PROJECT 2

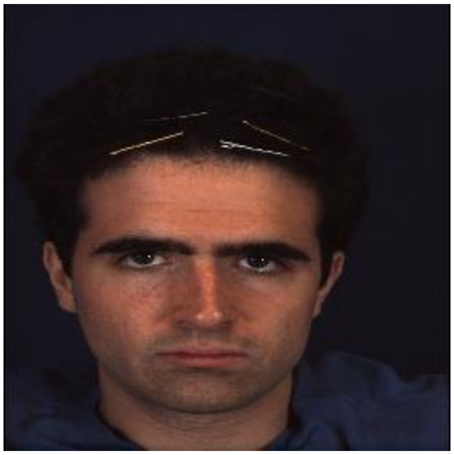
About the Project:

The main aim of the project is to distinguish between pain and no-pain among the set of images given in the database. The given database is sub-divided into three folders namely, training data, testing data, validation data, each dataset is further subdivided into pain and no-pain.

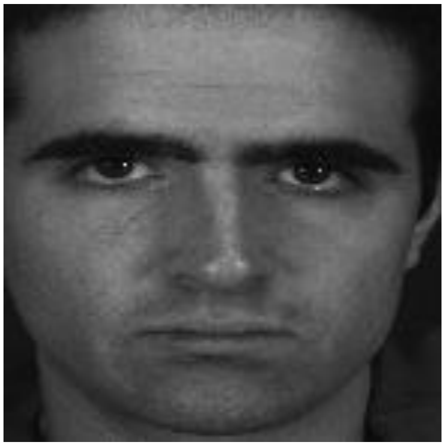
The objective of the project is to classify the images into pain and no-pain using deep-learning techniques such as CNN (convolution neural networks).

Initially the images were colored and were in dimensions 256x256, to get the facial part in the image we had cropped the image using OpenCV library, we also changed the color of the image into grayscale, at present all the images in the directories have been cropped.

Original image



cropped image(grayscale)



Our team has implemented a CNN (convolution neural network), the final script file consists of four layers, the explanation of the hidden layers is as follows, the input data is sent into the convolutional layer, the output of the convolutional layer is then passed into pooling layer, there are more than one convolution and pooling layer in this neural network, the output of the pooling is then flattened using flatten function. After flattening a fully connected layer is built. We have used sequential model for our CNN (convolution neural network).

The train data was used to train the data, while the validation data was used to see if the model is over fitting or not. Finally, the test data was used to see whether the trained model is making the right predictions or not.

Before finalizing the script file, our team has run it with different network configurations like increasing and decreasing the number of layers, changing the number of neurons in a particular layer, changing the target size of the image, testing the script on cropped dataset and original dataset.

The different set of accuracies by using different configurations is listed below:

Target size	**Type of data**	**Color of the dataset**	**Activation function**	**optimizer**	**No of layers**	**Dropout**	**Steps per epoch**	**Train accuracy**	**Final validation accuracy**	**Final test accuracy in percentage**
150x150	Cropped data	Grayscale	‘relu’	‘Adam’	4	0.5	100	87.85	66.3	36.16
150x150	Cropped data	Grayscale	‘relu’	‘Adam’	4	0.2	100	98.3	77.4	65.8
128x128	Cropped data	Grayscale	‘relu’	‘Adam’	4	0.5	100	96.6	58	65.03
180x180	Cropped data	Grayscale	‘relu’	‘Adam’	4	0.5	100	91.9	56.3	65.84
150x150	Original data	“RGB”	‘relu’	‘Adam’	4	0.5	100	98.3	53.1	63.33
180x180	Cropped data	grayscale	‘relu’	‘Adam’	3	0.5	100	95.25	75.5	65.804
180x180	Cropped data	Grayscale	‘relu’	‘Adam’	3	0.5	100	98.1	83.9	64.3

The Best Model:

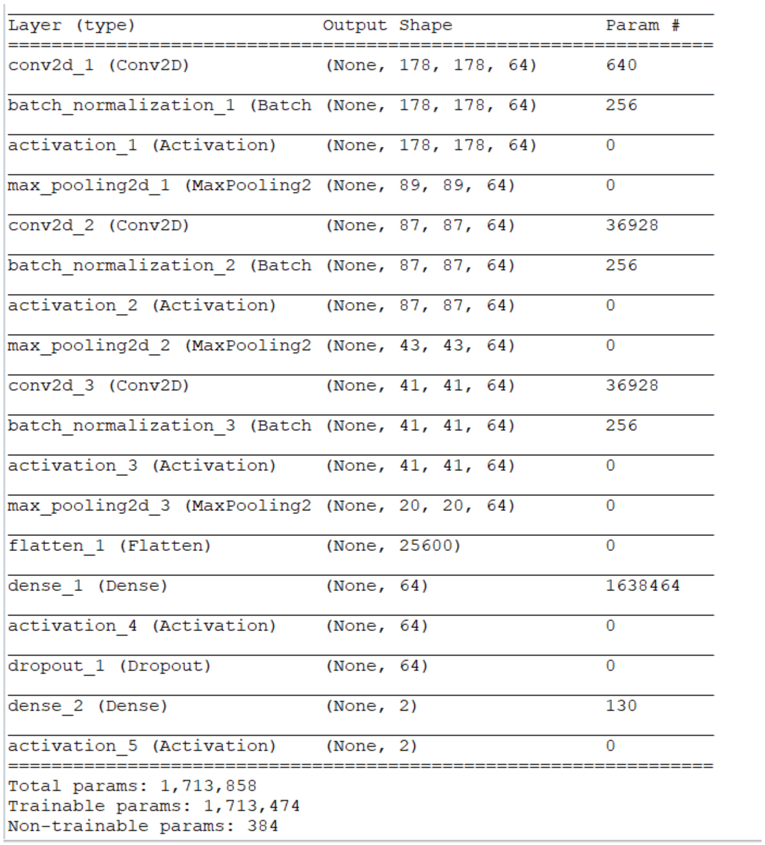
Initially the model was producing an accuracy 36.16%. This model had 4 convolutional layers (Number of layers in each layer n1=64,n2=32,n3=32,n4=32) and 2 dense layers with 64 neurons each.

