A brief Intro

What we have and what we do.

- ➔ **The RAVDESS dataset**
  Ryerson Audio-Visual Database of Emotional **Speech** and Song. 24 actors, 60 trials per actor. 1440 files.
- ➔ **Eight distinct emotions**
  neutral, calm, happy, sad, angry, fearful, disgust, surprised. 96 neutral samples, 192 samples for each others.
- ➔ **A CNN approach**
  Build a CNN network to train the classifier and test its accuracy.

How to distinguish these eight emotions effectively?

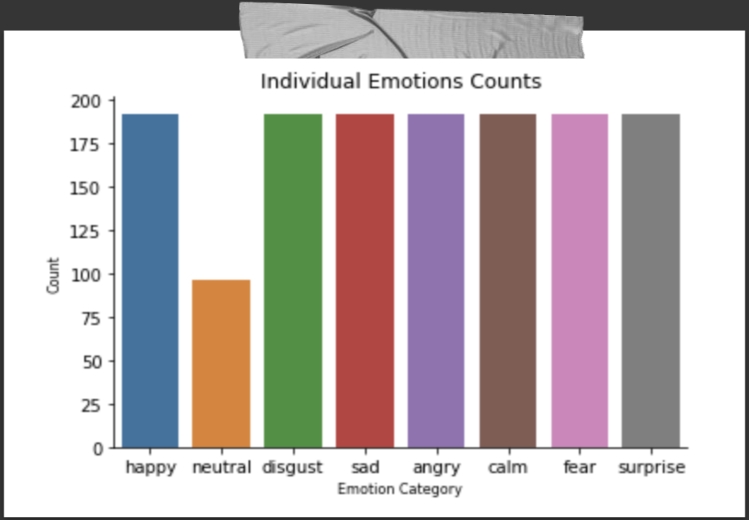


Figure showing the **emotion counts** of the investigated dataset.

We tried two approaches.

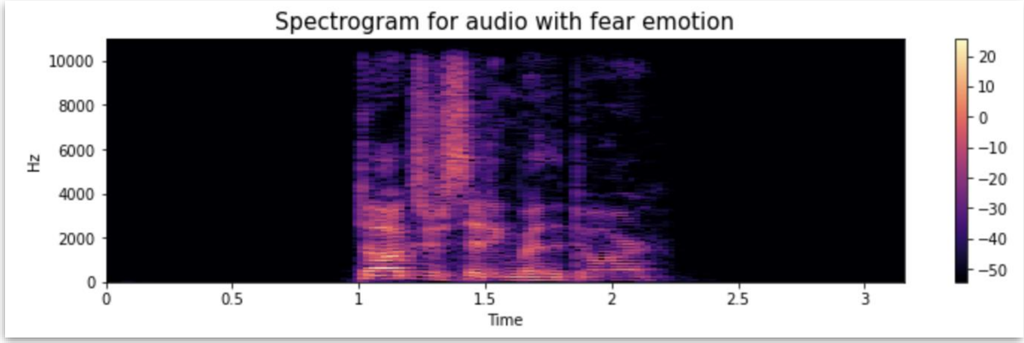
Based on Fequency Spectrogram

Based on Mel-Frequency Cepstral Coefficients (MFCCs)

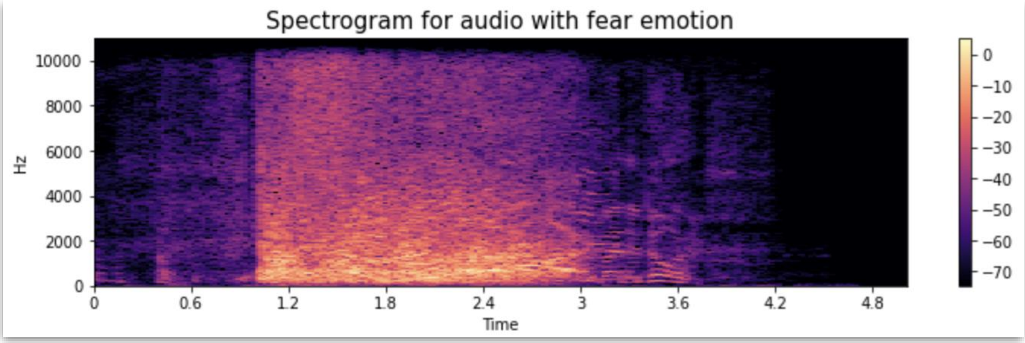
Spoiler alert!

We adopted the MFCCs to create figures used as the input data for the CNN networks. The reason for this will be explained later.

Fequency Spectrogram: Merge merge merge!

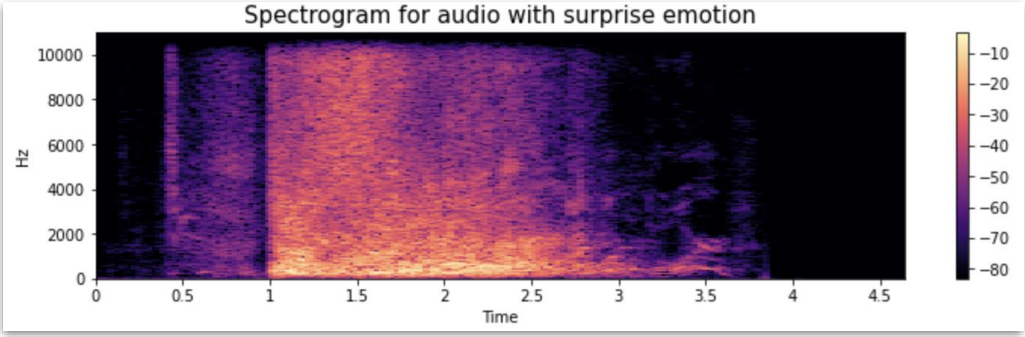


A spectrogram of **one speech clip** with a fear emotion.

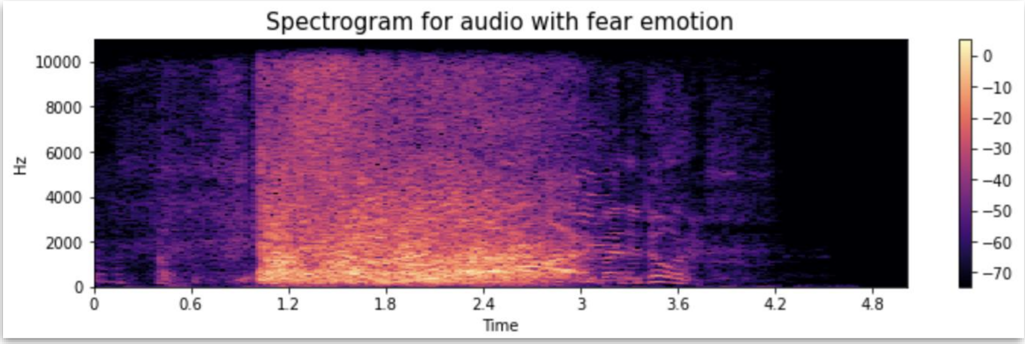


A spectrogram of the merging of **all speech clips** with a fear emotion.

Fequency Spectrogram: Different, but too much noise.



A spectrogram of the merging of all speech clips with a **surprise** emotion.



A spectrogram of the merging of all speech clips with a **fear** emotion.

Data Transformation

The process of transforming Audio waves to MFCC Spectrograms.

- ➔ **Audio(.wav)**

  Time Domain
  Original Human Speech Data

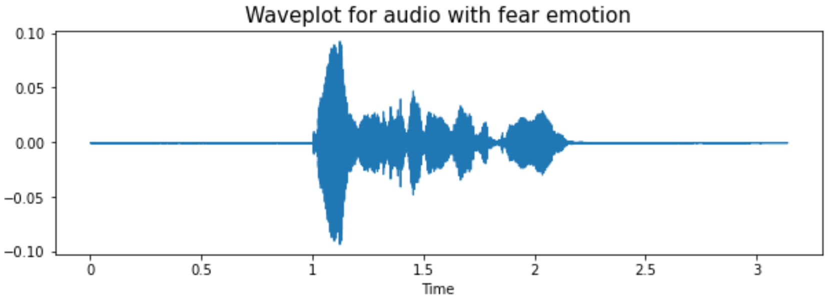
- ➔ **MFCC Spectrograms(.png)**

  Time Domain
  Image that contains all the features of the audio

Audio Wave

Normalization

Cut around all examples to 3 seconds



Example of a waveplot of an original audio file

Why not use Frequency Spectrograms?

Frequency spectrograms also contains features. However, humans are not able to tell the difference between these frequency domain features.

Thus, we introduce MFCC.

FFT

Frequencies

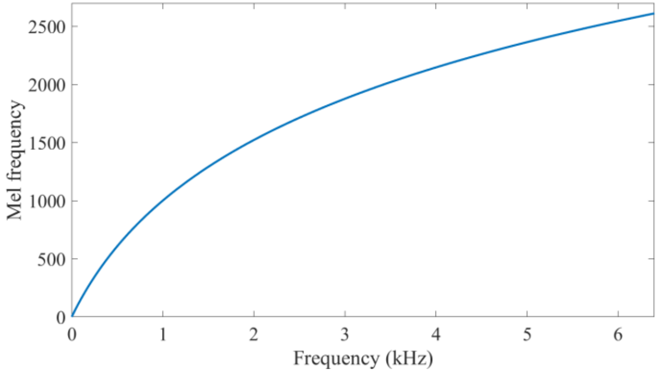
Transform all from time domain to frequency domain

Using STFT and calculating energy to generate Frequency Spectrograms

MFCC

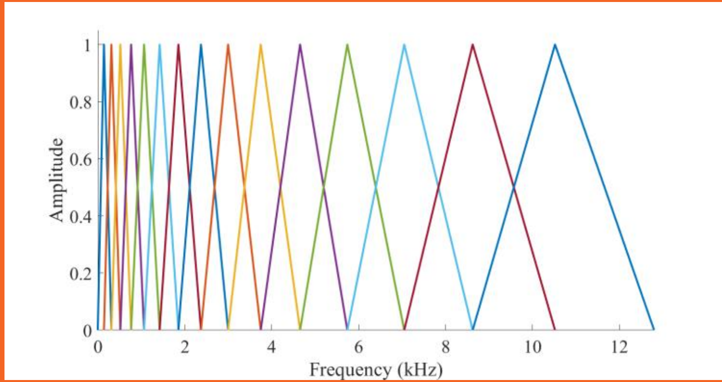
Mel-scale makes the spectrogram more human auditory-liked.

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right).$$



https://wiki.aalto.fi/display/ITSP/Cepstrum+and+MFCC

Triangular overlapping windows also leave more low frequencies information.



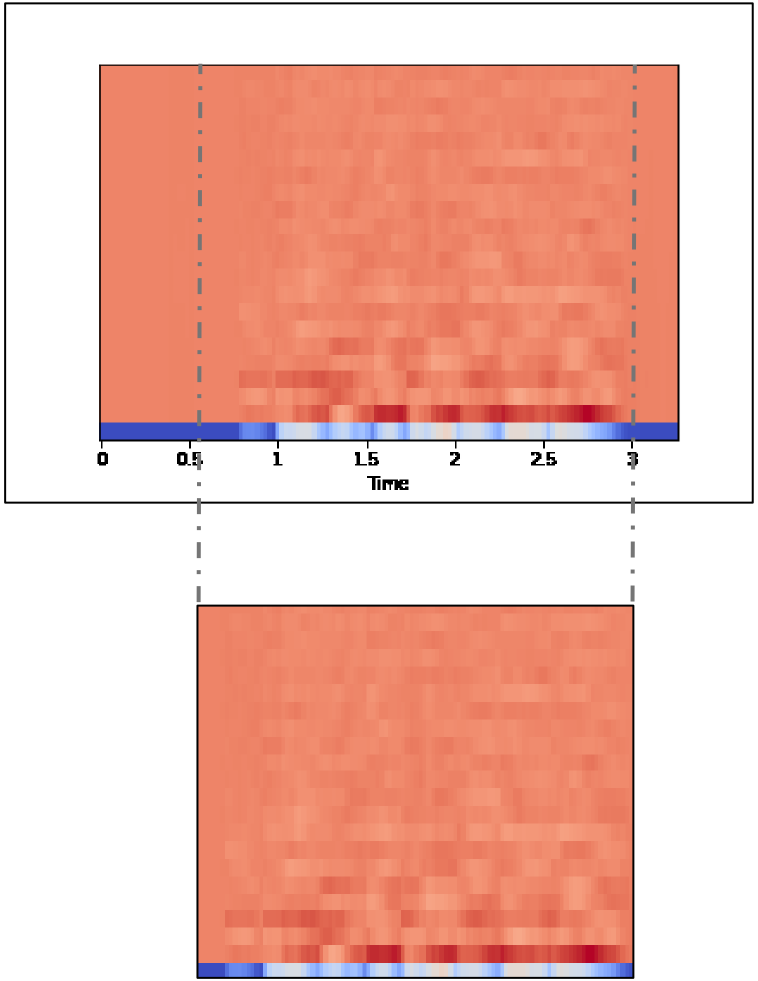
Note

We directly use

`librosa.feature.mfcc( )`

to transform audio wave to mfcc matrix.

N_mfcc = 22



Data Preparation

We cut down the spectrogram size from (432 x 288 x 3) to (240 x 210 x3).

Only reserve **the most dense information area** as CNN input.

Split the train set and test set with an ratio of 8 : 2.

CNN network: Use more conv layers to reduce params count

Layer (type)	Output Shape	Param #
conv2d_6 (Conv2D)	(None, 210, 240, 32)	128
conv2d_7 (Conv2D)	(None, 210, 240, 16)	4624
dropout_5 (Dropout)	(None, 210, 240, 16)	0
max_pooling2d_2 (MaxPooling2D)	(None, 42, 48, 16)	0
conv2d_8 (Conv2D)	(None, 42, 48, 16)	2320
dropout_6 (Dropout)	(None, 42, 48, 16)	0
conv2d_9 (Conv2D)	(None, 42, 48, 16)	2320
dropout_7 (Dropout)	(None, 42, 48, 16)	0
conv2d_10 (Conv2D)	(None, 42, 48, 16)	2320
dropout_8 (Dropout)	(None, 42, 48, 16)	0
max_pooling2d_3 (MaxPooling2D)	(None, 14, 16, 16)	0
conv2d_11 (Conv2D)	(None, 14, 16, 32)	4640
dropout_9 (Dropout)	(None, 14, 16, 32)	0
flatten_1 (Flatten)	(None, 7168)	0
dense_1 (Dense)	(None, 8)	57352

Total params: 73,704
Trainable params: 73,704

```
loss='categorical_crossentropy'
optimizer = Adam(learning_rate = 0.001),
metrics=['accuracy'])
batch_size = 16, epochs = 50
```

CNN network: A model with accuracy of roughly 60 percent.

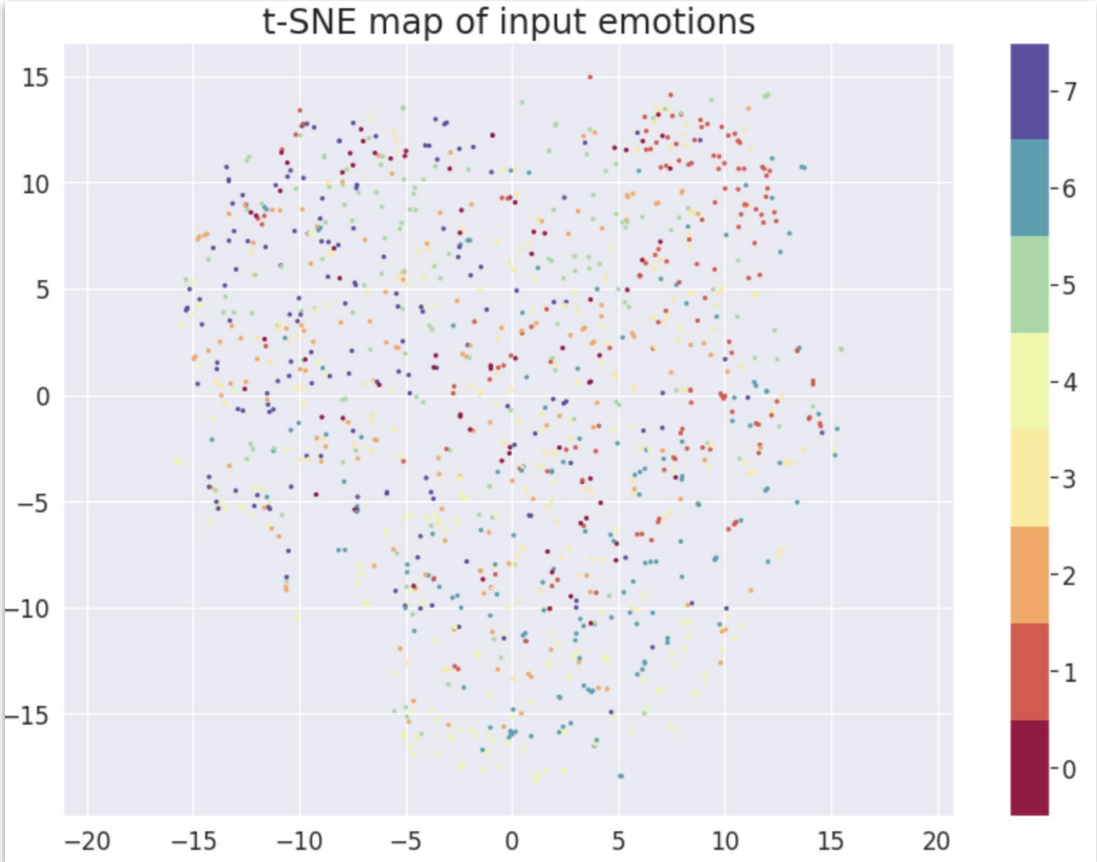
3 independent runs of the classifier, obtaining **a test accuracy of more than 60 percent**.
Stabilize below 70 percent as training budgets increase.
Unstable predicting correctness for now.

```
Epoch 50/50
72/72 [==============================] - 2s 34ms/step - loss: 0.3101 - accuracy: 0.9236 - val_loss: 1.3528 - val_accuracy: 0.6111
Test loss: 1.3528270721435547
Test accuracy: 0.6111111044883728
```

```
Epoch 50/50
72/72 [==============================] - 2s 32ms/step - loss: 0.2272 - accuracy: 0.9444 - val_loss: 1.2372 - val_accuracy: 0.6389
Test loss: 1.2371793985366821
Test accuracy: 0.6388888955116272
```

```
Epoch 50/50
72/72 [==============================] - 2s 34ms/step - loss: 0.2026 - accuracy: 0.9497 - val_loss: 1.5732 - val_accuracy: 0.6250
Test loss: 1.5732299089431763
Test accuracy: 0.625
```

Visualizing the boundary: Pattern within chaos



Surprise

disgust

fear

angry

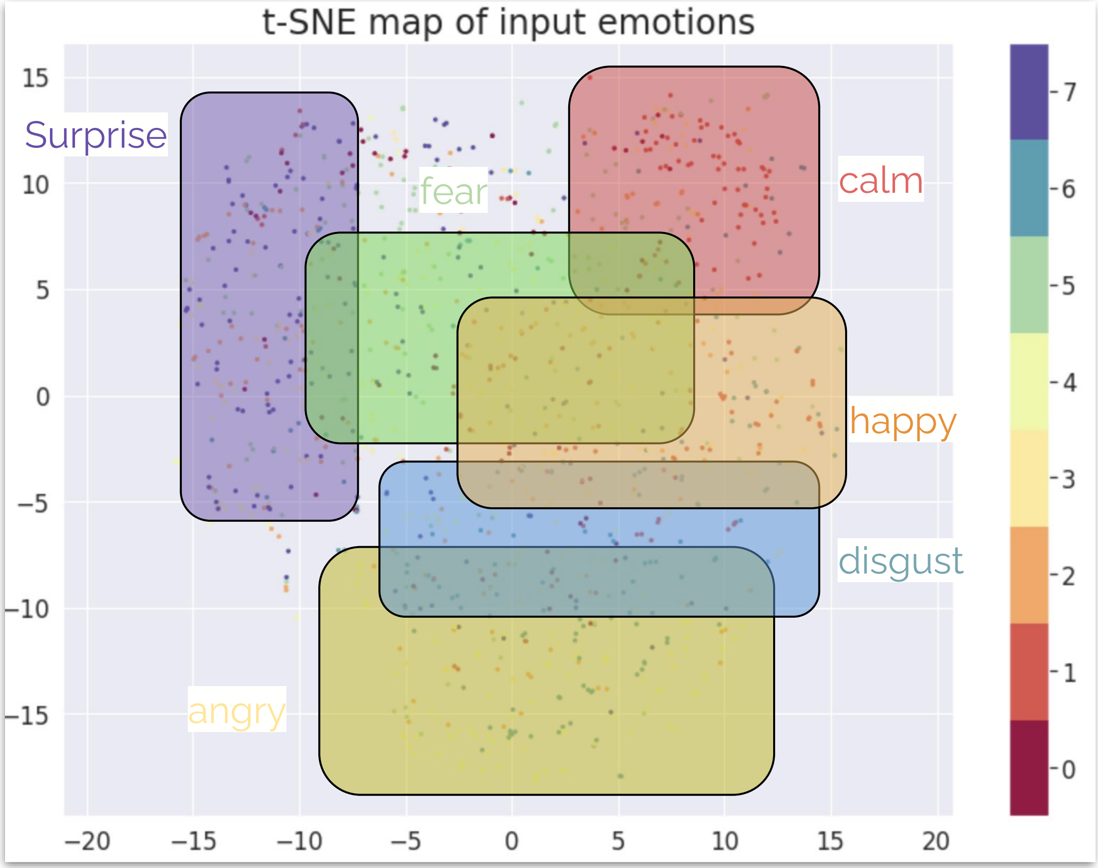
sad

happy

calm

neutral

Visualizing the boundary: Pattern within chaos



Confusion matrix: How's prediction going?

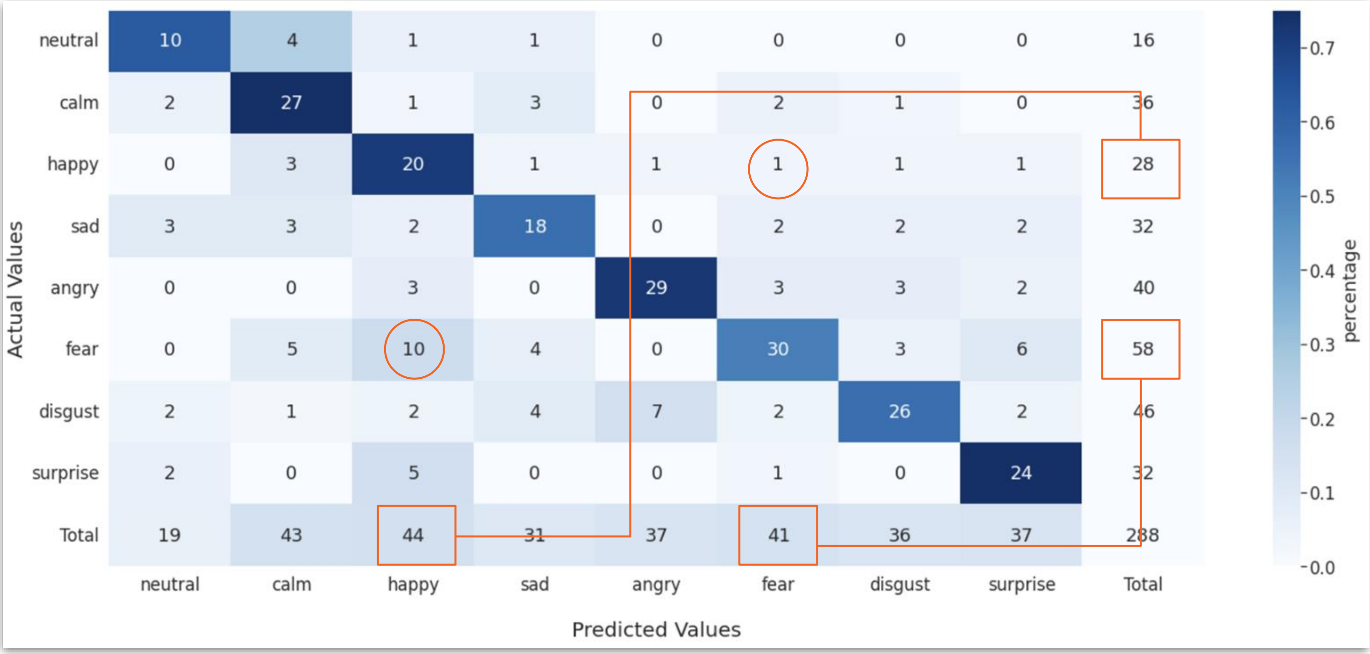
Surprise and **calm** are most likely to be **correctly identified**.
Fear and **disgust** are most likely to be **misinterpreted**.

Actual Values \ Predicted Values	neutral	calm	happy	sad	angry	fear	disgust	surprise	Total
neutral	10	4	1	1	0	0	0	0	16
calm	2	27	1	3	0	2	1	0	36
happy	0	3	20	1	1	1	1	1	28
sad	3	3	2	18	0	2	2	2	32
angry	0	0	3	0	29	3	3	2	40
fear	0	5	10	4	0	30	3	6	58
disgust	2	1	2	4	7	2	26	2	46
surprise	2	0	5	0	0	1	0	24	32
Total	19	43	44	31	37	41	36	37	288

percentage: 0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7

Confusion matrix: How's prediction going?

Fear may sound like happy, but happy sounds unlikely to be fear.
The classifier have **a tendency for being happy and not being fearful.**



Conclusions

- Extract features from audio speech to MFCC spectrogram.
- A CNN model that makes prediction on emotions.

Future Works

- Fine-Tune the CNN model.
- Create our own dataset to make predictions.
- Test the emotion classification performance of the model on other languages.