Report on

FACE EMOTION DETECTION AND MODIFICATION

*Submitted in partial fulfilment of the requirements for **Sem IV***

IMAGE PROCESSING AND DATA VISUALIZATION USING MATLAB

Bachelor of Technology
in
Computer Science & Engineering

Submitted by:

Swarnamalya A S	**PES2UG19CS418**
Rishi Patel	**PES2UG19CS329**
B Pravena	**PES2UG19CS076**
Varna Satyanarayana	**PES2UG19CS448**

Under the guidance of

<Prof. Swati Pratap Jagdale>
<Assistant Professor>
PES University, Bengaluru

January – May 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

TABLE OF CONTENTS

ABSTRACT

This project aims to detect and modify the emotion of a face present in a given image using matlab. This was solved by making use of AlexNet. AlexNet is a convolutional neural network (CNN) and therefore can be trained with any specific dataset in order to predict the result. This project made use of the FER 2013 dataset consisting of faces classified by emotions. This model predicts the emotion of any given face with 60% accuracy. The given face is further worked upon by detecting and identifying the mouth of the face using the vision cascade object detector. These coordinates are then used to remove the mouth of the given face using inpaint exemplar (so that during further changes, the skin-tone of the face remains the same). By segmenting just the lips of specific emotions such as sad, happy and neutral, we are able to replace it onto the given face to change the overall emotion of the face. The user is given the choice of which expression to replace onto the given face.

PROBLEM STATEMENT

To detect the emotion of the face detected in a given image by training and testing a dataset containing a large collection of images using deep learning models such as the Alexnet neural networks and further change the emotion of the face to either smiling, frowning or neutral by detecting the mouth of the detected face and replacing it with the chosen expression.

MODULE DESCRIPTION

Modify.m

Used to modify the images present in the training dataset to the size 227 × 227 × 3 from 48 × 48 because Alexnet is trained on 227 × 227 × 3.

Plaincode.m

This module is used to train the data using Alexnet to predict a certain emotion

Test_imgs.m

This is used to predict the given image using the pretrained data. Also calls EmotionChanger and the GUI which is used to select which expression is to be replaced onto the face. This is the driver module.

freezeWeights.m

It sets the learning rate of the first 10 layers to zero freezing prevents the weights of a neural network layer from being modified during the backward pass of training. You progressively 'lock-in' the weights for each layer to reduce the amount of computation in the backward pass and decrease training time.

findLayersToReplace.m

finds the single classification layer and the preceding learnable (fully connected or convolutional) layer of the layer graph lgraph. It then traverse the layer graph in reverse starting from the classification layer.

Test_imgs.m

This is used to predict the given image using the pretrained data. Also calls EmotionChanger and the GUI which is used to select which expression is to be replaced onto the face. This is the driver module.

createLgraphUsingConnections.m

creates a layer graph with the layers in the layer array |layers| connected by the connections in |connections|. It reconnects all the layers in the original order. The new layer graph contains the same layers, but with the learning rates of the earlier layers set to zero.

segmentImage_sad.m

This returns the segmented lips of the frown from the selected picture.

segmentImage_smile.m

This returns the segmented lips of the smile from the selected picture.

segmentImage.m

This returns the segmented lips of the neutral expression from the selected picture.

sad_lip_seg.m

Using the returned values from EmotionChanger, replaces a frown (returned from segmentImage_sad) on the face with no mouth, therefore changing its expression.

