Assignment 3

Name:	**ID:**
Bishoy Refaat	5804
Karim El Sayad	6023
Islam Tarek Fahmy	5785
Seif Eldin Amr	5481
Yehia El Dallal	5607

Speech emotion recognition

Problem statement:

Speech Emotion Recognition, abbreviated as SER, is the act of attempting to recognize human emotion and affective states from speech. This is capitalizing on the fact that voice often reflects underlying emotion through tone and pitch. This is also the phenomenon that animals like dogs and horses employ to be able to understand human emotion.

Discovering the dataset:

In this project we used the CREMA dataset from Kaggle which is available in the following link:

https://www.kaggle.com/dmitrybabko/speech-emotion-recognition-en

The dataset consists of 7442 audio files in (.wav) format, each of which is named as follows:

aaaa_bbb_ccc_dd.wav

aaaa: 4 numbers representing the speaker's ID

bbb: 3 letters representing the abbreviation for the sentence being said

ccc: 3 letters representing the Emotion (aka labels)

dd: 2 letters representing the speaker's tone

We are mostly interested in the Emotion part as it represents our classes.

The dataset has 6 classes abbreviated as follows:

SAD → Sadness

ANG → Angry

DIS → Disgust

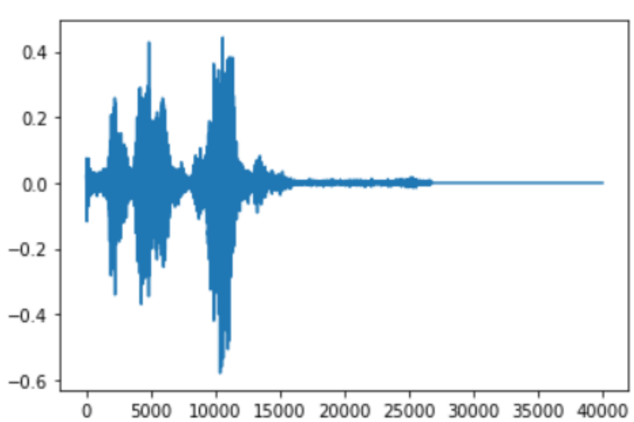
FEA → Fear

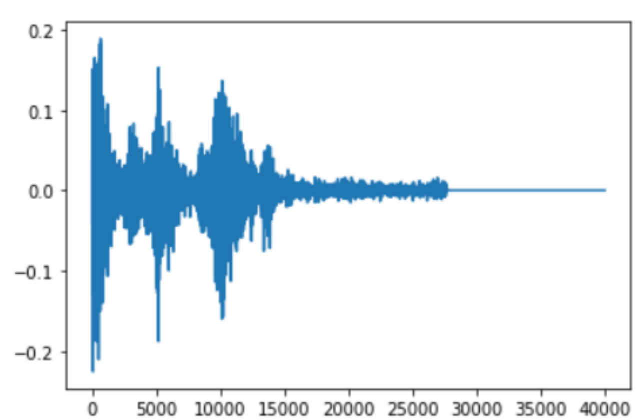
HAP → Happy

NEU → Neutral

Sample waveplots:

We extracted the audio files and labels using the file names as explained above.





Data Augmentation:

There are so many data augmentation techniques but after a lot of trials and observation, we decided to use the following 4 techniques as they yielded the best results:

1. Noise injection
2. Pitching

Feature extraction:

We considered a lot of features including the following:

1. Zero Crossing Rate: The rate of sign changes of the signal during the duration of a particular frame.
2. Energy: The sum of squares of the signal values, normalized by the respective frame length.
3. MFCC: Mel Frequency Cepstral Coefficients form a cepstral representation where the frequency bands are not linear but distributed according to the mel-scale.
4. RMSE: Compute root-mean-square (RMS) energy for each frame, either from the audio samples *y* or from a spectrogram *S*

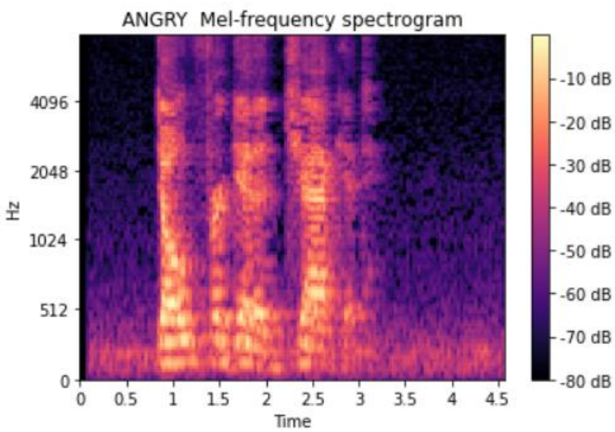
But after experimenting them we decided to use only the following 3 features as they yielded the best results:

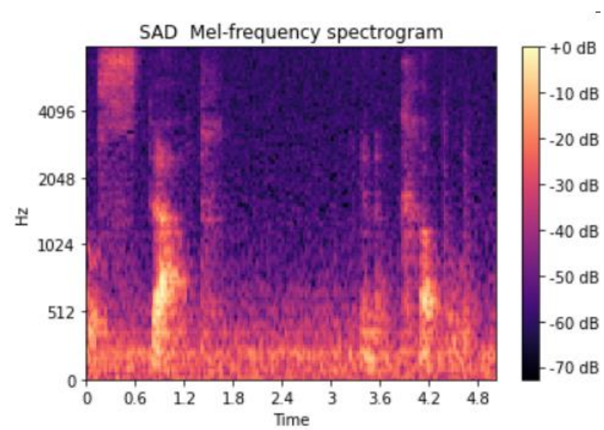
1. Zero Crossing Rate
2. RMS
3. MFCC

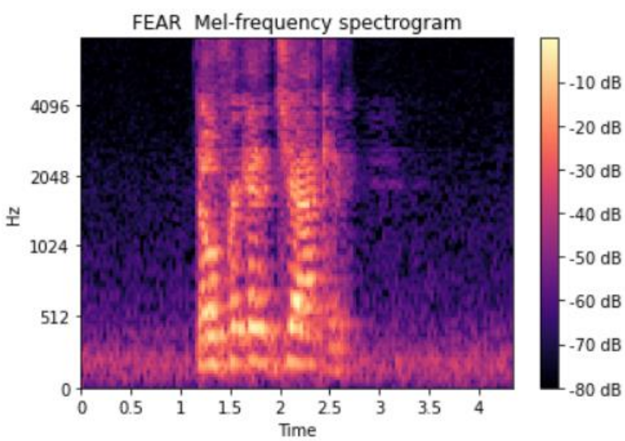
Mel-Spectrograms:

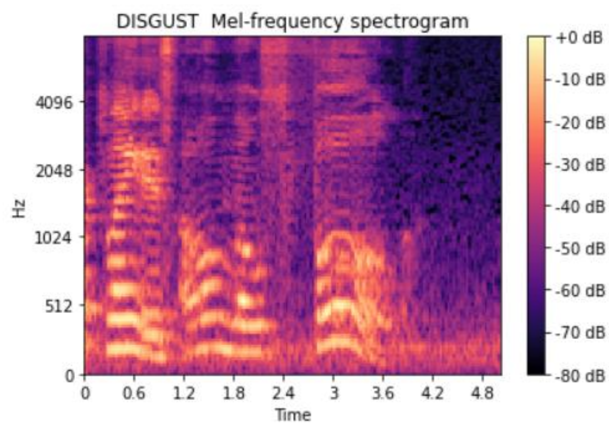
For our 2D model we used Mel-Spectrograms as our feature space.

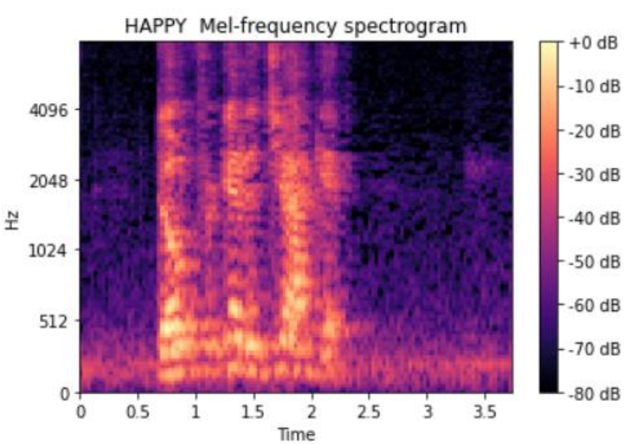
Here are samples from each class:

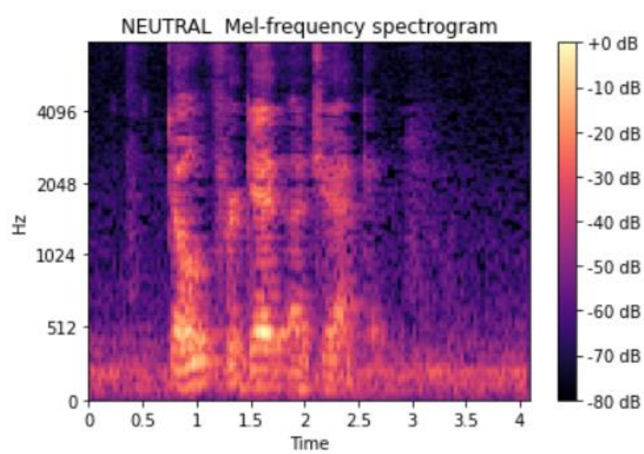












Splitting the dataset:

We first split the data into 70% for training and 30% for testing.

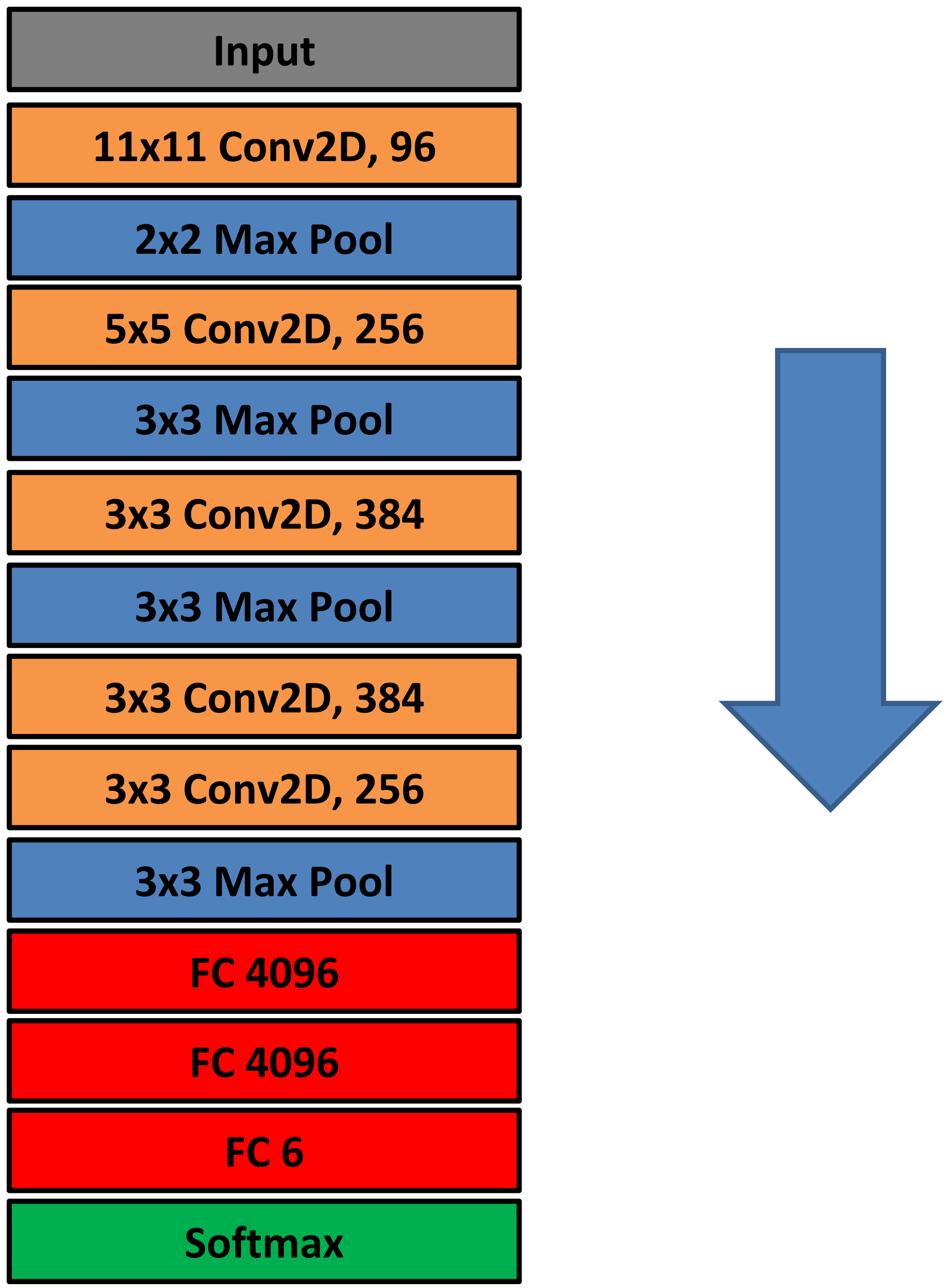
Building the models:-

1. 2D model:

For our 2D model we implemented AlexNet's model from scratch and used the Mel-Spectrograms as the feature space.
Here is the model's structure:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 128, 188, 96)      11712
_________________________________________________________________
module_wrapper (ModuleWrappe (None, 128, 188, 96)      384
_________________________________________________________________
activation (Activation)      (None, 128, 188, 96)      0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 64, 94, 96)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 64, 94, 256)       614656
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 64, 94, 256)       1024
_________________________________________________________________
activation_1 (Activation)    (None, 64, 94, 256)       0
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 21, 31, 256)       0
_________________________________________________________________
zero_padding2d (ZeroPadding2 (None, 23, 33, 256)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 23, 33, 384)       885120
_________________________________________________________________
module_wrapper_2 (ModuleWrap (None, 23, 33, 384)       1536
_________________________________________________________________
activation_2 (Activation)    (None, 23, 33, 384)       0
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 7, 11, 384)        0
_________________________________________________________________
zero_padding2d_1 (ZeroPaddin (None, 9, 13, 384)        0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 9, 13, 384)        1327488
_________________________________________________________________
module_wrapper_3 (ModuleWrap (None, 9, 13, 384)        1536
_________________________________________________________________
activation_3 (Activation)    (None, 9, 13, 384)        0
_________________________________________________________________
zero_padding2d_2 (ZeroPaddin (None, 11, 15, 384)       0
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 11, 15, 256)       884992
_________________________________________________________________
module_wrapper_4 (ModuleWrap (None, 11, 15, 256)       1024
_________________________________________________________________
activation_4 (Activation)    (None, 11, 15, 256)       0
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 3, 5, 256)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 3840)              0
_________________________________________________________________
dense_2 (Dense)              (None, 4096)              15732736
_________________________________________________________________
module_wrapper_5 (ModuleWrap (None, 4096)              16384
_________________________________________________________________
activation_5 (Activation)    (None, 4096)              0
_________________________________________________________________
dense_3 (Dense)              (None, 4096)              16781312
_________________________________________________________________
module_wrapper_6 (ModuleWrap (None, 4096)              16384
_________________________________________________________________
activation_6 (Activation)    (None, 4096)              0
_________________________________________________________________
dense_4 (Dense)              (None, 6)                 24582
_________________________________________________________________
module_wrapper_7 (ModuleWrap (None, 6)                 24
_________________________________________________________________
activation_7 (Activation)    (None, 6)                 0
=================================================================
Total params: 36,300,894
Trainable params: 36,281,746
Non-trainable params: 19,148
_________________________________________________________________
```

Model's diagram:

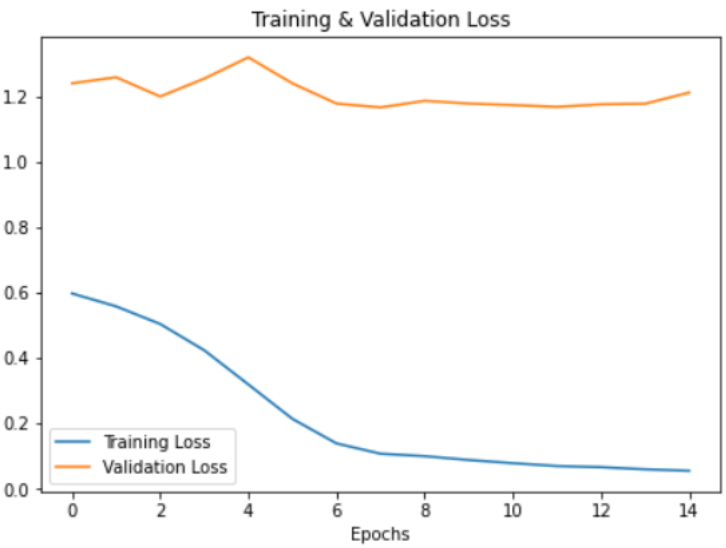


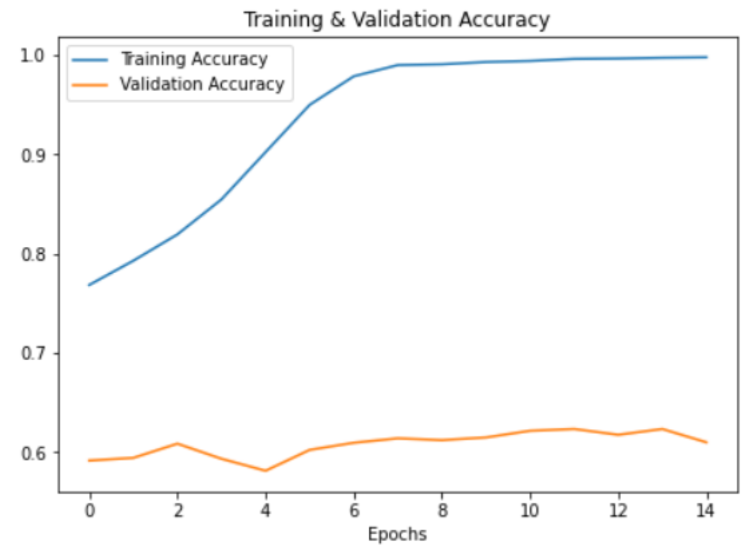
We ran the above model for 15 Epochs achieving the following results:

```
Epoch 1/15
489/489 [==============================] - 113s 160ms/step - loss: 1.5500 - accuracy: 0.3512 - val_loss: 1.5026 - val_accuracy: 0.4084
Epoch 2/15
489/489 [==============================] - 78s 160ms/step - loss: 1.4496 - accuracy: 0.3943 - val_loss: 1.3292 - val_accuracy: 0.4993
Epoch 3/15
489/489 [==============================] - 80s 164ms/step - loss: 1.3685 - accuracy: 0.4284 - val_loss: 1.5098 - val_accuracy: 0.3887
Epoch 4/15
489/489 [==============================] - 80s 165ms/step - loss: 1.3005 - accuracy: 0.4594 - val_loss: 1.6984 - val_accuracy: 0.3659
Epoch 5/15
489/489 [==============================] - 81s 165ms/step - loss: 1.2233 - accuracy: 0.4931 - val_loss: 1.4445 - val_accuracy: 0.4313

Epoch 00005: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.
Epoch 6/15
489/489 [==============================] - 81s 165ms/step - loss: 1.0768 - accuracy: 0.5565 - val_loss: 1.4438 - val_accuracy: 0.5128
Epoch 7/15
489/489 [==============================] - 81s 165ms/step - loss: 0.9698 - accuracy: 0.6023 - val_loss: 1.2211 - val_accuracy: 0.5540
Epoch 8/15
489/489 [==============================] - 80s 165ms/step - loss: 0.8837 - accuracy: 0.6418 - val_loss: 1.3882 - val_accuracy: 0.5405
Epoch 9/15
489/489 [==============================] - 81s 165ms/step - loss: 0.8234 - accuracy: 0.6596 - val_loss: 1.3942 - val_accuracy: 0.5343
Epoch 10/15
489/489 [==============================] - 81s 165ms/step - loss: 0.7665 - accuracy: 0.6827 - val_loss: 1.4272 - val_accuracy: 0.5199

Epoch 00010: ReduceLROnPlateau reducing learning rate to 0.0002500000118743628.
Epoch 11/15
489/489 [==============================] - 81s 165ms/step - loss: 0.6960 - accuracy: 0.7138 - val_loss: 1.2769 - val_accuracy: 0.5987
Epoch 12/15
489/489 [==============================] - 81s 165ms/step - loss: 0.6715 - accuracy: 0.7214 - val_loss: 1.2729 - val_accuracy: 0.5884
Epoch 13/15
489/489 [==============================] - 81s 165ms/step - loss: 0.6659 - accuracy: 0.7273 - val_loss: 1.3580 - val_accuracy: 0.5683
Epoch 14/15
489/489 [==============================] - 81s 165ms/step - loss: 0.6498 - accuracy: 0.7359 - val_loss: 1.3319 - val_accuracy: 0.5817

Epoch 00014: ReduceLROnPlateau reducing learning rate to 0.0001250000059371814.
Epoch 15/15
489/489 [==============================] - 81s 165ms/step - loss: 0.6185 - accuracy: 0.7543 - val_loss: 1.2597 - val_accuracy: 0.6059
```

```
Epoch 1/15
489/489 [==============================] - 78s 159ms/step - loss: 0.5981 - accuracy: 0.7685 - val_loss: 1.2410 - val_accuracy: 0.5916
Epoch 2/15
489/489 [==============================] - 80s 163ms/step - loss: 0.5587 - accuracy: 0.7930 - val_loss: 1.2594 - val_accuracy: 0.5943
Epoch 3/15
489/489 [==============================] - 81s 165ms/step - loss: 0.5047 - accuracy: 0.8195 - val_loss: 1.2006 - val_accuracy: 0.6086
Epoch 4/15
489/489 [==============================] - 81s 165ms/step - loss: 0.4243 - accuracy: 0.8548 - val_loss: 1.2553 - val_accuracy: 0.5934
Epoch 5/15
489/489 [==============================] - 81s 165ms/step - loss: 0.3198 - accuracy: 0.9025 - val_loss: 1.3205 - val_accuracy: 0.5813
Epoch 6/15
489/489 [==============================] - 81s 165ms/step - loss: 0.2145 - accuracy: 0.9501 - val_loss: 1.2407 - val_accuracy: 0.6023

Epoch 00006: ReduceLROnPlateau reducing learning rate to 6.25000029685907e-05.
Epoch 7/15
489/489 [==============================] - 81s 165ms/step - loss: 0.1393 - accuracy: 0.9789 - val_loss: 1.1789 - val_accuracy: 0.6095
Epoch 8/15
489/489 [==============================] - 81s 165ms/step - loss: 0.1079 - accuracy: 0.9902 - val_loss: 1.1672 - val_accuracy: 0.6140
Epoch 9/15
489/489 [==============================] - 81s 166ms/step - loss: 0.1004 - accuracy: 0.9910 - val_loss: 1.1874 - val_accuracy: 0.6122
Epoch 10/15
489/489 [==============================] - 81s 166ms/step - loss: 0.0887 - accuracy: 0.9933 - val_loss: 1.1790 - val_accuracy: 0.6149
Epoch 11/15
489/489 [==============================] - 81s 166ms/step - loss: 0.0789 - accuracy: 0.9944 - val_loss: 1.1745 - val_accuracy: 0.6216
Epoch 12/15
489/489 [==============================] - 81s 165ms/step - loss: 0.0702 - accuracy: 0.9964 - val_loss: 1.1689 - val_accuracy: 0.6234
Epoch 13/15
489/489 [==============================] - 81s 165ms/step - loss: 0.0668 - accuracy: 0.9968 - val_loss: 1.1768 - val_accuracy: 0.6176
Epoch 14/15
489/489 [==============================] - 81s 165ms/step - loss: 0.0602 - accuracy: 0.9976 - val_loss: 1.1784 - val_accuracy: 0.6234
Epoch 15/15
489/489 [==============================] - 81s 165ms/step - loss: 0.0557 - accuracy: 0.9980 - val_loss: 1.2128 - val_accuracy: 0.6099

Epoch 00015: ReduceLROnPlateau reducing learning rate to 3.125000148429535e-05.
```

We then visualized these result for the last 15 epoch for better analysis.

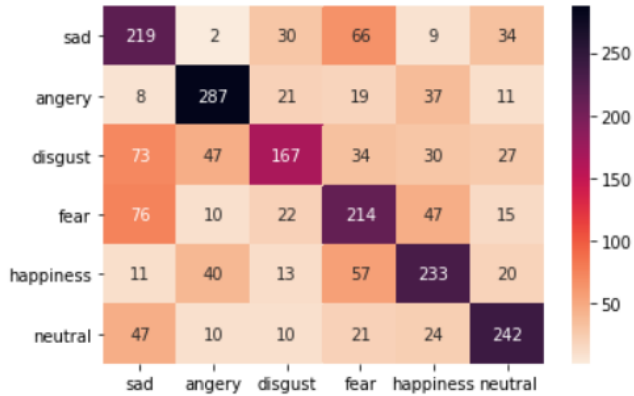




Finally, we computed different evaluation metrics

(Accuracy and F1-Score) and plotted the Confusion matrix.

	precision	recall	f1-score	support
0	0.50	0.61	0.55	360
1	0.72	0.75	0.74	383
2	0.63	0.44	0.52	378
3	0.52	0.56	0.54	384
4	0.61	0.62	0.62	374
5	0.69	0.68	0.69	354
accuracy			0.61	2233
macro avg	0.62	0.61	0.61	2233
weighted avg	0.62	0.61	0.61	2233



2. 1D model:

For our 1D model we created our own architecture and used the time domain feature space.

Here is the model's structure:

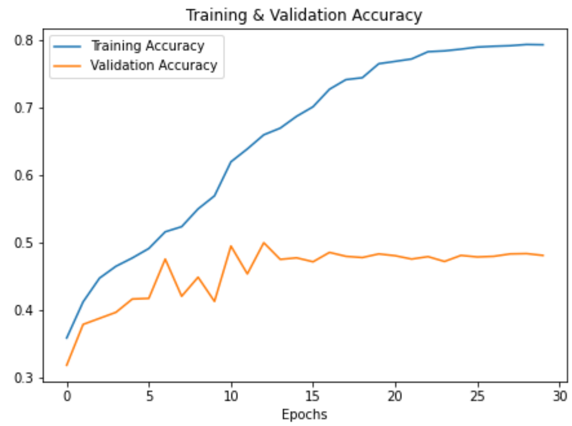
```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 2162, 512)         3072
_________________________________________________________________
module_wrapper (ModuleWrappe (None, 2162, 512)         2048
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 1081, 512)         0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 1081, 512)         1311232
_________________________________________________________________
module_wrapper_1 (ModuleWrap (None, 1081, 512)         2048
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 541, 512)          0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 541, 256)          655616
_________________________________________________________________
module_wrapper_2 (ModuleWrap (None, 541, 256)          1024
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 271, 256)          0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 271, 256)          196864
_________________________________________________________________
module_wrapper_3 (ModuleWrap (None, 271, 256)          1024
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 136, 256)          0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 136, 128)          98432
_________________________________________________________________
module_wrapper_4 (ModuleWrap (None, 136, 128)          512
_________________________________________________________________
max_pooling1d_4 (MaxPooling1 (None, 68, 128)           0
_________________________________________________________________
flatten (Flatten)            (None, 8704)              0
_________________________________________________________________
dense (Dense)                (None, 512)               4456960
_________________________________________________________________
module_wrapper_5 (ModuleWrap (None, 512)               2048
_________________________________________________________________
dense_1 (Dense)              (None, 6)                 3078
=================================================================
Total params: 6,733,958
Trainable params: 6,729,606
```

We ran the above model for 30 Epochs achieving the following results:

```
Epoch 1/30
489/489 [==============================] - 85s 154ms/step - loss: 1.6454 - accuracy: 0.3592 - val_loss: 4.1234 - val_accuracy: 0.3189
Epoch 2/30
489/489 [==============================] - 80s 164ms/step - loss: 1.4470 - accuracy: 0.4127 - val_loss: 1.8234 - val_accuracy: 0.3793
Epoch 3/30
489/489 [==============================] - 80s 164ms/step - loss: 1.3799 - accuracy: 0.4478 - val_loss: 1.8939 - val_accuracy: 0.3883
Epoch 4/30
489/489 [==============================] - 80s 164ms/step - loss: 1.3384 - accuracy: 0.4653 - val_loss: 1.5701 - val_accuracy: 0.3972
Epoch 5/30
489/489 [==============================] - 80s 164ms/step - loss: 1.2980 - accuracy: 0.4777 - val_loss: 1.9052 - val_accuracy: 0.4169
Epoch 6/30
489/489 [==============================] - 80s 164ms/step - loss: 1.2580 - accuracy: 0.4916 - val_loss: 1.5721 - val_accuracy: 0.4178
Epoch 7/30
489/489 [==============================] - 80s 164ms/step - loss: 1.2155 - accuracy: 0.5163 - val_loss: 1.5264 - val_accuracy: 0.4760
Epoch 8/30
489/489 [==============================] - 80s 164ms/step - loss: 1.1851 - accuracy: 0.5240 - val_loss: 1.5190 - val_accuracy: 0.4210
Epoch 9/30
489/489 [==============================] - 80s 164ms/step - loss: 1.1342 - accuracy: 0.5503 - val_loss: 1.4902 - val_accuracy: 0.4492
Epoch 10/30
489/489 [==============================] - 80s 164ms/step - loss: 1.0865 - accuracy: 0.5696 - val_loss: 2.3146 - val_accuracy: 0.4133

Epoch 00010: ReduceLROnPlateau reducing learning rate to 0.0005000000237487257.
Epoch 11/30
489/489 [==============================] - 80s 164ms/step - loss: 0.9608 - accuracy: 0.6200 - val_loss: 1.6666 - val_accuracy: 0.4953
Epoch 12/30
489/489 [==============================] - 80s 164ms/step - loss: 0.9129 - accuracy: 0.6391 - val_loss: 1.7164 - val_accuracy: 0.4541
Epoch 13/30
489/489 [==============================] - 80s 164ms/step - loss: 0.8683 - accuracy: 0.6599 - val_loss: 1.8913 - val_accuracy: 0.5002
Epoch 14/30
489/489 [==============================] - 80s 164ms/step - loss: 0.8355 - accuracy: 0.6698 - val_loss: 2.0099 - val_accuracy: 0.4756
Epoch 15/30
489/489 [==============================] - 80s 164ms/step - loss: 0.8012 - accuracy: 0.6873 - val_loss: 1.9278 - val_accuracy: 0.4778
Epoch 16/30
489/489 [==============================] - 80s 164ms/step - loss: 0.7666 - accuracy: 0.7015 - val_loss: 1.9394 - val_accuracy: 0.4720

Epoch 00016: ReduceLROnPlateau reducing learning rate to 0.0002500000118743628.
Epoch 17/30
489/489 [==============================] - 80s 164ms/step - loss: 0.6950 - accuracy: 0.7274 - val_loss: 2.0523 - val_accuracy: 0.4859
Epoch 18/30
489/489 [==============================] - 80s 164ms/step - loss: 0.6676 - accuracy: 0.7416 - val_loss: 2.0800 - val_accuracy: 0.4801
```

```
Epoch 19/30
489/489 [==============================] - 80s 164ms/step - loss: 0.6513 - accuracy: 0.7444 - val_loss: 2.4364 - val_accuracy: 0.4783

Epoch 00019: ReduceLROnPlateau reducing learning rate to 0.0001250000059371814.
Epoch 20/30
489/489 [==============================] - 80s 164ms/step - loss: 0.6095 - accuracy: 0.7652 - val_loss: 2.1757 - val_accuracy: 0.4837
Epoch 21/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5978 - accuracy: 0.7686 - val_loss: 2.3189 - val_accuracy: 0.4810
Epoch 22/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5942 - accuracy: 0.7722 - val_loss: 2.2576 - val_accuracy: 0.4760

Epoch 00022: ReduceLROnPlateau reducing learning rate to 6.25000029685907e-05.
Epoch 23/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5687 - accuracy: 0.7829 - val_loss: 2.2731 - val_accuracy: 0.4796
Epoch 24/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5628 - accuracy: 0.7842 - val_loss: 2.3682 - val_accuracy: 0.4725
Epoch 25/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5618 - accuracy: 0.7867 - val_loss: 2.2949 - val_accuracy: 0.4814

Epoch 00025: ReduceLROnPlateau reducing learning rate to 3.125000148429535e-05.
Epoch 26/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5475 - accuracy: 0.7898 - val_loss: 2.3093 - val_accuracy: 0.4792
Epoch 27/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5482 - accuracy: 0.7910 - val_loss: 2.3505 - val_accuracy: 0.4801
Epoch 28/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5454 - accuracy: 0.7918 - val_loss: 2.3653 - val_accuracy: 0.4837

Epoch 00028: ReduceLROnPlateau reducing learning rate to 1.5625000742147677e-05.
Epoch 29/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5388 - accuracy: 0.7936 - val_loss: 2.3656 - val_accuracy: 0.4841
Epoch 30/30
489/489 [==============================] - 80s 164ms/step - loss: 0.5374 - accuracy: 0.7932 - val_loss: 2.3407 - val_accuracy: 0.4814
```

We then visualized these result for better analysis.



Finally, we computed different evaluation metrics

(Accuracy and F1-Score) and plotted the Confusion matrix.

	precision	recall	f1-score	support
0.0	0.47	0.40	0.43	382
1.0	0.61	0.69	0.65	381
2.0	0.44	0.44	0.44	381
3.0	0.40	0.39	0.40	381
4.0	0.47	0.46	0.47	381
5.0	0.47	0.52	0.49	327
accuracy			0.48	2233
macro avg	0.48	0.48	0.48	2233
weighted avg	0.48	0.48	0.48	2233