FACE EMOTIONS RECOGNITION

Team:
Alessandro Foglia 0000980147
Federico Pomponii 0000980381

Index

Index **1**

Introduction **2**

Dataset **3**

Network **4**

Results 5

Test 7

Final network considerations 9

Application **10**

Link to GitHub repository

Introduction

The project target is to create a ML Model that can classify a facial emotion on a subject. The client of the application is a mobile phone.
We used the FER2013 dataset (https://www.kaggle.com/msambare/fer2013) to train and test the model that is exported in a format compatible with TensorflowLite on Mobile phones.

In real-time the application recognises the emotion and prints the result in a label on the screen. There are 7 target results : *Angry - Disgust - Fear - Happy - Sad - Surprise - Neutral*.

The accuracy of the model is not really high, that depends on the difficulty of the classification, faces have different shapes and points of interest, it's not very easy to predict an emotion.

Dataset

The dataset has *35887* samples and 3 labels (emotion - pixels - usage)

Dataset is splitted in order to be used for training and test, based on the value of the *usage* column. Then each image is normalized before being processed.

(35887, 3)

	emotion	pixels	Usage
0	0	70 80 82 72 58 58 60 63 54 58 60 48 89 115 121...	Training
1	0	151 150 147 155 148 133 111 140 170 174 182 15...	Training
2	2	231 212 156 164 174 138 161 173 182 200 106 38...	Training
3	4	24 32 36 30 32 23 19 20 30 41 21 22 32 34 21 1...	Training
4	6	4 0 0 0 0 0 0 0 0 0 0 0 3 15 23 28 48 50 58 84...	Training

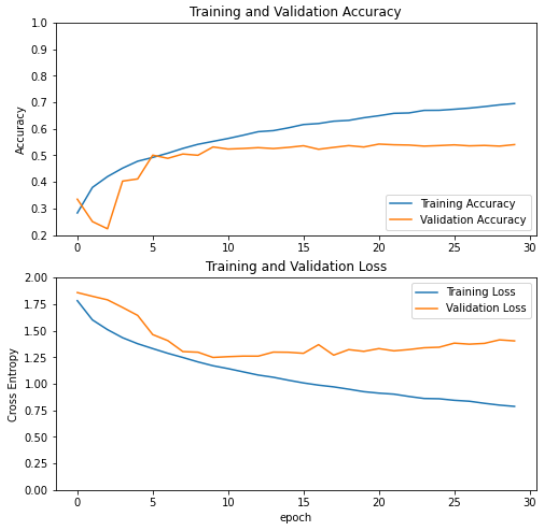
Network

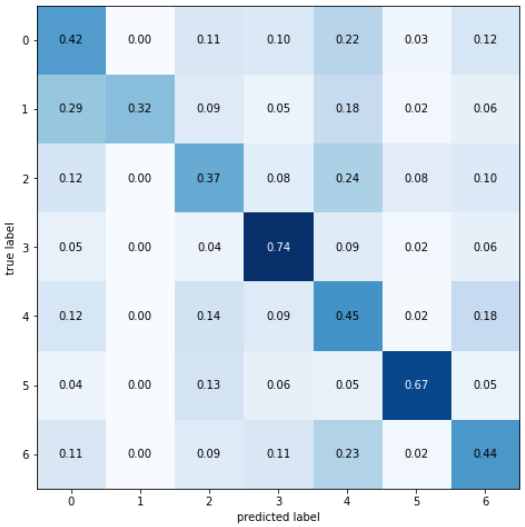
The model is a neural network based on a sequential model.
The network has two layers of *Convolution* with relu activation, mixed to two layers of *Pooling*. At the end there is a *Dropout* level followed by a *Batch Normalization.*
The input shape of the model is *(48,48,1)*