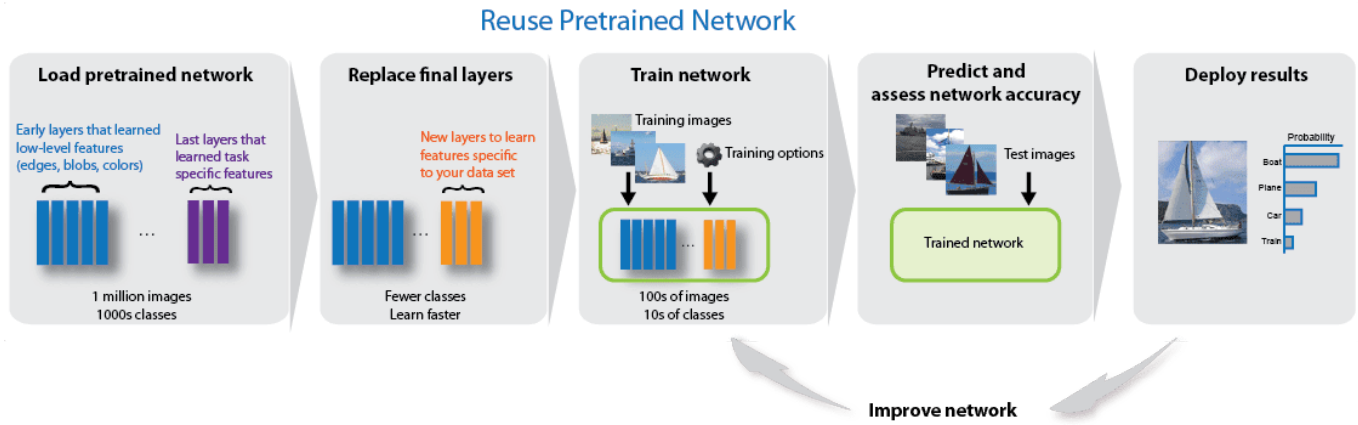
smile_lip_seg.m

Using the returned values from EmotionChanger, replaces a smile (returned from segmentImage_smile) on the face with no mouth, therefore changing its expression.

