)


     #SPLIT TRAIN INTO 5% VALIDATION AND 95% TRAIN
     training_data_time , validation_data_time , trainig_labels_time , validation_labels_time = train_test_split(train_data_time , train_labels_time , test_size= 0.05 , stratify= train_labels_time)
     training_data_freq , validation_data_freq , trainig_labels_freq , validation_labels_freq = train_test_split(train_data_freq , train_labels_freq , test_size= 0.05 , stratify= train_labels_freq)

     training_data_time.shape , test_data_time.shape , validation_data_time.shape , freq_features.shape

     ((4954, 432), (2236, 432), (261, 432), (7451, 1))
```

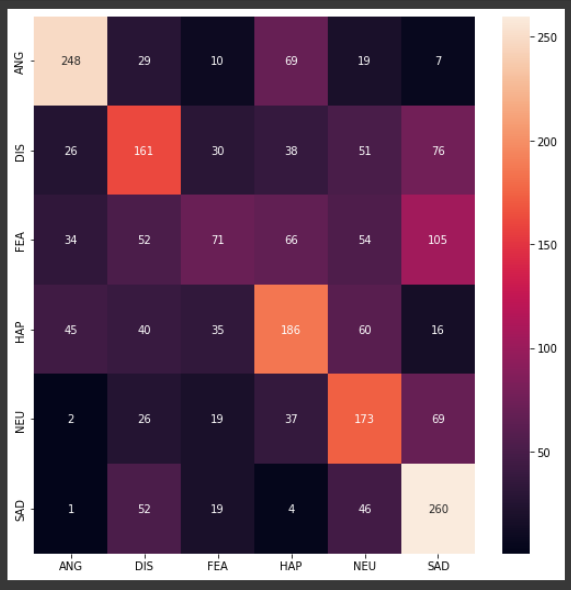
1D – Model:

```
#WORK ON DATA IN TIME DOMAIN
#REQ: TO SEND THE DATA IN TIME DOMIAN
with tf.device(tf.test.gpu_device_name()):
  model_time = Sequential()
  model_time.add(Conv1D(512, kernel_size=5,  padding='same', strides=1, activation ='relu', input_shape=(x_trainig_time.shape[1],1)))
  model_time.add(BatchNormalization())
  model_time.add(MaxPooling1D(pool_size=5,   padding='same', strides=2))

  model_time.add(Conv1D(512, kernel_size=5,  padding='same', strides=1))
  model_time.add(BatchNormalization())
  model_time.add(MaxPooling1D(pool_size=5,   padding='same', strides=2))

  model_time.add(Conv1D(128, kernel_size=5,  padding='same', strides=1))
  model_time.add(BatchNormalization())
  model_time.add(MaxPooling1D(pool_size = 5, padding='same', strides=2))

  model_time.add(Flatten())
  model_time.add(Dense(256, activation ='relu'))
  model_time.add(Dropout(0.3))
  model_time.add(Dense(6, activation ='softmax'))
  model_time.compile(optimizer="rmsprop", loss=tf.keras.losses.CategoricalCrossentropy(), metrics=["accuracy", f1_m])