The best model we had obtained has the best test accuracy of 65.804. For this, we had used a sequential model. We had applied 3 CNN layers where the number of neurons are 64 in each layer. MaxPooling was done with size (2,2). We had applied batch normalization for each layer and the activation function we used is relu.

After the convolutional layers we applied one fully connected dense layer with 64 neurons. And the activation function used for the final output layer is softmax.

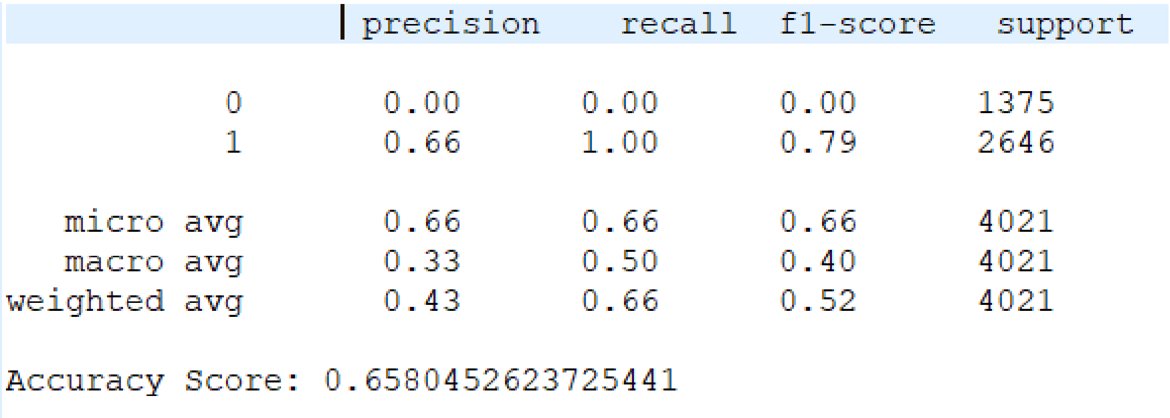
The final accuracies of the model are as follows, Train accuracy=91.9, validation accuracy=56.3, test accuracy=65.84.

Best Model Summary:



```
Layer (type)                 Output Shape              Param #   
=================================================================
conv2d_1 (Conv2D)            (None, 178, 178, 64)      640       
_________________________________________________________________
batch_normalization_1 (Batch (None, 178, 178, 64)      256       
_________________________________________________________________
activation_1 (Activation)    (None, 178, 178, 64)      0         
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 89, 89, 64)        0         
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 87, 87, 64)        36928     
_________________________________________________________________
batch_normalization_2 (Batch (None, 87, 87, 64)        256       
_________________________________________________________________
activation_2 (Activation)    (None, 87, 87, 64)        0         
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 43, 43, 64)        0         
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 41, 41, 64)        36928     
_________________________________________________________________
batch_normalization_3 (Batch (None, 41, 41, 64)        256       
_________________________________________________________________
activation_3 (Activation)    (None, 41, 41, 64)        0         
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 20, 20, 64)        0         
_________________________________________________________________
flatten_1 (Flatten)          (None, 25600)             0         
_________________________________________________________________
dense_1 (Dense)              (None, 64)                1638464   
_________________________________________________________________
activation_4 (Activation)    (None, 64)                0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0         
_________________________________________________________________
dense_2 (Dense)              (None, 2)                 130       
_________________________________________________________________
activation_5 (Activation)    (None, 2)                 0         
=================================================================
Total params: 1,713,858
Trainable params: 1,713,474
Non-trainable params: 384
```

precision ,recall, f1-score and accuracy:



```
              precision    recall  f1-score   support

           0       0.00      0.00      0.00      1375
           1       0.66      1.00      0.79      2646

   micro avg       0.66      0.66      0.66      4021
   macro avg       0.33      0.50      0.40      4021
weighted avg       0.43      0.66      0.52      4021

Accuracy Score: 0.6580452623725441
```

Confusion Matrix:

	Pain	No Pain
Pain	32	1343
No Pain	64	2582

Team members and their contributions:

1. Sharath Vanamala: file I/O, training and testing the model.
2. Dhiren Raj Korukonda: cropping the facial part of the image and changing it into grayscale, recording different configurations of the network.
3. Sowmya Munaganuri: data augmentation and crafting final report.