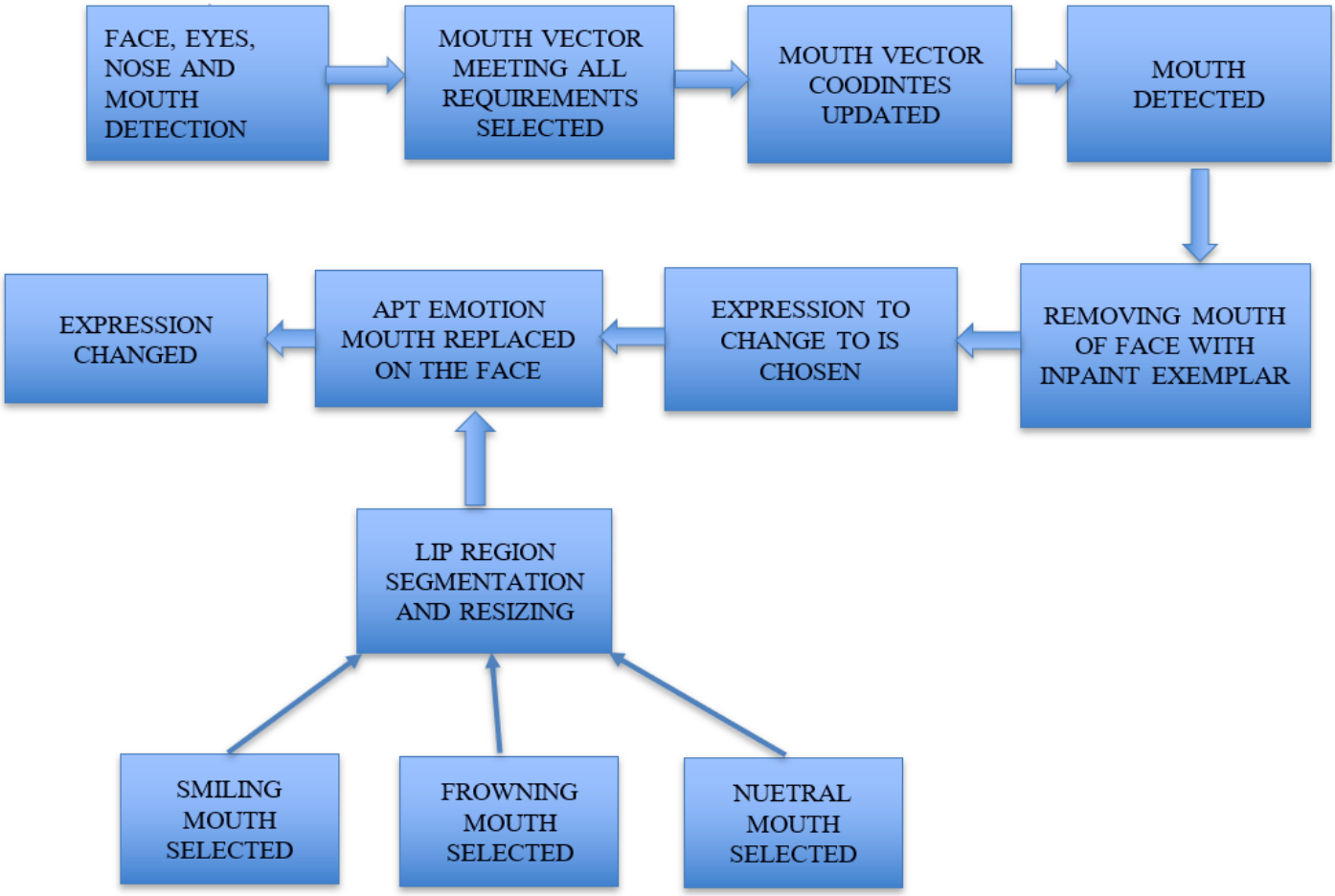
neutral_lip_seg.m

Using the returned values from EmotionChanger, replaces a neutral mouth (returned from segmentImage) on the face with no mouth, therefore changing its expression.

HIGH LEVEL DESIGN/ARCHITECTURE



A dataset consisting of a huge number of images is divided into two parts: 70% for training and 30% for testing. The pre-trained Googlenet network is loaded with early layers that learn low-level features and the last few layers learn task-specific features. We then replace the final layers specific to our dataset. The model is then trained using training options and the accuracy of the training is plotted into a graph for better visualization. Finally, the results are deployed and the trained model is now tested with new images to predict the facial emotion successfully and efficiently. The expression of the image can be changed now in the following manner:



Mouth Detection: On detecting the face in the given image, we will detect the mouth using the **CascadeObjectDetector** that makes use of the Viola Davis algorithm. We will store the coordinates of the mouth.

Isolating the mouth of an expression: Using the Image segmenter App and other apps from the Image processing toolbox in Matlab to manually cut out the mouth from a face depicting a particular expression. This is done to remove the skin and other extra features which is not required while replacing the mouth for a different expression. This mouth will, later on, be replaced in the user-entered picture.

Changing of expression: With the obtained coordinates of the mouth, using inpaint exemplar, we will remove the expression of the detected face by removing the mouth. On resizing the isolated mouth of the chosen expression, we will place it on the coordinates of the picture's original mouth. This changes the overall expression of the face.

IMPLEMENTATION

In order to predict the emotion of a given face, we start by finding a suitable dataset for our program to train using alexnet. FER 2013 dataset from Kaggle was used for this purpose. We then divide the dataset in 7:3 ratio for training and testing the data. The original dataset has a size of 48 × 48. We convert the size of each images in the training data to 227 × 227 × 3 using modify.m, this is because alexnet is trained on 227 × 227 × 3. After that we train the Finaldata using plaincode.m. Training the data takes around 400 minutes based on the learning rate and the gpu. Alexnet finishes training data and has an accuracy of 60%. We save this in AlexNet_64p03.mat so that we can use this pretrained data.