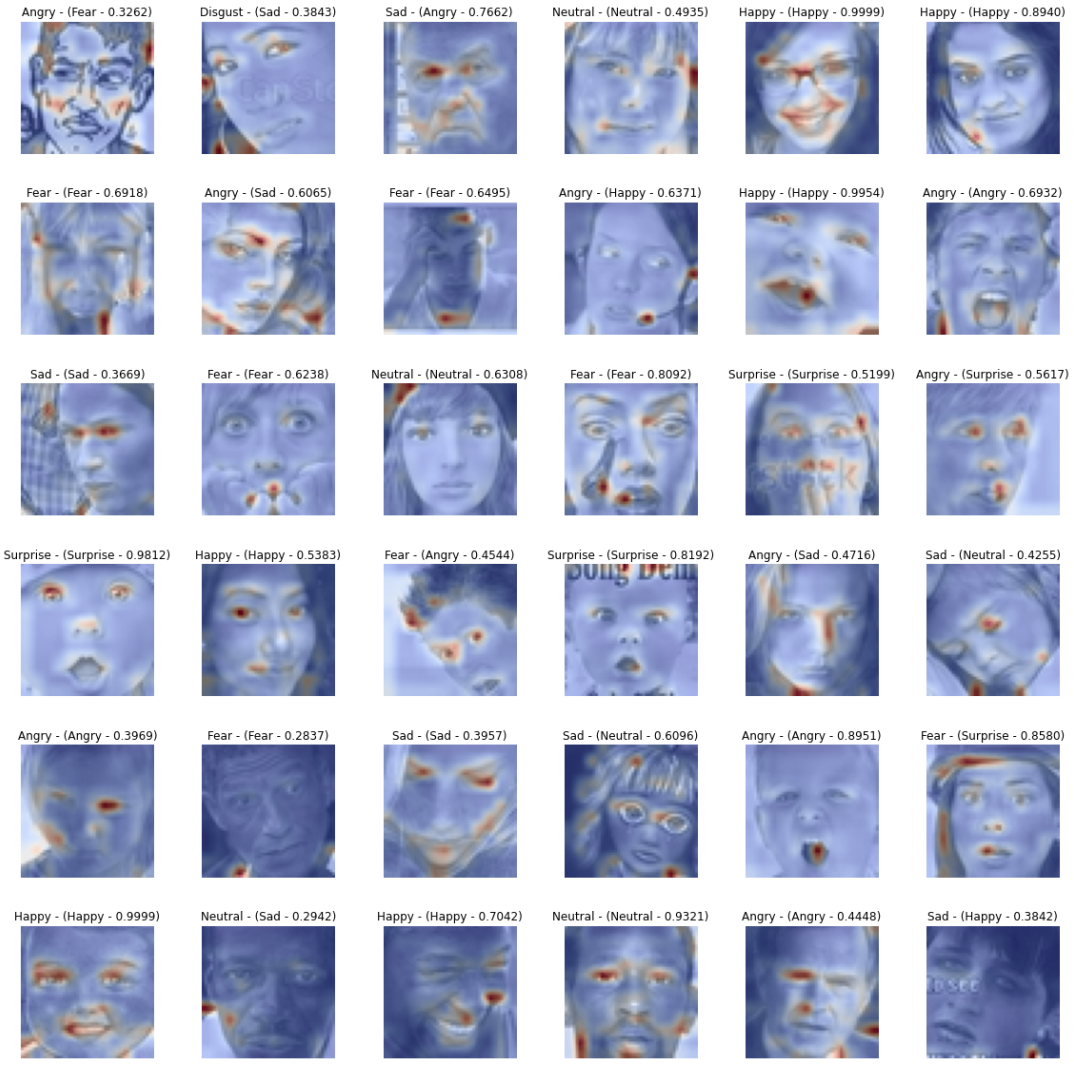
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 48, 48, 128)       1280

 max_pooling2d (MaxPooling2D  (None, 24, 24, 128)      0
 )

 conv2d_1 (Conv2D)           (None, 24, 24, 128)       147584

 max_pooling2d_1 (MaxPooling  (None, 12, 12, 128)      0
 2D)

 dropout (Dropout)           (None, 12, 12, 128)       0

 batch_normalization (BatchN  (None, 12, 12, 128)      512
 ormalization)

 flatten (Flatten)           (None, 18432)             0

 dense (Dense)               (None, 64)                1179712

 dropout_1 (Dropout)         (None, 64)                0

 dense_1 (Dense)             (None, 7)                 455

=================================================================
Total params: 1,329,543
Trainable params: 1,329,287
Non-trainable params: 256
_________________________________________________________________
```

Results

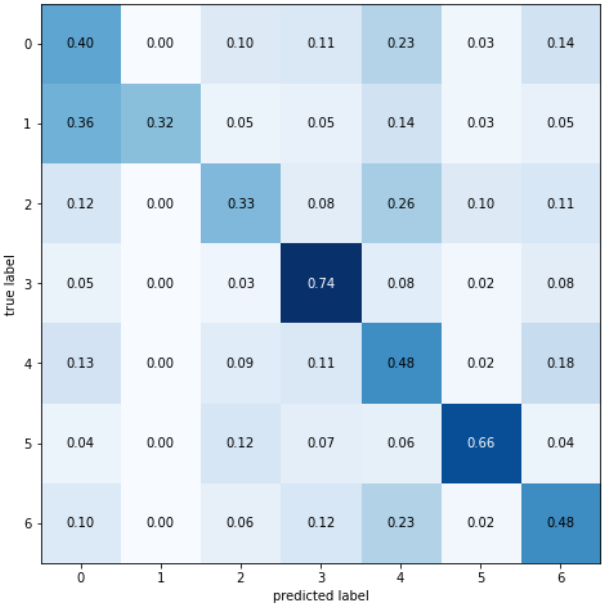
To train the model, a first fit was performed with the following parameters: *pool_size(4, 4), epochs=30* e *batch_size=256* obtaining the following results: loss: 0.7866 - accuracy: 0.6953 - val_loss: 1.4033 - val_accuracy: 0.5406 - confusion matrix: 0.5289774310392867

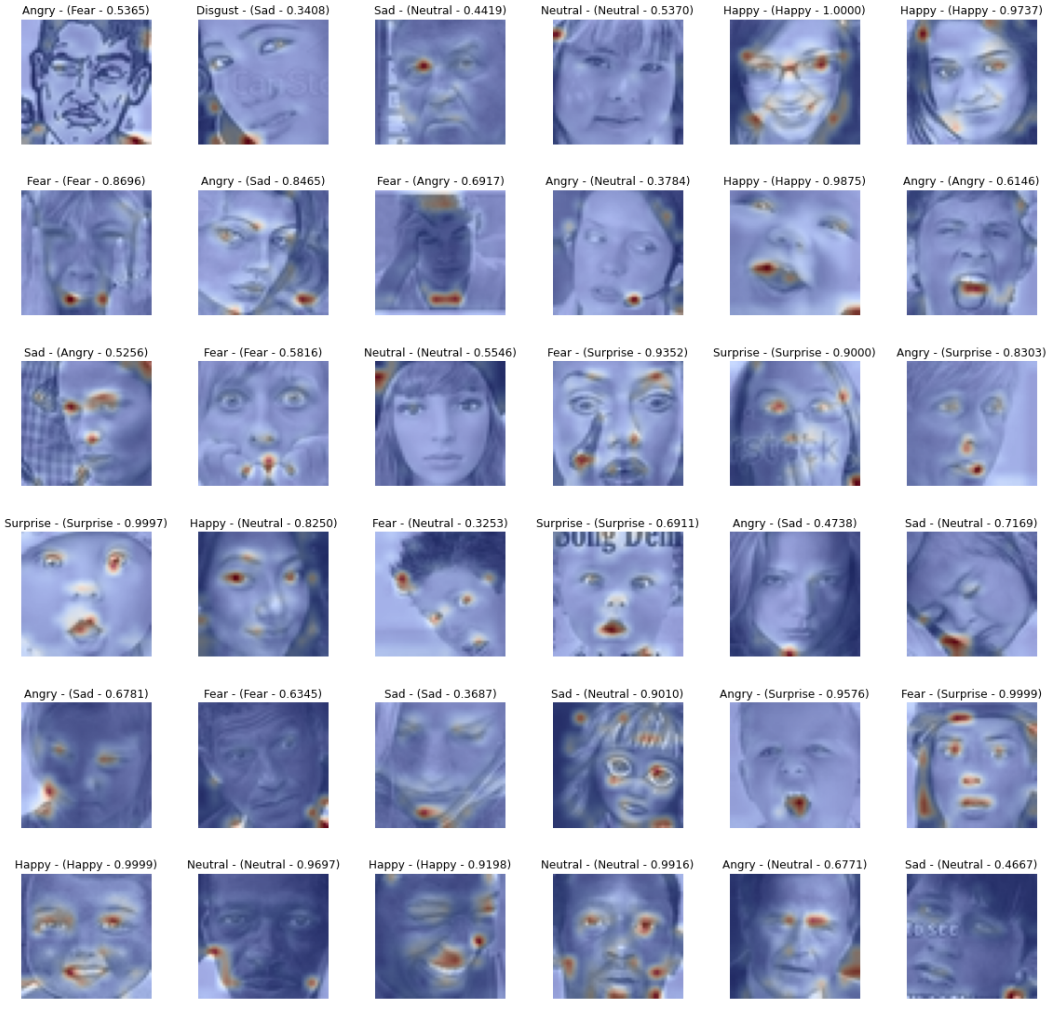






A second training test was carried out by changing only the number of epochs from 30 to 60 and obtaining the following results: loss: 0.6099 - accuracy: 0.7647 - val_loss: 1.5900 - val_accuracy: 0.5362 - confusion matrix: 0.529674003900808

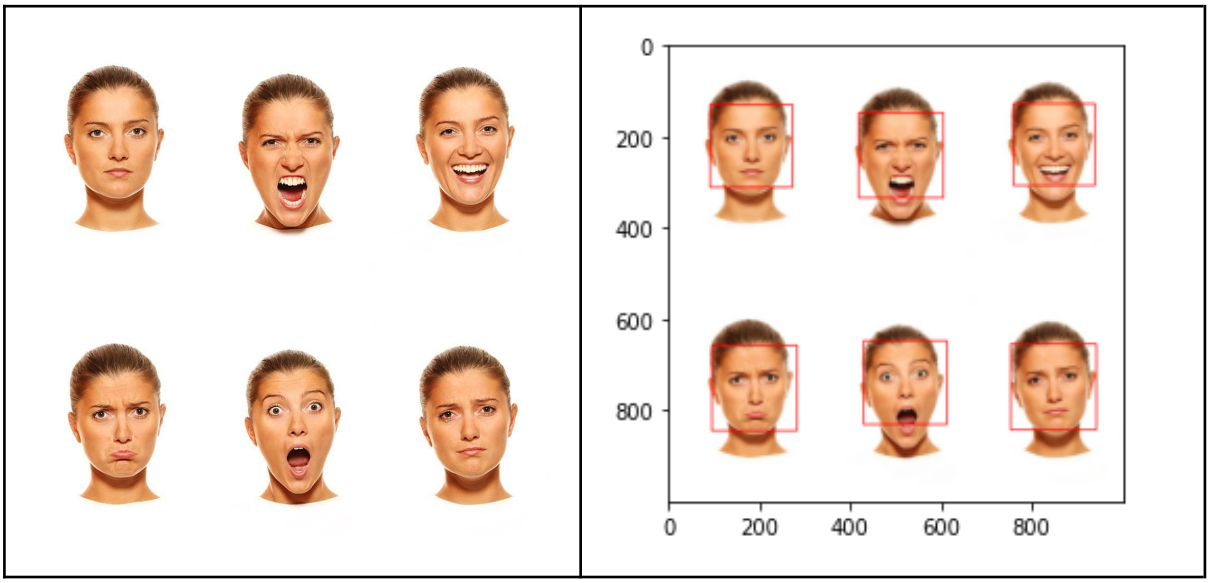


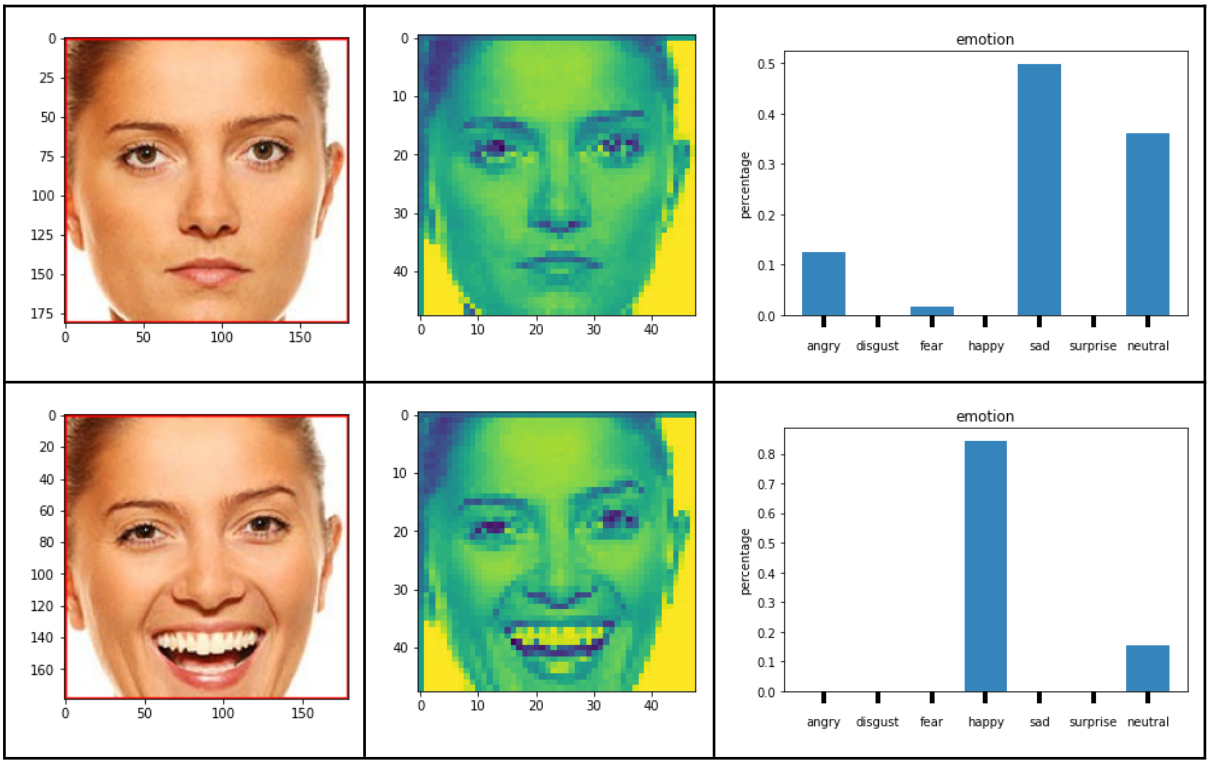


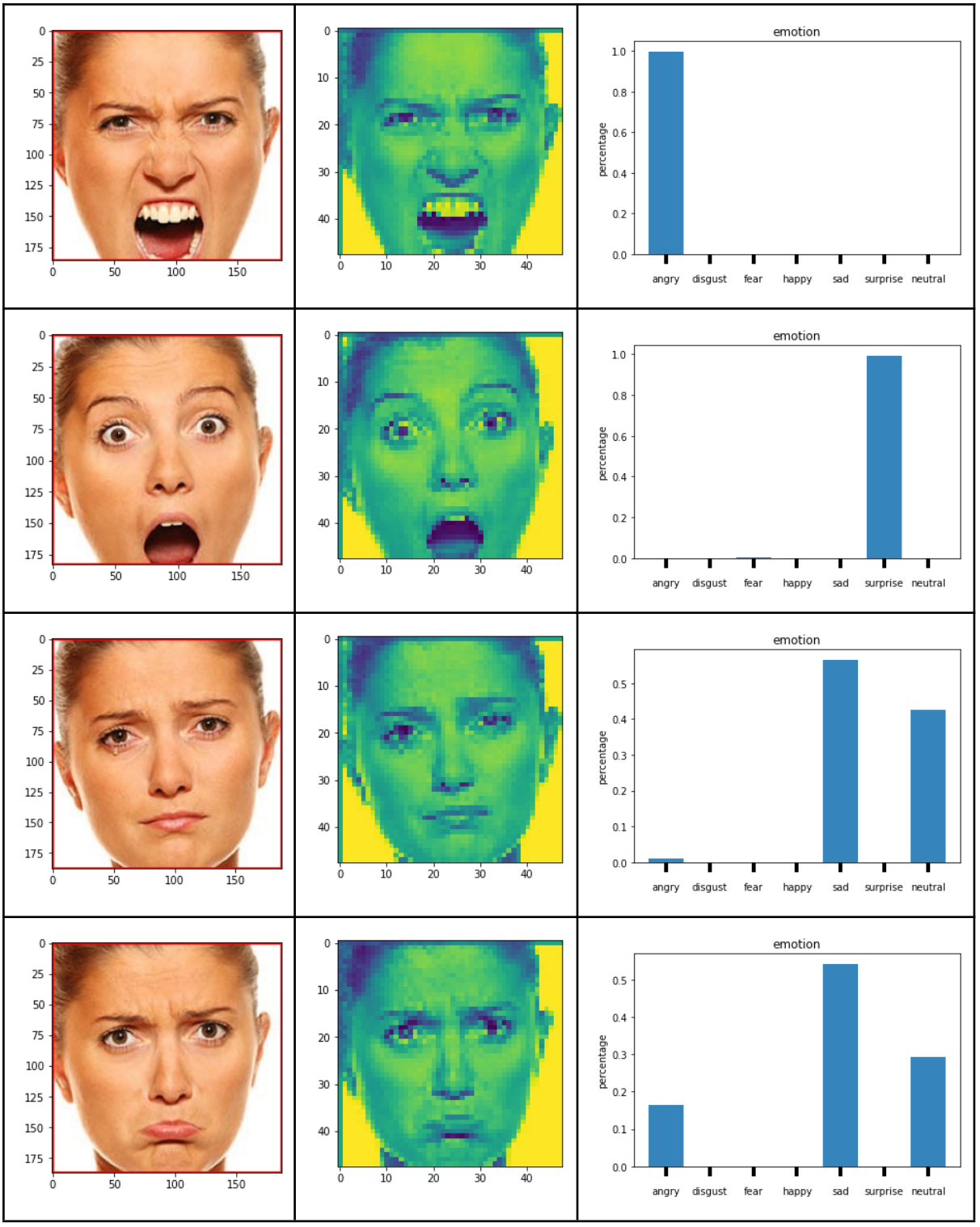
Test

The generated model was then tested with static images captured by the webcam or taken from the internet.

Through the *haarcascade frontal face* library it is possible to identify and cut out the face in the image. In this way, only the portion of the image it needs is supplied to the network. If there are more faces in the image, they are all analyzed separately.






emotion
percentage
angry
disgust
fear
happy
sad
surprise
neutral

Final network considerations

The *confusion matrix* and the class *activation map* were used to evaluate the model. The first relates the predicted label to the correct one, the ideal situation would be to have the diagonal with all 1s and the other cells at 0. As regards the class *activation map*, it indicates the regions used by CNN to identify the label. In the images above are those colored in red.

From the data obtained it is clear that by increasing the number of epochs or even by performing a second fit of the model there are no significant advantages. However from the final tests conducted, the network trained with 60 epochs, turns out to have a slightly higher accuracy in some cases.

A constraint was given by the final dimensions of the model as it must then be imported into the mobile application and consequently maintain small dimensions.

Application

The mobile application is developed using *React Native* that works on multiple platforms and not on a single environment.

There is just one use case: the user opens the application and the frontal camera of the smartphone. The prediction of the emotion is displayed on a label, on the bottom of the view.