  model_time.summary()
```

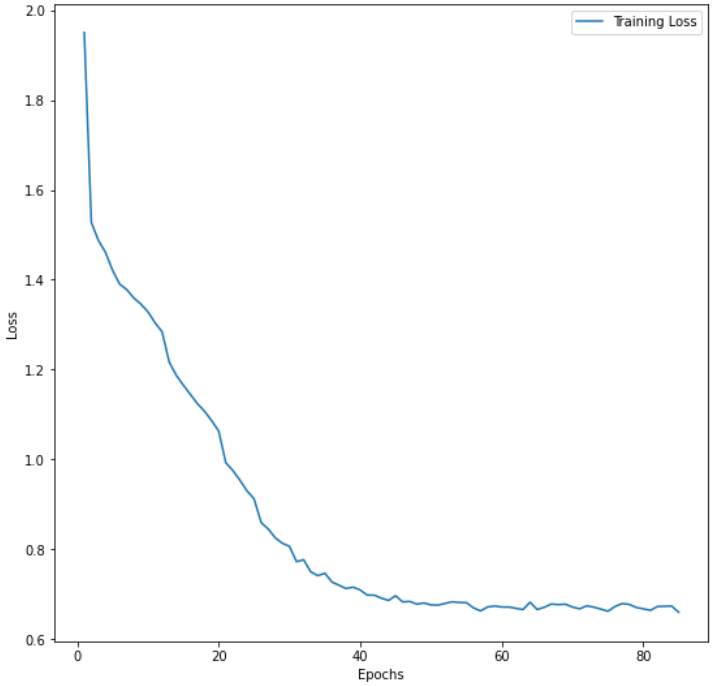
```
[ ] model_time =  tf.keras.models.load_model('/content/model_checkpoints/time_model_60')
```

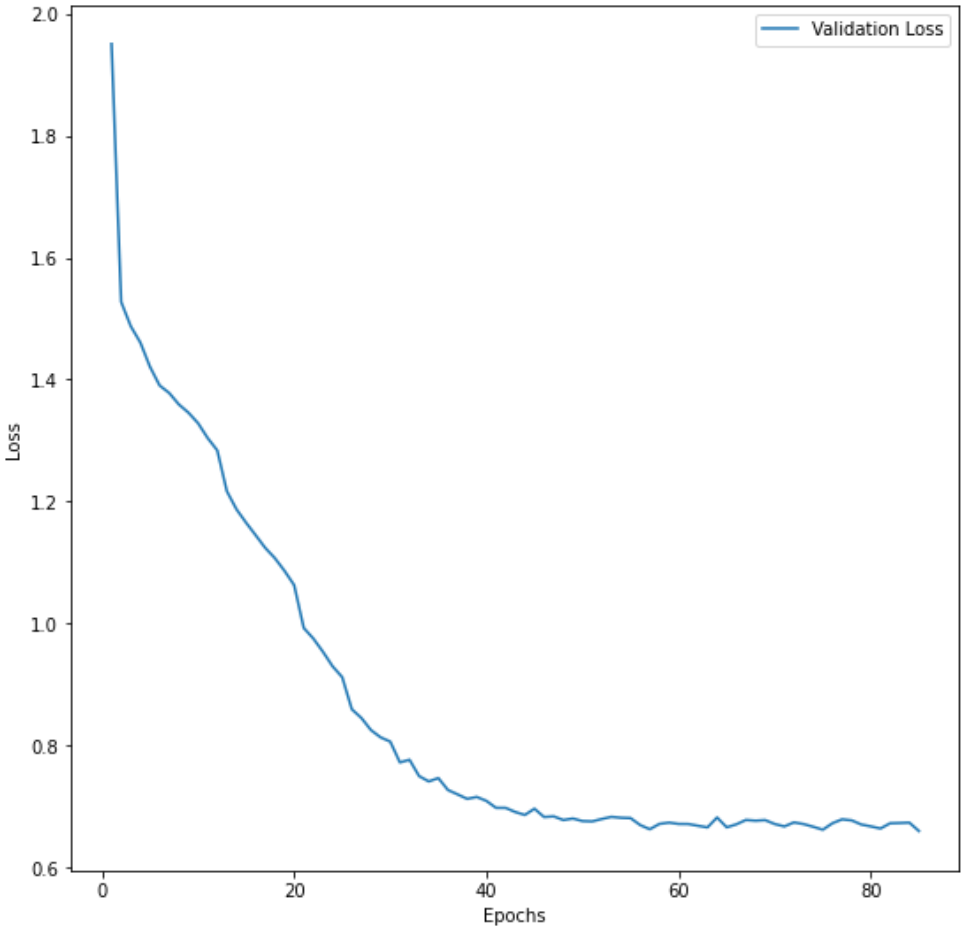
```
[ ] history = model_time.fit(x_trainig_time, trainig_labels_time, validation_data=(x_val_time, validation_labels_time),
                    epochs=EPOCHS, batch_size=32,
                    callbacks=[earlystopping, learning_rate_reduction, cp_callback])
```

```
[ ] from sklearn.metrics import confusion_matrix , ConfusionMatrixDisplay

    test_acc = model_time.evaluate(x_test_time , test_labels_time)[1]
    print("Test Accuracy = ", test_acc * 100 , "%")