Using AlexNet_64p03.mat we test an image using test_imgs.m, which is capable of detecting 7 different emotions which are: angry, disgust, fear, happy, neutral, sad and surprise. Inorder to successfully detect the emotion, the background of the image must be ignored. This is done by vision.CascadeObjectDetector. The faces are then resized to 227 × 227 × 3 again and are then tested using the pretrained network AlexNet_64p03.mat.

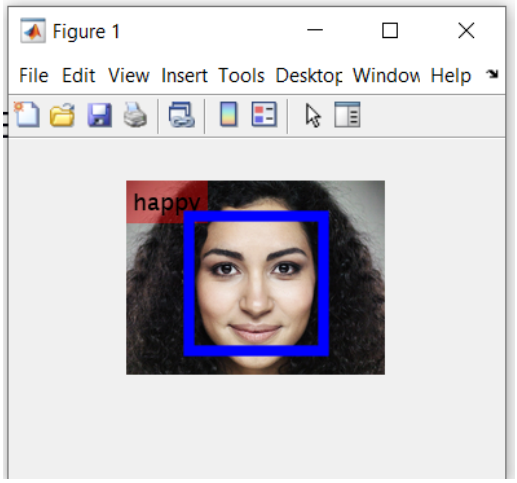
In order to modify the expression of the face, we consider that there is only one face in the image provided and the face is right side up. The first task is to detect the correct position of the mouth. This was done using vision.CascadeObjectDetector. First the face, eyes and mouth are detected. As the cascadeObjectDetector identifies the eyes, nostrils, mouth and any other curved line as the mouth, we need the coordinates of the face, eyes and nose. If eyes are detected, the detected mouth vector whose beginning y-coordinates are the farthest from the ending y-coordinates of the eyes but also lie within the coordinates of the face is considered. If the eyes are not detected, the vector with the largest starting y-coordinate that is within the coordinates of the detected face is considered. If the nose is detected, the beginning y-coordinate of the chosen mouth vector is updates to the ending y-coordinate of the detected nose and the height of the detected mouth vector is updates accordingly. Next, an ellipse is drawn on the image (face) provided with the coordinates of the calculated mouth vector on the mouth of the face. Using inpaintExemplar, a mask is created for that region and is replaced on the original image therefore, removing the mouth. Using this image and the coordinates of the calculated mouth, either a smile, a frown or a neutral mouth that has been obtained by mouth segmentation can be replaced pixel by pixel in the original mouth's position, therefore changing the overall expression of the detected face.

To obtain the segmented lips that will be used to change the given face's emotion, we selected images of a frown, a smile and a neutral expression. To ensure that the skin-tone of the selected face matches the mouth of the selected emotion, we segment just the lips from each of the selected images. Using the Matlab image segmenter tool, the lip from each of these pictures were isolated using the Local Graph cut and Graph cut options. The binary mask is then checked and converted to a function form. Therefore, on calling the function, the binary mask of the required segmented lips is returned. When the expression is to be changed, the segmented mouth is aptly resized in order to match the ratios of the face to be modified using the coordinates of the mouth detected in the face.

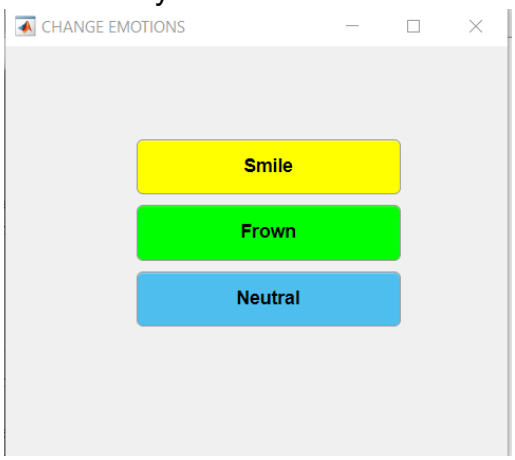
RESULT SNAPSHOTS

Example 1:

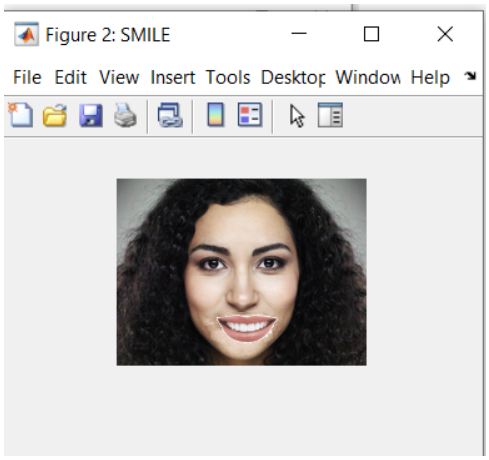
Detected emotion: happy



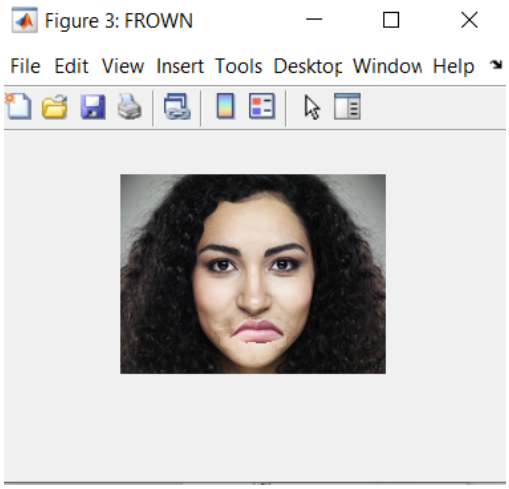
UI to modify emotion:



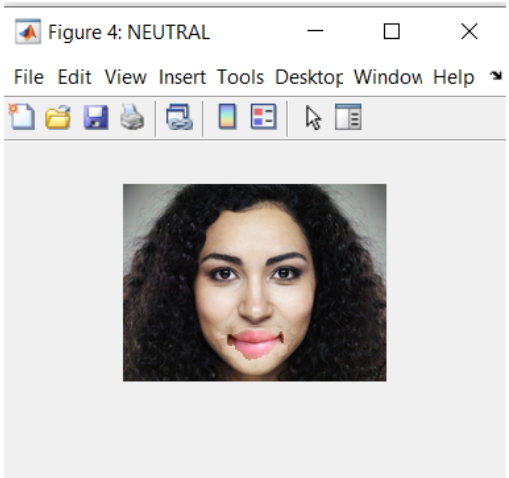
Smile selected:



Frown selected:

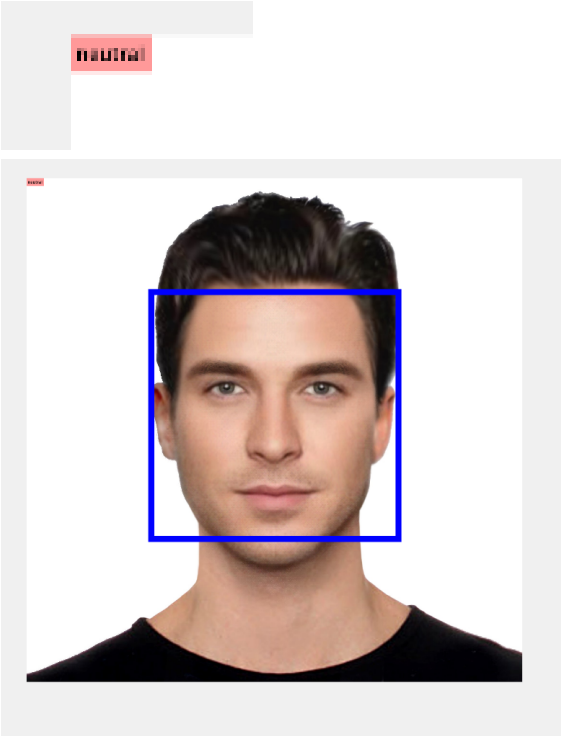


Neutral selected:

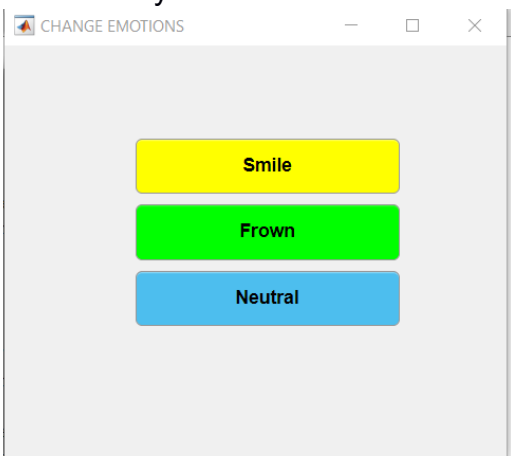


Example 2:

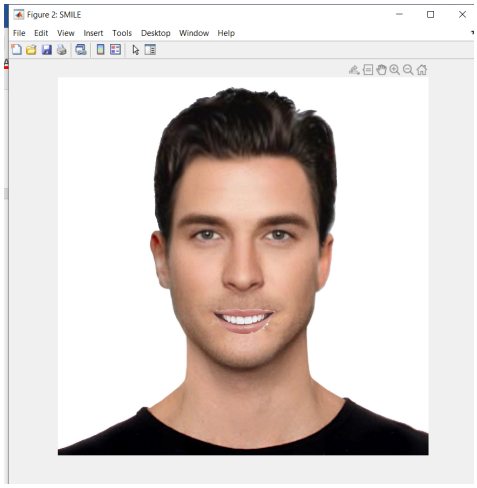
Emotion detected: neutral



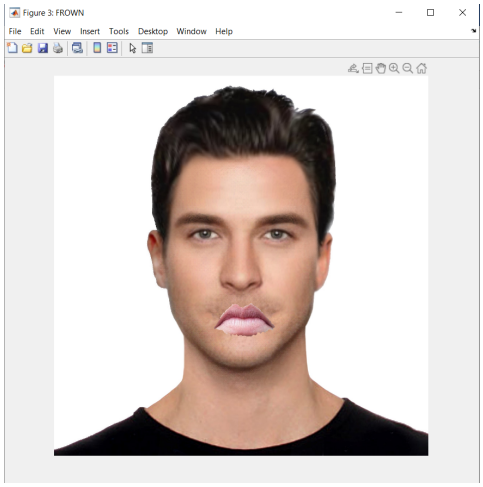
UI to modify emotion:



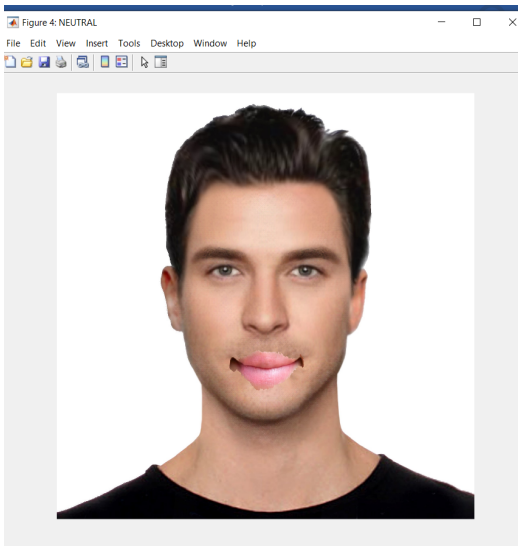
Smile selected:



Frown selected:

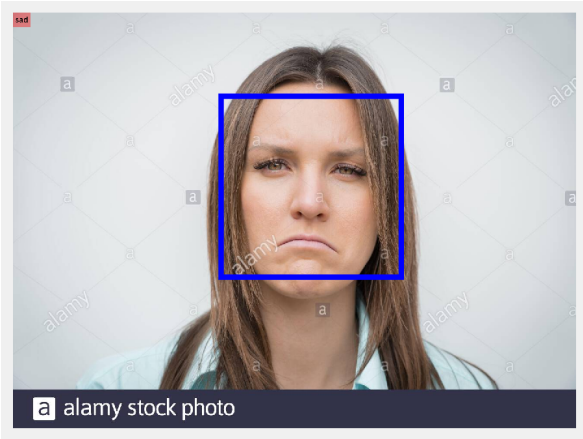


Neutral selected:

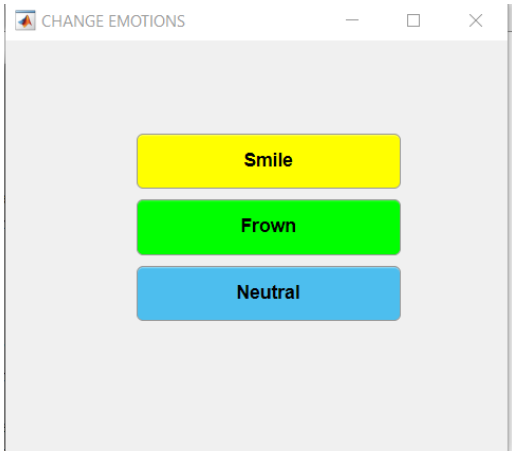


Example 3:

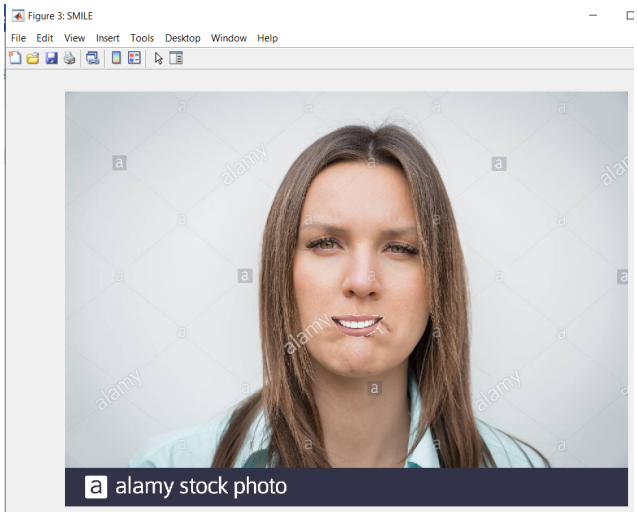
Emotion detected: sad



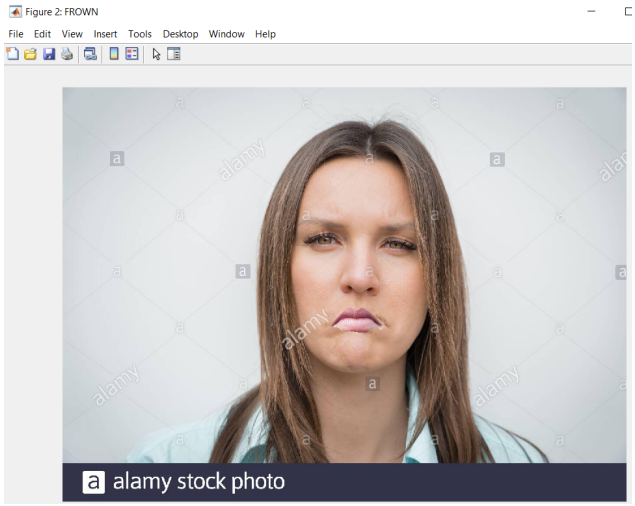
UI to modify emotion:



Smile selected:



Frown selected:



Neutral selected: