Pattern Recognition – Spring 2023

Speech Emotion Recognition

AHMED DUSUKI	6856
MANAR AMGAD	7113
NADA ELWAZANE	6876

Paper Implementation

We attempt to implement the feature extraction and the model architecture used in the following paper.

Convolutional Neural Network (CNN) Based Speech-Emotion Recognition.

Alif Bin Abdul Qayyum, Asiful Arefeen*, Celia Shahnaz
Department of Electrical and Electronic Engineering,
Bangladesh University of Engineering and Technology, Dhaka 1205, Bangladesh.
*E-mail: asifularefeen1234@gmail.com

The paper used a CNN based approach without input data stream preprocessing to achieve superior accuracy over other work.

TABLE II: Accuracies for MVR, SVM, RNN and proposed CNN methods.

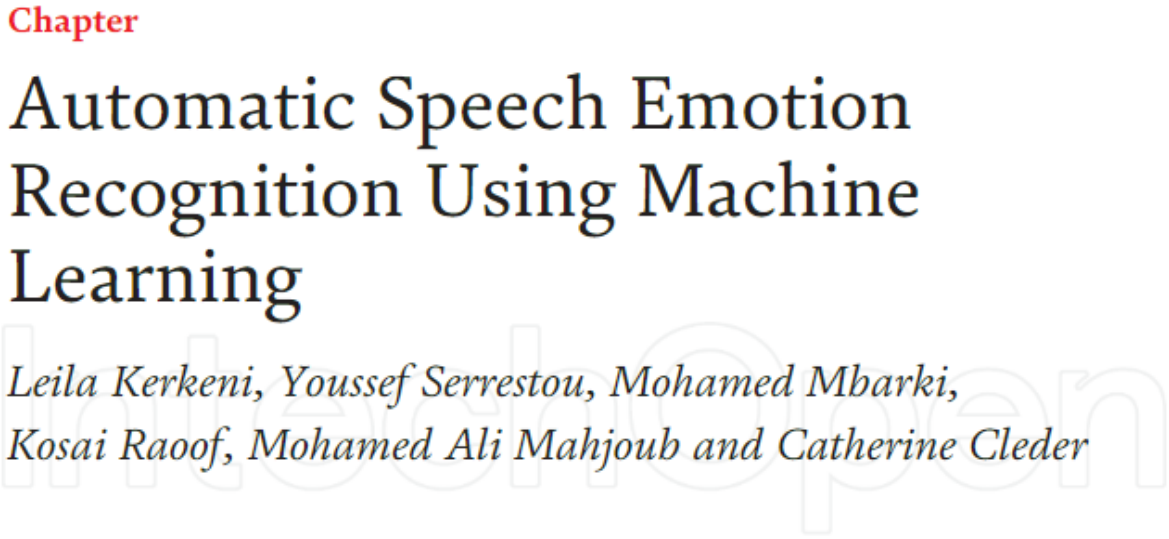
Feature	Method	A	D	F	H	N	SA	SU	AVG
MFCC	MVR	45.68	45.44	42.79	67.08	59.43	69.91	71.94	57.47
MS		41.93	41.03	39.97	68.97	52.17	61.11	69.87	53.58
MFCC+MS		58.47	57.59	60.04	75.13	68.74	79.26	83.07	68.90
MFCC	SVM	53.17	57.31	48.97	79.81	68.34	78.19	81.48	66.75
MS		49.95	52.31	49.79	77.79	63.71	72.13	78.91	63.51
MFCC+MS		69.87	72.69	71.41	78.18	75.47	82.63	85.14	76.48
MFCC	RNN	61.87	57.47	53.98	74.81	68.31	78.87	82.67	68.28
MS		53.85	61.13	52.36	82.69	78.22	72.35	78.84	68.49
MFCC+MS		70.32	75.22	77.54	78.19	79.47	87.17	86.47	79.20
	CNN	73.50	80.88	80.12	87.38	86.38	89.75	87.25	83.61

Methodology:

1- Feature Extraction:

In the feature extraction based process, modulation spectral features and MFCC have been selected.

This is similar feature extraction to what is described in the cited paper:


Chapter

Automatic Speech Emotion Recognition Using Machine Learning

Leila Kerkeni, Youssef Serrestou, Mohamed Mbarki, Kosai Raoof, Mohamed Ali Mahjoub and Catherine Cleder

MFCC:

The paper describes extracting by transforming the signal to the frequency domain using fourier transform and then mapping onto the Mel-frequency scale. Where the first 12 DCT coefficients of the Mel log energies are used.

MSF:

For MSF, the speech signal is decomposed by an auditory filter bank and the Hilbert envelopes of the critical-band outputs are computed to form modulation signals. A modulation filter bank is applied to these signals to obtain modulation spectra, which are used as MSF features.

We made 2 attempts at extracting the MSF as it was not clear how it was implemented.

After extracting the feature, speaker normalization is applied, removing the mean of the features and normalizing to unit variance.

Input to the model was raw audio withd uration of 8 seconds. The audio files with duration less than 8 seconds was zero padded.

Model Architecture:

The paper describes a 7 layered 1D CNN model, listing the number of filters and the kernel sizes of each layer. One noticeable point is that there was no mention of dropout layers, only batch normalization layers. The model's last dense layer had a node count of 6 or 7 depending on the dataset used.

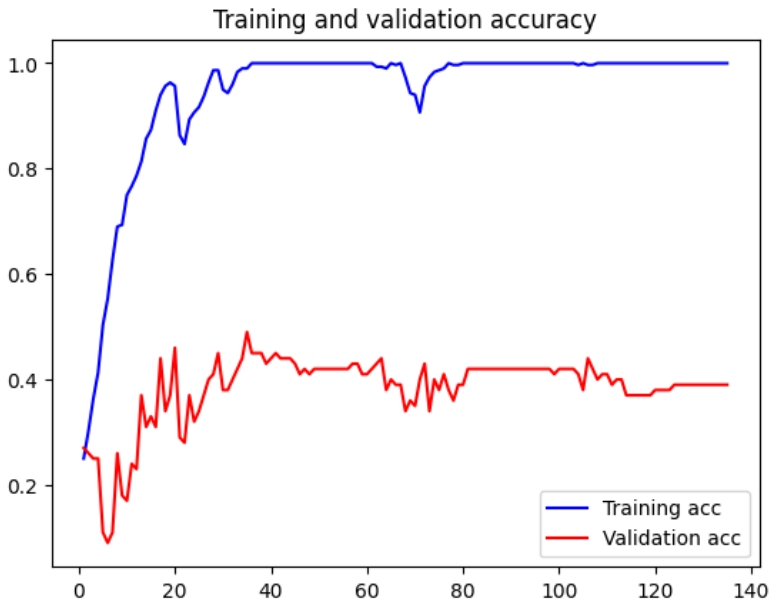
For training, Adam optimizer wasused with an initial learning rate of 0.001, beta 1 value of 0.9 and beta 2 value of 0.999. Loss function used for thispurpose is 'Categorical Crossentropy'.

```
def create_model():
    model=Sequential()
    model.add(Conv1D(32, kernel_size=21, strides=1, padding='same', activation='relu',
                     input_shape=(X_train_normalized.shape[1], 1)))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(64, kernel_size=19, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(128, kernel_size=17, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(256, kernel_size=15, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(512, kernel_size=13, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(1024, kernel_size=11, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(MaxPooling1D(pool_size=2, strides = 2, padding = 'same'))

    model.add(Conv1D(1024, kernel_size=9, strides=1, padding='same', activation='relu'))
    model.add(BatchNormalization())
    model.add(GlobalMaxPooling1D())

    model.add(Flatten())
    model.add(Dense(128, activation='relu'))

    model.add(Dense(6, activation='softmax'))
    opt=tensorflow.keras.optimizers.Adam(
      learning_rate=0.001,
      beta_1=0.9,
      beta_2=0.999,
  )
    model.compile(loss = 'categorical_crossentropy',optimizer=opt,metrics = ['accuracy',get_f1])
    return model
```

Attempt 1:

1- SAVEE:

As the paper used the SAVEE dataset, the first attempt was to try to implement it using the same dataset.
Used similar data splitting (300 training, 100 validation, 80 testing).
Batch size = 16.



As shown on the previous figure, the model was quickly overfitting on the training data, this is most likely due to the model's complexity as well as the limited training data. Reaching a maximuim 100% training accuracy and a maximum 49% validation accuracy.

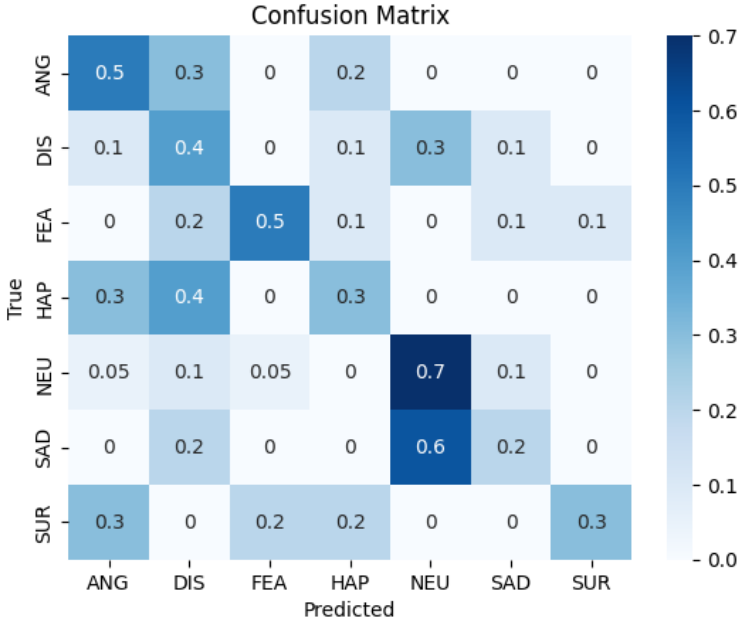
Trying higher batch size such as 32, 64, 128, did not lead to better results, infact lower batch size 16, 8, even 4, was the best.

<u>Testing result:</u>

An accuracy of only 45% was achieved, this is expected looking at the training and validation accuracies during training which showed the model quickly overfitting the data.

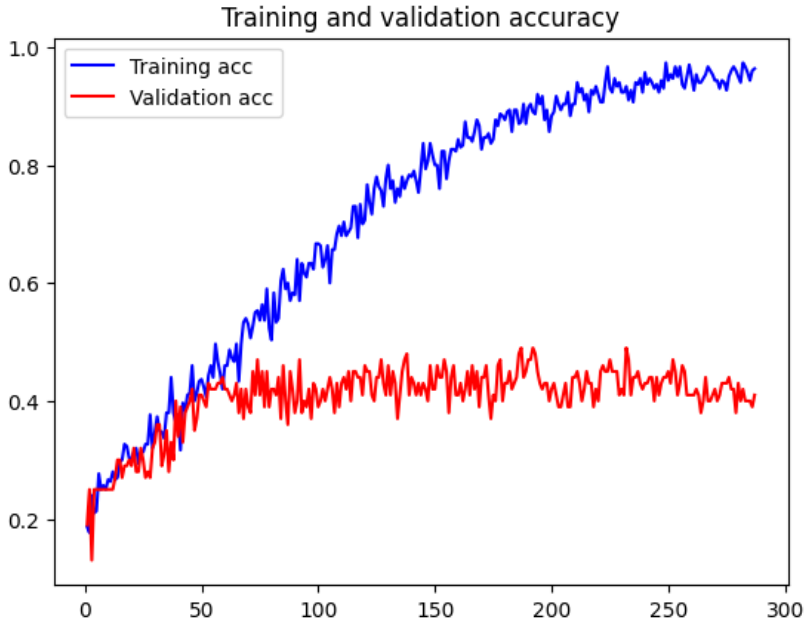
```
5/5 [==============================] - 0s 8ms/step
              precision    recall  f1-score   support

           0       0.38      0.50      0.43        10
           1       0.24      0.40      0.30        10
           2       0.62      0.50      0.56        10
           3       0.33      0.30      0.32        10
           4       0.61      0.70      0.65        20
           5       0.33      0.20      0.25        10
           6       0.75      0.30      0.43        10

    accuracy                           0.45        80
   macro avg       0.47      0.41      0.42        80
weighted avg       0.48      0.45      0.45        80
```

The model was able to predict neutral emotion correctly most of the time at 70% accuracy, but it also incorrectly guessed neutral for sad emotion frequently, making it the least accurate emotion.



Dropouts:

To attempt to avoid overfitting, we tried adding dropouts to the model.



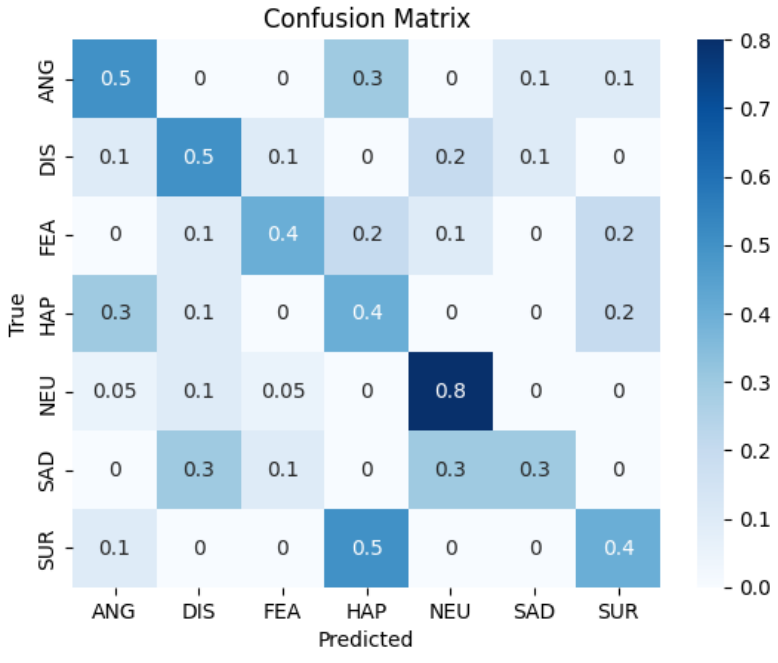
This did slow down the overfitting of the model onto the training data, but did not improve the best validation accuracy reached at 49%.

Testing result:

Significantly higher test accuracy at 51% compared to previous 45%.

```
5/5 [==============================] - 10s 6ms/step
              precision    recall  f1-score   support

           0       0.45      0.50      0.48        10
           1       0.42      0.50      0.45        10
           2       0.57      0.40      0.47        10
           3       0.29      0.40      0.33        10
           4       0.73      0.80      0.76        20
           5       0.60      0.30      0.40        10
           6       0.44      0.40      0.42        10

    accuracy                           0.51        80
   macro avg       0.50      0.47      0.47        80
weighted avg       0.53      0.51      0.51        80
```

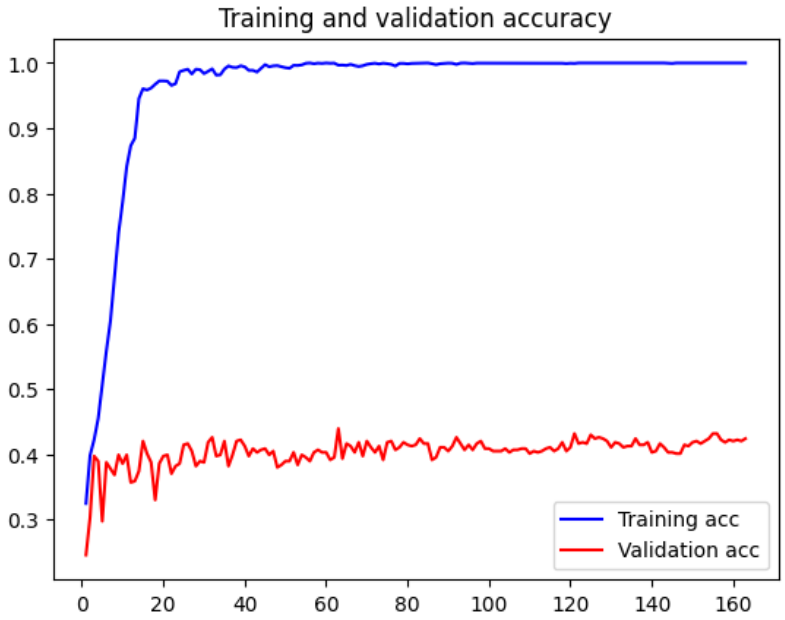
Neutral emotion still has the best accuracy, the most confusing being sad again.



2- CREMA:

Trying on CREMA dataset, with data splitting 66.5% training, 3.5% validation, 30% testing.

Batch size = 32 (due to much larger training data count, ~4700).

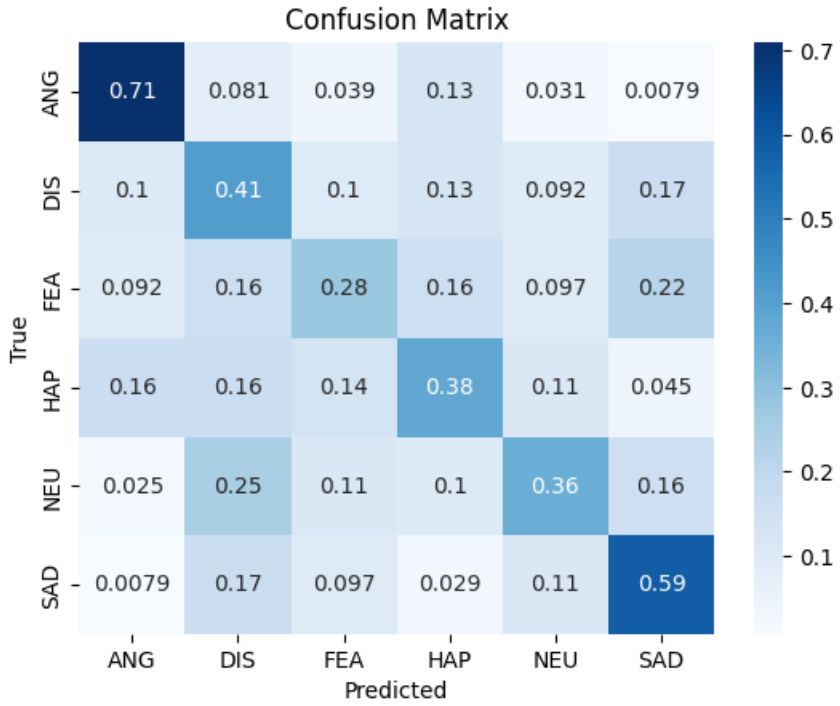


Surprisingly even when using a larger training dataset, the model still overfit relatively quickly to the training data, reaching a maximum 100% training accuracy, and a maximum 43.95% validation accuracy.

Testing result:

```
140/140 [==============================] - 1s 6ms/step
              precision    recall  f1-score   support

           0       0.65      0.71      0.68       382
           1       0.35      0.41      0.38       381
           2       0.37      0.28      0.32       381
           3       0.42      0.38      0.40       382
           4       0.41      0.36      0.38       326
           5       0.51      0.59      0.54       381

    accuracy                           0.46      2233
   macro avg       0.45      0.45      0.45      2233
weighted avg       0.45      0.46      0.45      2233
```

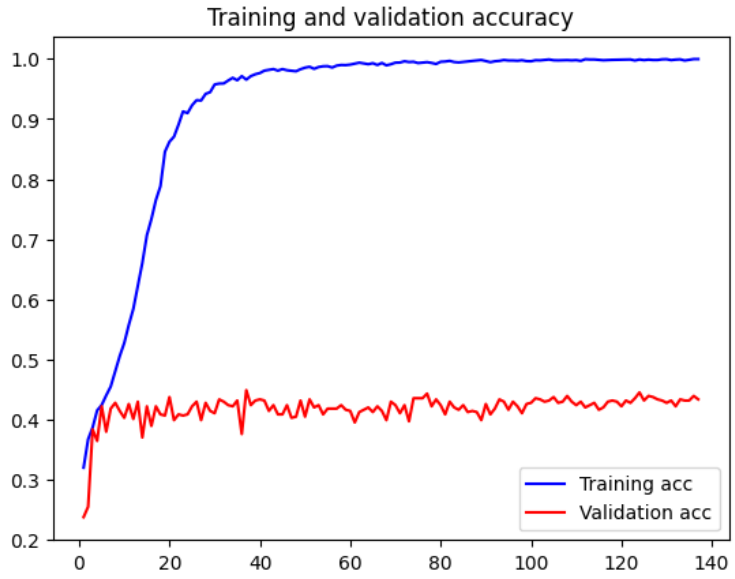
A 46% testing accuracy was measured, with 45% F1 score.



Anger was the easiest emotion to discern, with fear being the hardest.

Drouputs:

We tried 2 different dropout models, the one with less dropout performed better, as follows:



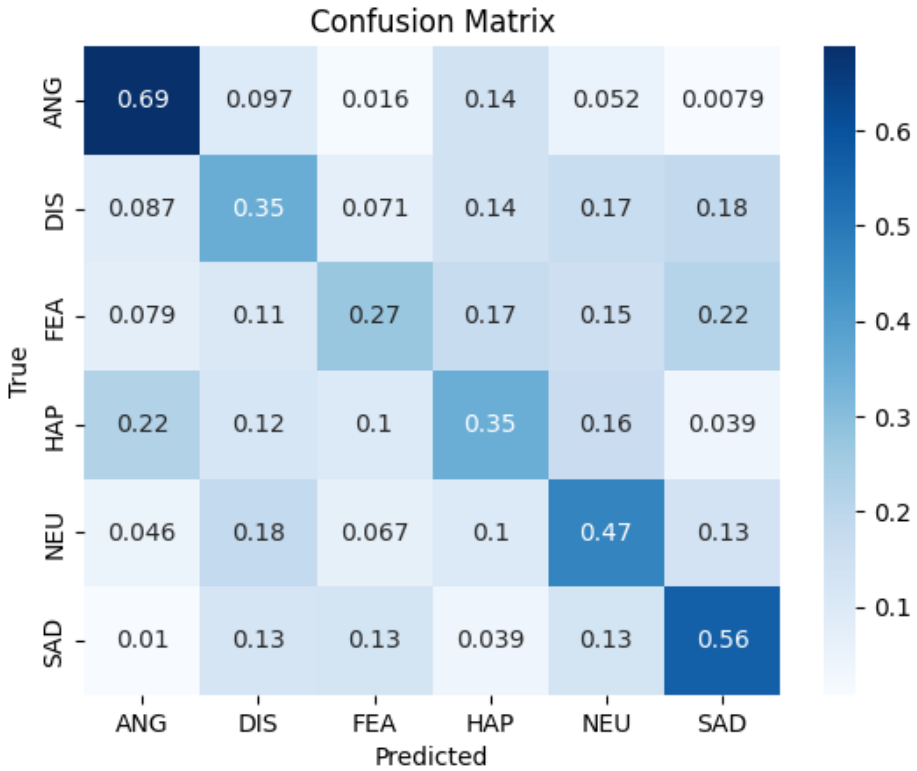
It didn't appear to have much benefit.

Testing result:

45% testing accuracy was achieved, slightly worse than with no dropouts, no significant improvement was found.

```
140/140 [==============================] - 1s 5ms/step
              precision    recall  f1-score   support

           0       0.61      0.69      0.65       382
           1       0.37      0.35      0.36       381
           2       0.42      0.27      0.33       381
           3       0.38      0.35      0.36       382
           4       0.37      0.47      0.42       326
           5       0.50      0.56      0.53       381

    accuracy                           0.45      2233
   macro avg       0.44      0.45      0.44      2233
weighted avg       0.44      0.45      0.44      2233
```

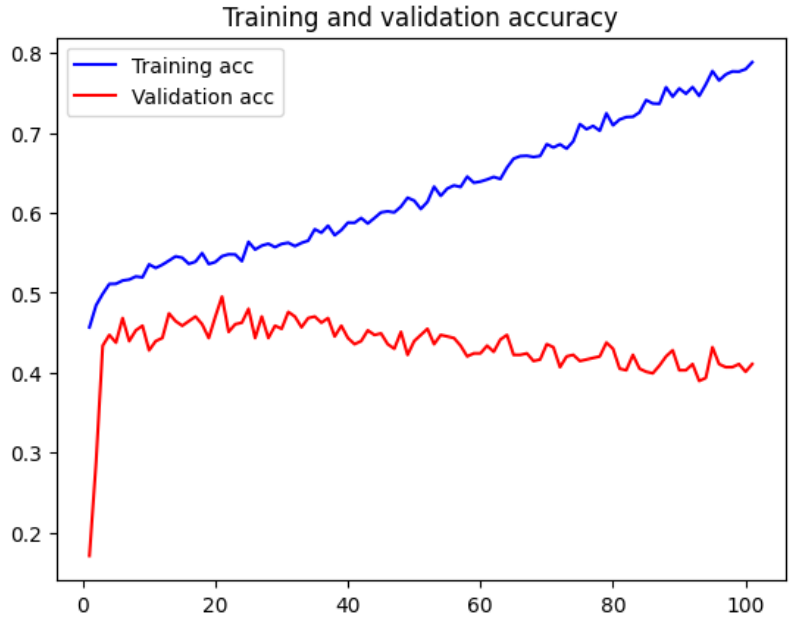
Similar confusion matrix to no dropout model.



Dimensionality reduction:

We made a few attempts at dimensionality reduction (LDA=5,LDA=3,Mutual Information=10, Mutual Information=3), having the best result with LDA n=5.

The model had a maximum validation accuracy of 49.5%.

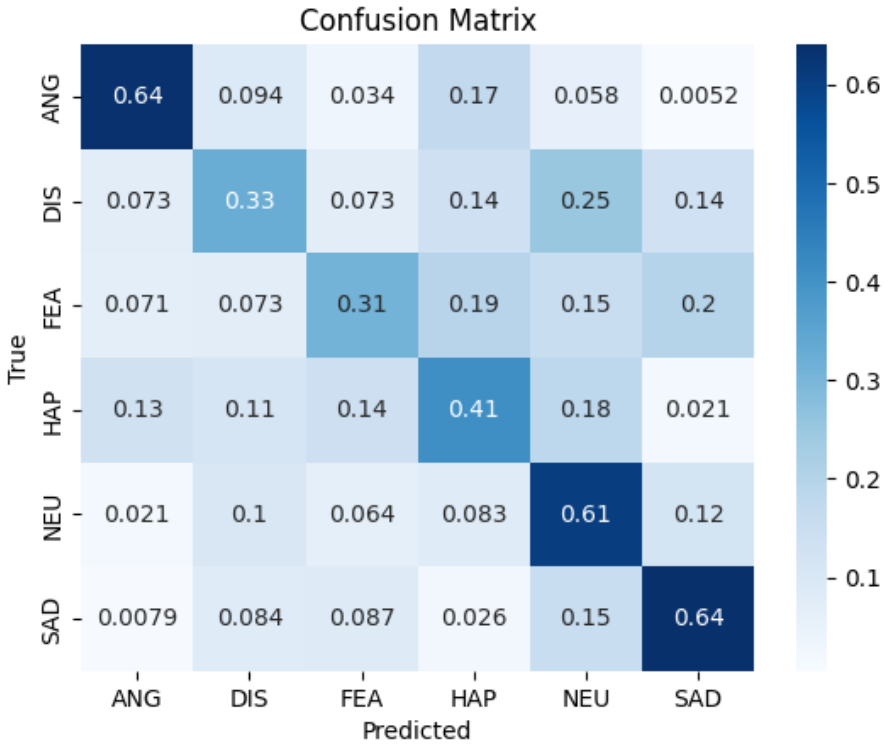


Testing result:

At 48% testing accuracy it was a small improvement.

```
140/140 [==============================] - 1s 4ms/step
              precision    recall  f1-score   support

           0       0.68      0.64      0.66       382
           1       0.42      0.33      0.37       381
           2       0.44      0.31      0.36       381
           3       0.40      0.41      0.41       382
           4       0.39      0.61      0.48       326
           5       0.58      0.64      0.61       381

    accuracy                           0.48      2233
   macro avg       0.49      0.49      0.48      2233
weighted avg       0.49      0.48      0.48      2233
```

Confusion matrix showing anger having less of a lead and fear emotion having closer accuracy to the second worst disgust emotion.



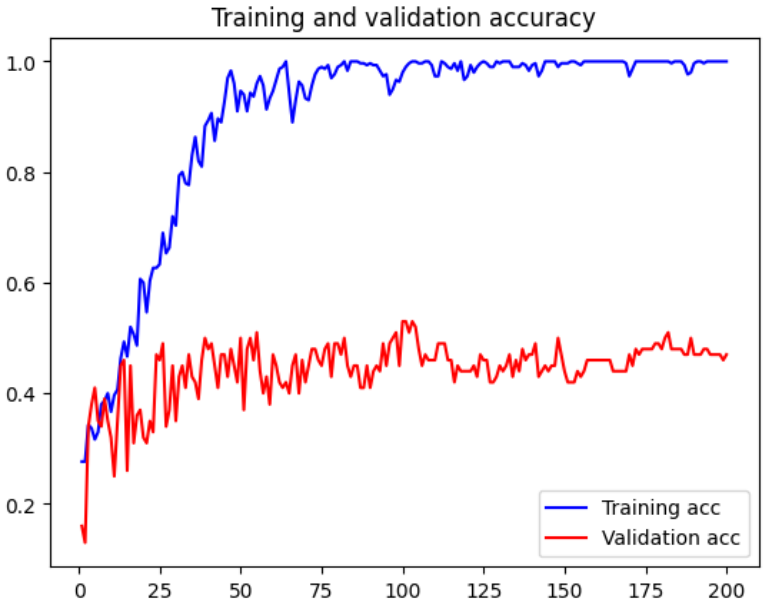
Attempt 2:

Since the there were no implementation detail for the MSF extraction, we made a second slightly different attempt.

1- SAVEE:

Data splitting (300 training, 100 validation, 80 testing).

Batch size = 4. (8 and 16 had similar results, 32, 64 and up had worse results)

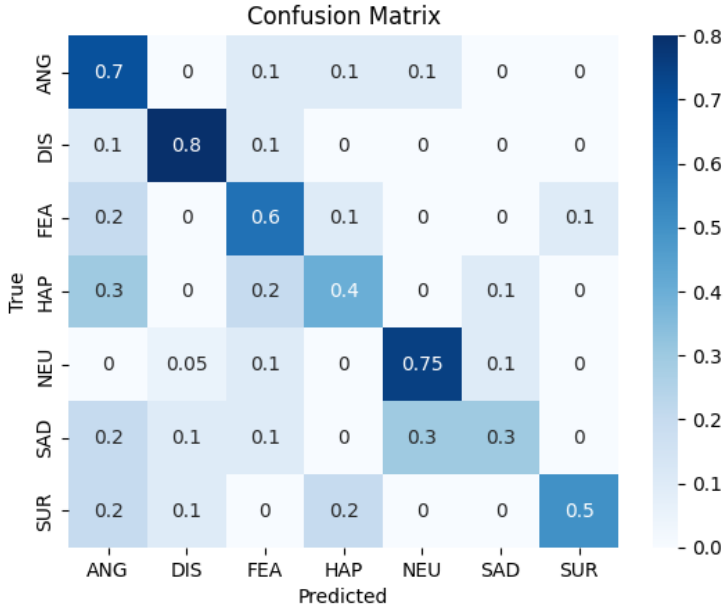


The model still overfits similarly to the previous attempt. However, it reached a higher 53% validation accuracy.

Testing result:

```
5/5 [==============================] - 0s 8ms/step
              precision    recall  f1-score   support

           0       0.41      0.70      0.52        10
           1       0.73      0.80      0.76        10
           2       0.46      0.60      0.52        10
           3       0.50      0.40      0.44        10
           4       0.79      0.75      0.77        20
           5       0.50      0.30      0.37        10
           6       0.83      0.50      0.62        10

    accuracy                           0.60        80
   macro avg       0.60      0.58      0.57        80
weighted avg       0.63      0.60      0.60        80
```

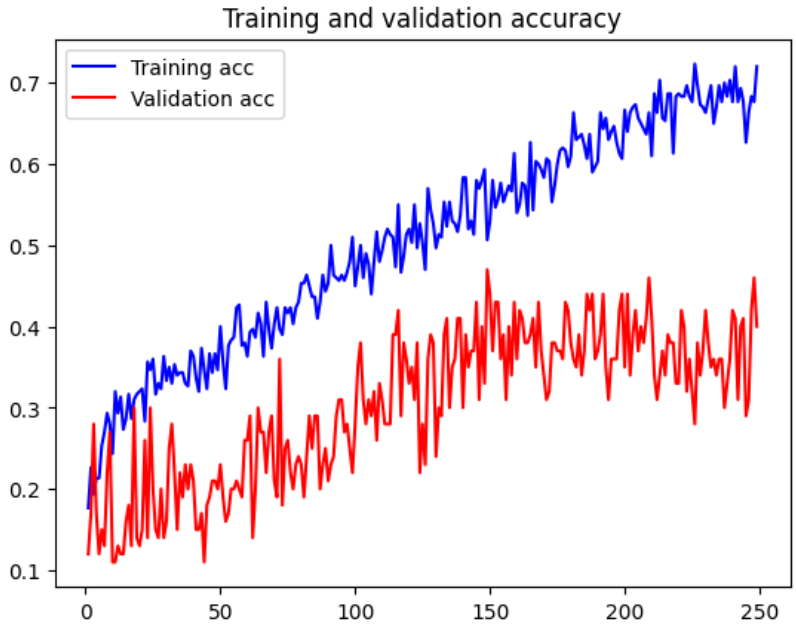
A testing accuracy of 60% with a weighted avg F1 score of 60%, this is the best attempt.



Confuision matrix now shows disgust as the best emotion accuracy, followed by neutral, sad is the worse, closely followed by happy.

Dropouts:

The dropouts did seem to make the model overfit less on the training data, but it didn't improve the validation accuracy reaching only 47%.

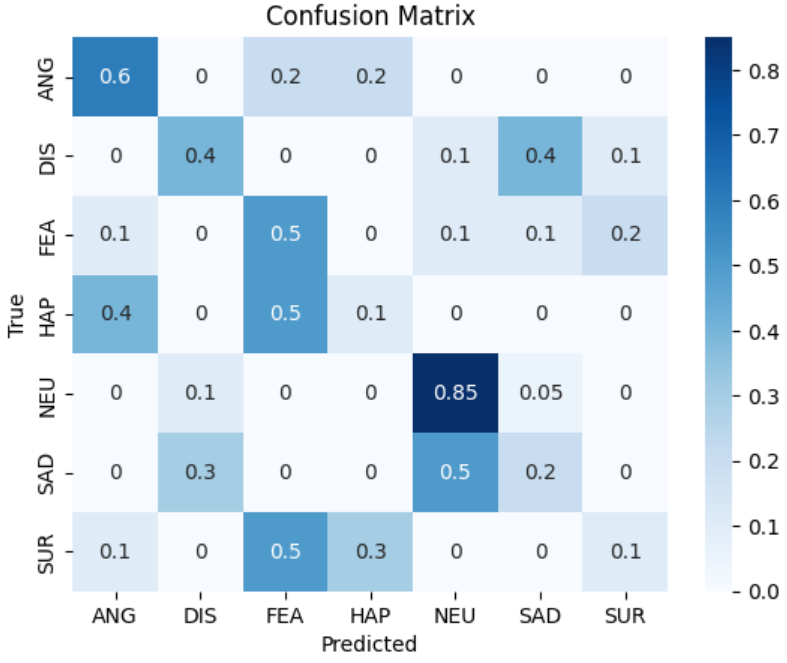


Testing result:

The testing accuracy was only 45%, the dropouts lead to a worse model.

```
5/5 [==============================] - 0s 8ms/step
              precision    recall  f1-score   support

           0       0.50      0.60      0.55        10
           1       0.44      0.40      0.42        10
           2       0.29      0.50      0.37        10
           3       0.17      0.10      0.12        10
           4       0.71      0.85      0.77        20
           5       0.25      0.20      0.22        10
           6       0.25      0.10      0.14        10

    accuracy                           0.45        80
   macro avg       0.37      0.39      0.37        80
weighted avg       0.42      0.45      0.42        80
```

The confusion matrix shows the model was very bad at identifying happy, surprise, and sad emotions.

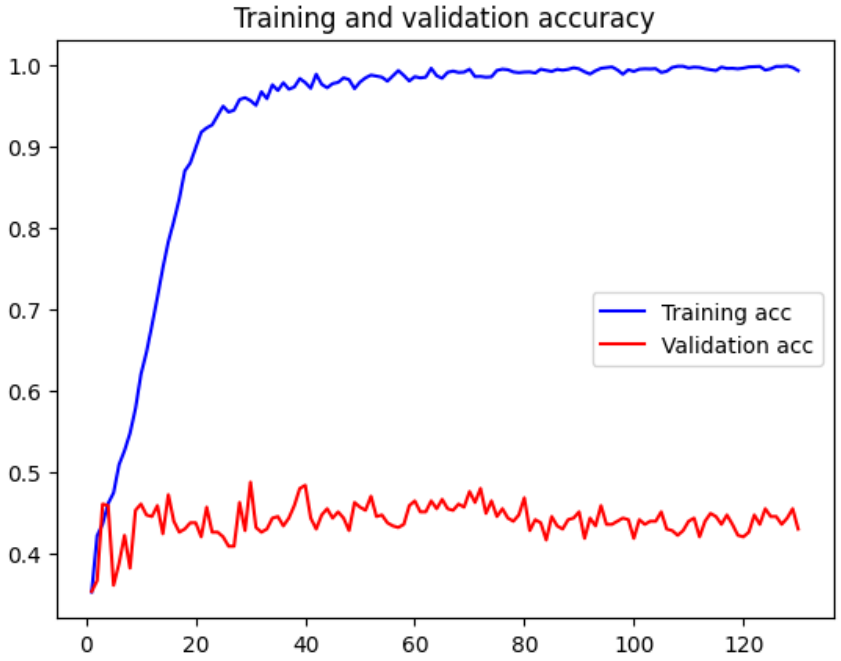


2- CREMA:

Data splitting 66.5% training, 3.5% validation, 30% testing.

Batch size = 16 (due to much larger training data count, ~4700).

A maximum validation accuracy of 48.75% was reached.

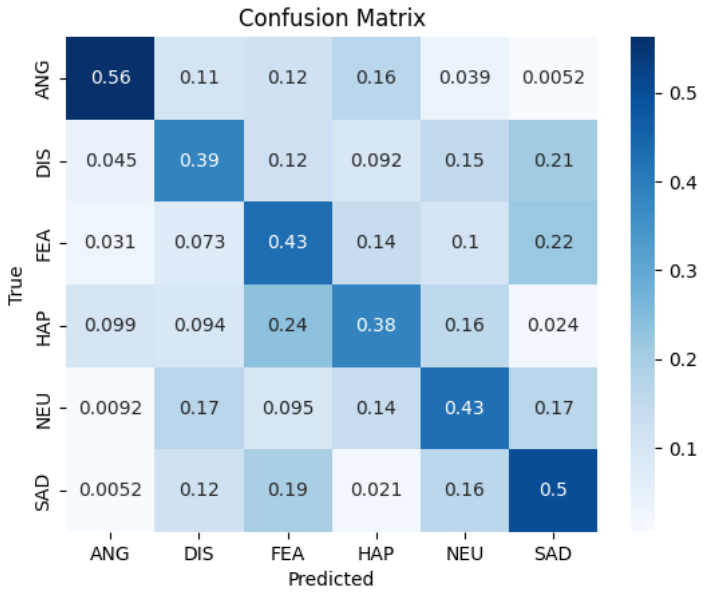


Testing accuracy:

A testing accuracy of only 45% was achieved (1% lower than the first attempt).

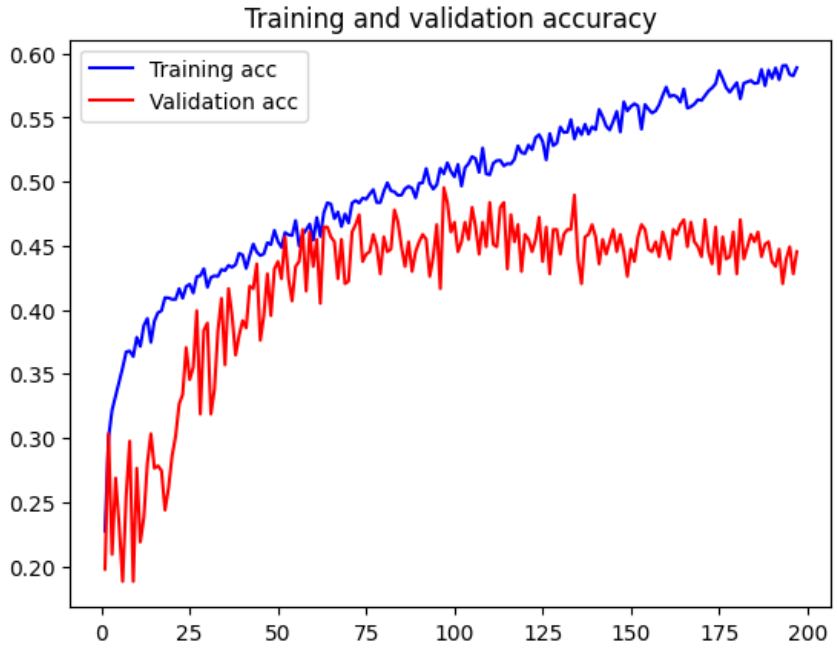
```
140/140 [==============================] - 1s 5ms/step
              precision    recall  f1-score   support

           0       0.75      0.56      0.64       382
           1       0.42      0.39      0.40       381
           2       0.36      0.43      0.39       381
           3       0.42      0.38      0.40       382
           4       0.37      0.43      0.40       326
           5       0.46      0.50      0.48       381

    accuracy                           0.45      2233
   macro avg       0.46      0.45      0.45      2233
weighted avg       0.46      0.45      0.45      2233
```

Similar confusion matrix, with anger being the best emotion accuracy.



Dropouts:

The dropouts decreased the overfitting but it still only reached 49.5% validation accuracy.



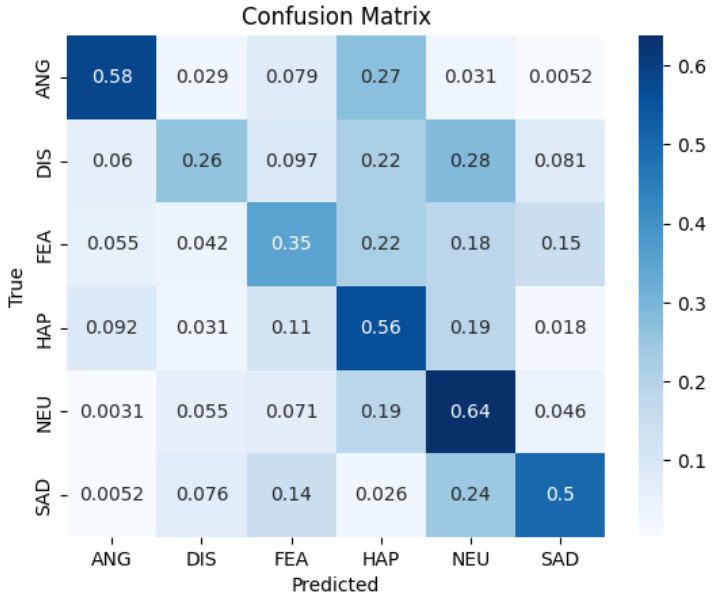

```
140/140 [==============================] - 1s
5ms/step
```

Testing result:

48% testing accuracy, similar to attempt 1 with LDA reduction.

	precision	recall	f1-score	support
0	0.73	0.58	0.65	382
1	0.54	0.26	0.35	381
2	0.42	0.35	0.38	381
3	0.39	0.56	0.46	382
4	0.37	0.64	0.47	326
5	0.63	0.50	0.56	381
accuracy			0.48	2233
macro avg	0.51	0.48	0.48	2233
weighted avg	0.52	0.48	0.48	2233

Neutral is now the emotion with the highest accuracy, followed by both happy and anger, the worst being disgust.



Conclusion:

Using the second MSF extraction method resulted in significantly better testing accuracy on the SAVEE dataset 45% → 60%, but not a significant improvement on the CREMA dataset. It is clear that the details of the MSF extraction are important and have a large effect on the model's performance, it's likely the paper extracted it in a different way that resulted in their superior accuracy.

It's also worth noting that the SAVEE dataset was recorded from 4 native english male speakers aged 27 to 31 years. On the other hand, CREMA-D is from 91 actors, 48 male, 43 female, ages 20 to 74, from a variety of races and ethnicities.

This would make it easier to achieve higher accuracies using the SAVEE dataset.