    70/70 [==============================] - 26s 364ms/step - loss: 1.4206 - accuracy: 0.4915 - f1_m: 0.4181
    Test Accuracy =  49.150267243385315 %

[ ] test_prediction = np.argmax(model_time.predict(x_test_time) , axis=1)
    test_true = np.argmax(test_labels_time , axis=1)
    conf_matrix = confusion_matrix(y_pred= test_prediction , y_true= test_true)

[ ] import seaborn as snNew
    import matplotlib.pyplot as plt
    plt.rcParams['figure.figsize'] = [9, 9]

    snNew.heatmap(conf_matrix, annot=True , xticklabels= encode.classes_ , yticklabels= encode.classes_,fmt="d")
    plt.show()
```



Most confusing class is FEA


Training Loss
Loss
Epochs
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0
20
40
60
80


Validation Loss
Loss
Epochs
2.0
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0
20
40
60
80

2D-Model:

LOAD MODEL 2D FROM DRIVE

```
[46] model_spect= tf.keras.models.load_model('/content/checkpoints/spec')
```

```
[61] train_data_freq = training_data_freq['mel_spectrogram'].reset_index(drop=True)
     train_data_freq = train_data_freq.to_list()
     train_data_freq = tf.constant(train_data_freq)
```

```
[62] val_data_freq = validation_data_freq['mel_spectrogram'].reset_index(drop=True)
     val_data_freq = val_data_freq.to_list()
     val_data_freq = tf.constant(val_data_freq)
```

```
[63] tst_data_freq = test_data_freq['mel_spectrogram'].reset_index(drop=True)
     tst_data_freq = tst_data_freq.to_list()
     tst_data_freq = tf.constant(tst_data_freq)
```

```
[ ] train_data_freq.shape
```

```
TensorShape([4954, 64, 216])
```

```
[ ] del data
    del time_feature_space
```

REMOVE UNNECCESSARY VARIABLES FROM RAM

```
[77] import gc
     with tf.device('/device:GPU:0'):
       for _ in range(100):
         gc.collect()
         tf.keras.backend.clear_session()
```

```
[82] with tf.device('/device:GPU:0'):
       model_spect.fit(train_data_freq,trainig_labels_freq,
                       validation_data=(val_data_freq,validation_labels_freq),
                       epochs=EPOCHS,
                       batch_size=BATCH_SIZE,
                       callbacks=[earlystopping,learning_rate_reduction,cp_callback])
```

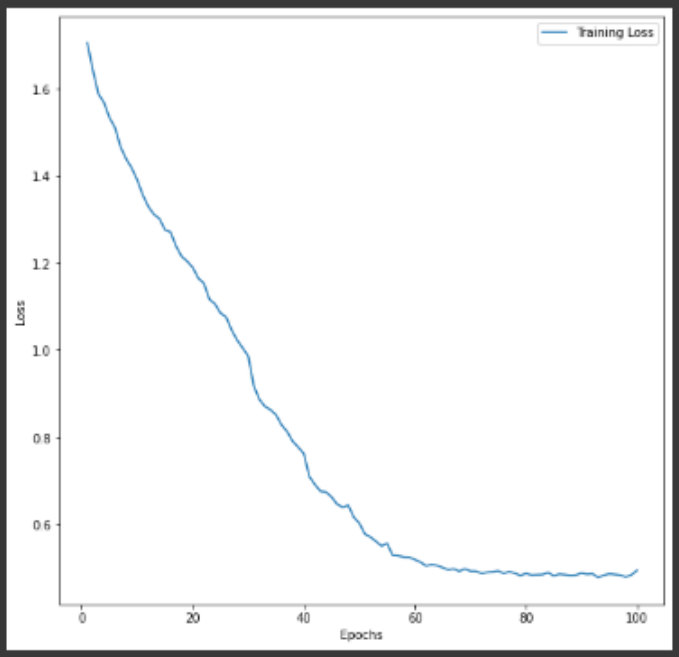
```
[144] from sklearn.metrics import confusion_matrix , ConfusionMatrixDisplay
      with tf.device('/device:GPU:0'):
        test_acc_2D = model_spect.evaluate(tst_data_freq , test_labels_freq)[1]
        print("Test Accuracy = ", test_acc_2D * 100 , "%")
```

```
70/70 [==============================] - 8s 108ms/step - loss: 1.4375 - accuracy: 0.5371 - f1_m: 0.5082
Test Accuracy =  53.71198654174805 %
```

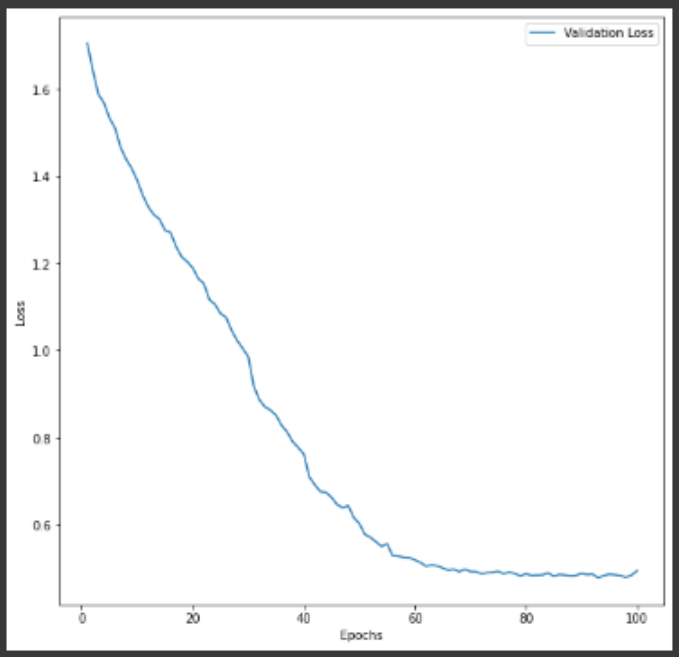
```
[57] test_labels_freq
```

```
array([[0., 0., 0., 1., 0., 0.],
       [0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 0., 1.],
       ...,
       [0., 0., 1., 0., 0., 0.],
       [0., 0., 0., 0., 1., 0.],
       [0., 0., 0., 0., 1., 0.]], dtype=float32)
```

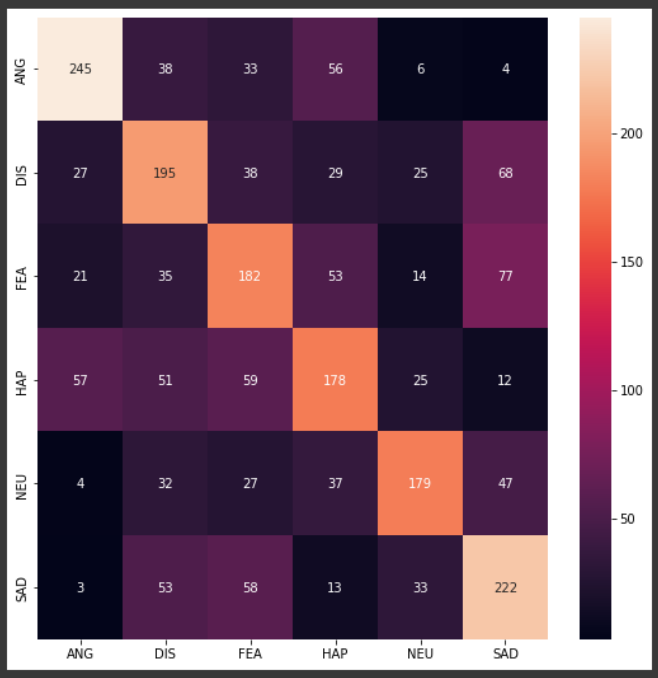
```
loss_values = history['loss']
epochs = range(1, len(loss_values)+1)
plt.plot(epochs, loss_values, label='Training Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```




```
[135] valid_loss = history['val_loss']
plt.plot(epochs, loss_values, label='Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()

plt.show()
```



```
import seaborn as snNew
import matplotlib.pyplot as plt
plt.rcParams['figure.figsize'] = [9, 9]

snNew.heatmap(conf_matrix_freq, annot=True , xticklabels= encode.classes_ , yticklabels= encode.classes_,fmt="d")
plt.show()
```



Most confusing class HAP