A Project
Report on

Emotion Recognition using Gabor filters

Submitted in partial fulfillment of
the requirements for the award of the degree of

Bachelor of Engineering
in
Computer Science and Engineering

Submitted by

USN	Names of Students
1DS16CS720	MAYUR S SHARMA
1DS16CS717	K VISHAL WARRIER

Under the guidance of
Dr. Preeti Satish
Associate Professor, CSE, DSCE

Dayananda Sagar College of Engineering
Department of Computer Science and Engineering
Bangalore, Karnataka, India - 560078

Dayananda Sagar College of Engineering
Department of Computer Science and Engineering
Bangalore, Karnataka, India - 560078

Certificate

This is to certify that the project entitled **Emotion Recognition using gabor Filters** is a bonifide work carried out by **Mayur S Sharma** [**1DS16CS720**] and **K Vishal Warrier** [**1DS16CS720**] in partial fulfillment of 8th semester, Bachelor of Engineering in Computer Science and Engineering under Visvesvaraya Technological University, Belgaum during the year 2019-20.

Dr. Preeti Satish	**Dr. Ramesh Babu**	**Dr. CPS Prakash**
(Internal Guide)	Vice Principle and HOD	Principal
CSE, DSCE	CSE, DSCE	DSCE
Signature:.....	Signature:.....	Signature:.....

Name of the Examiners	Signature with Date
1.....	.....
2.....	.....

Abstract

The emotions evolved in human face have a great influence on decisions and arguments about various subjects. In psychological theory, emotional states of a person can be classified into six main categories: surprise, fear, disgust, anger, happiness and sadness. Automatic extraction of these emotions from the face images can help in human computer interaction as well as many other applications. Machine learning algorithms and especially deep neural network can learn complex features and classify the extracted patterns. The proposed framework uses the Gabor filters for feature extraction and then a Convolutional Neural Network (CNN) for classification. The experimental results show that the proposed methodology increases both of the speed training process of CNN and the recognition accuracy.

The progression of the decades of scientific research has been conducted for developing methods for automated emotion recognition. Now, there is an extensive literature proposing and evaluating various methods, leveraging techniques from multiple fields, such as signal processing, computer vision, machine learning, and speech processing. Given an image of arbitrary size, the job is to identify an emotion of a human face appearing in the image. Face detection in complex environments is disputing since the faces may appear in different scales, different head poses, and orientations. External factors also play a vital role; for instance, the lighting conditions, facial expressions, and shadows are few other sources of variations that need to be taken into account. The approach is to yield a better classification performance implemented in real-world scenarios with fewer exemptions and more generalized accuracy.

A method for detecting facial regions by combining a Gabor filter and a convolutional neural network. The first stage uses the Gabor filter which extracts intrinsic facial features. The second stage of the method concerns the application of the convolutional neural network to these four images. The approach yields better classification performance in comparison to the results obtained by the convolutional neural network alone.

Acknowledgments

We have immense pleasure in successful completion of this project on Emotion Recognition using Gabor Filters

We would like to take this opportunity to express our gratitude to **Dr. C P S Prakash**, Principal of DSCE, for permitting us to utilize all the necessary facilities of the institution.

We are also very grateful to our respected Vice Principal, HOD of Computer Science and Engineering, DSCE, Bangalore, **Dr. Ramesh Babu D R**, for his support and encouragement.

We are immensely grateful to our respected and learned guide, **Dr. Preeti Satish**, Assistant professor, CSE, DSCE for their valuable help and guidance. We are extremely thankful to them for all the encouragement and guidance they have given us during every stage of the project.

We would like to thank our project coordinator **Dr. Vindhya P Malagi** Associate Professor, CSE, DSCE and Ms. Annapoorna B R, Assistant Professor, CSE, DSCE for their guidance and support.

We are also thankful to all the other faculty and staff members of our department for their kind co-operation and help.

Lastly, we would like to express our deep appreciation towards our classmates and our family for providing us with constant moral support and encouragement.

MAYUR S SHARMA
1DS16CS720
K VISHAL WARRIER
1DS16CS717

Contents

List of Figures

Chapter 1

Introduction

1.1 Emotion Recognition

Human emotions can be identified through emotion recognition. People vary widely in accuracy at recognizing the emotions of all people. The implementation of modern technology available these days help people in emotion identification which happens to be a rather nascent research area. Nowadays, most of the work has been going on in the area of automating the recognition of facial expressions through written, audio, video expressions from the physiology, and text which are measured through wearables technology.

Humans show plenty of cognitive changes in the course of their abilities to recognize emotion over the generations. A pivotal point to keep in mind when studying automatic emotion recognition is that there are many sources of truth about real emotion. For instance, Kyle may feel unhappy, but he has a huge, happy face that many people perceive that Kyle looks happy. If an automated method results in the same conclusion as a set of observers, it might be concluded as accurate, even though it is not actually show Alex 'truly' feels. Normally, getting to know the truth of as to which emotion is actually being expressed could take some amount of work. It can vary depending on the criteria that are selected and will involve sustaining some level of uncertainty.

1.2 Automatic emotion Detection

The progression of the decades of scientific research has been conducted for developing methods for automated emotion recognition. Now, there is an extensive literature proposing and evaluating various methods, leveraging techniques from multiple fields, such as signal processing, computer vision, machine learning, and speech processing. These various techniques and methodologies are used in order to interpret emotions like Hidden Markov Models, Gaussian Mixture Models and Bayesian networks.

The correctness of the emotion recognition is usually increased when it uses the analysis of human expressions from multimodal forms such as video, audio, physiology, or text. Emotion types are detected by the combination of information that are got from expressions, speech, gestures, and body movement. The upcoming technologies are known to contribute to the emergence of the emotive or emotional Internet.

The existing approaches in emotion recognition to classify certain emotion types can be frequently classified into two main categories: statistical methods and knowledge based techniques.

1.2.1 Knowledge based approach

Knowledge based techniques use domain knowledge and the semantic syntactic characteristics of the language in order to detect the various emotion types. Here, it is common to make use of knowledge based resources during classification process. The advantage of this approach is the accessibility brought about by the vast availability of such database resources. The limitation of this technique, on the other hand, is its incompetence in handling the nuances in complex linguistic rules.

This technique can be further classified into two categories: dictionary-based and corpus-based approaches. The dictionary-based approach finds emotion seed words and searches for their synonyms and antonyms in the dictionary and tries to expand a list of opinions or emotions. While the Corpus-based approach starts with a seed list of emotion words, and then expand the database by using other words with context-specific properties in a large corpus. While corpus-based approaches take into account the emotions' context, their performance can still vary in different domains since a word in one set of domains can have a different orientation in other domains.

1.2.2 Statistical methods approach

This is the primary approach used in this project to identify, classify, and label the emotions.

Statistical methods usually involve supervised machine learning algorithms in which a broad set of interpreted data is used in the algorithms for the computer to learn and predict the relevant emotion types. Machine learning algorithms provide more rational classification accuracy when compared to other strategies, but the challenges in achieving valid results in the exact classification process require an extensive training set.

Few of the most regularly used learning algorithms are: SVM (Support Vector Machines), Maximum Entropy, and Naive Bayes. Deep learning belongs to one of the unsupervised family of machine learning, so it is greatly used at emotion recognition. Prominent Deep learning algorithms include ANN such as CNN, Long Short term Memory, and Extreme Learning Machine. The popularity of these deep learning approaches may be mainly due to its success in German usage such as in NLP, computer vision, and speech recognition.

1.3 Datasets

Data is an integral part of emotion recognition and in most cases it is a hurdle to get expounded data which is required in the training of machine learning algorithms, classifying emotion types through multimodal sources through physiological signals, texts, or videos.

The following are a few datasets that are available to use :

1. MELD: is a multiparty dataset where each response is labeled with emotion and sentiment. MELD provides usefulness in atypical fashion of multimodal sentiment analysis and emotion recognition, emotion recognition in conversations and dialogue systems.

2. DREAMER: provides EEG and ECG recordings, and the special cases of emotion annotations liek valence, arousal, and dominance of people watching film clips.

3. SEMAINE: recordings linking a person and a virtual agent and contains variety of emotion annotations.

4. IEMOCAP: recordings of dyadic sessions between actors and contains variety of emotion annotations like happy, neutral and anger.

5. eNTERFACE: audiovisual recordings of subjects from seven nationalities.

6. DEAP: electroencephalography (EEG), electrocardiography (ECG), and face video recordings, as well as emotion in terms of arousal and dominance of people watching the clips.

7. Belfast database: a variety of emotions from TV recordings and programs.

8. HUMAINE: Yields natural clips containing emotion words in multiple modalities.

1.4 Ways to Detect Emotion

1.4.1 Emotion recognition in text

Text data is usually used as a research object for emotion recognition. Compared to other types of data, text storing in the form of data is lighter and comfortable to process to the best performance due to the repetition of words and characters. Emotions can be extracted from text in two essential forms: written texts and dialogues. Many scholars focus on using sentences with levels to extract words/phrases representing emotions.

1.4.2 Emotion recognition in audio

Vocal signals are used for the recognition to extract emotions from audio.

1.4.3 Emotion recognition in video

Video data is a combination of audio, image, and texts (subtitles). Therefore, emotion recognition yields in the video are generally better because of more features are put in the detection process. On the contrary, the processing time may be longer. Nowadays, solutions like ReelTimeAI can analyze emotions from a human face in real-time using neural networks and AI.

1.4.4 Emotion recognition in conversation

ERC (Emotion recognition in conversation) extracts opinions between participants from massive conversational data in social media platforms. ERC takes input data like text, video, audio, or a combination form to detect several emotions.

Chapter 2

Problem Statement

The existing systems use on the neural networks whose accuracy of emotion detection is less. The existing face databases are not secure for real-world applications such as human-computer interaction.

Because of the face's enormous characteristic features, the training and testing of each face require extensive data to accomplish a reasonable accuracy. The detection of facial emotion requires large datasets to train the machine, which takes ample data storage and more time.

These might come across as significant problems while implementing emotion and face detection during real work applications in Graphical Designing, Visual FX, Snapchat/Instagram Filters, Facial psychology, etc.

2.1 Existing Systems

1. Kwolek et al. performed facial recognition using the Gabor filter to extract features. In that paper, it was shown that using the Gabor filter augments the accuracy of CNN from 79

2. Wu et al. suggested that Gabor motion energy filters (GME) could be used to detect facial expressions. In this method, the GME filter was compared with the Gabor energy (GE) filter, and the results showed that the proposed method was 7

3. Li et al. presented a deep fusion convolutional neural network (DF-CNN) method using 2D+3D images. In this method, three-dimensional scan images of the faces are used. These images make up a total of 32 dimensions. This paper explains that prediction is made using two methods. 1- Classification by using the SVM from the 32dimensional features. 2- The typical prediction of the Softmax function using the

six-state probability vector. Experimental results show that DF-CNN achieved excellent results.

4. Tzirakis et al. suggested that both voice and image processing can detect human situations where the sound is separately conveyed to a neural network.

Chapter 3

Proposed System

The proposed framework uses the Gabor filters for feature extraction and then passed onto the deep convolutional neural network. The experimental results show that the proposed features increase the speed and accuracy of training the neural network.

Image Capturing
Proposed System consists of a web high definition camera, placed in the System to capture all the person. From these captured image frames, the person's faces are detected using the OpenCV face detection technique.

Gabor filter
In most cases, Gabor filters are used in texture analysis, feature extraction, and edge detection. When a Gabor filter is applied to an image, it gives the highest response at edges at points with texture changes.

Convolutional Neural Network
The CNN architecture is designed in such a way that it uses four primary functions available in the libraries, namely, MaxPooling, Flatten, Softmax, Renu Functions, to narrow down the feature extracted image to 7 emotions: Neutral, Happy, Sad, Surprised, Fear, Disgust and Anger.

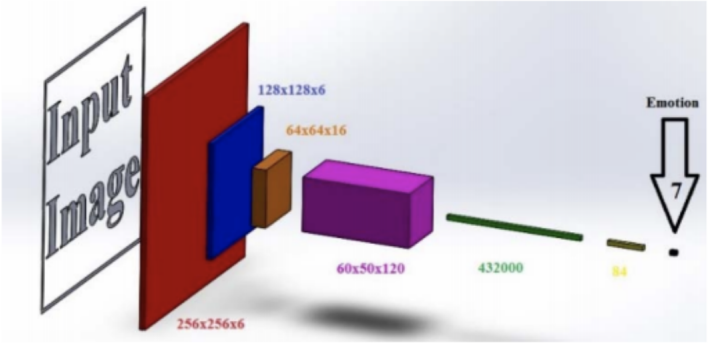

Figure 3.1: Architecture of CNN

3.1 System Requirements

Hardware Requirements
Processor : Intel Core i5 - 2.4 Ghz
Hard Disk : 1 TB
Monitor : 13" LED
Input Devices : Keyboard, Mouse
Ram : 8 GB
System Type : 64-bit Operating System

Software Requirements
Operating system : Windows 10
Coding Language : Python
Operating Environment : Tensor flow

Functionality Requirements
These following points describe the system's functional requirements, expressed in the natural language style:

1. Create a Desktop application using Tkinter.
2. User should train the emotions faces expression.
3. The system will capture the face using a webcam and extract the features.
4. The system will train data using the CNN algorithm.

5. The application should accurately recognize the face and detect emotions based on the trained data.

Non-Functionality Requirements These are non-functional requirements, that is, constraints within which the system should work.

1. Self-contained so that it can easily be portable from one computer to another.

2. Capacity, availability, and scalability.

   - The system shall achieve 100 percent availability at all times.
   - The system should be scalable to support additional clients.

3. Maintainability.

   - The system should be optimized for ease of maintenance as far as possible. Documentation of coding standards, class libraries, and abstraction, naming conventions can be followed to satisfy the user.

4. Randomness, load balancing, and verifiability.

   - It is optimized for supportability or ease of maintenance. It may be achieved through the use of documentation of coding standards, class libraries, abstraction, and naming conventions. Randomness should be present to check the nodes and should be load balanced.

Chapter 4

Literature Survey

1. The first proposed solution we go through is a paper where it focuses on the extraction of discriminative features from salient facial patches is vital in emotion recognition. The accuracy of the detection of facial landmarks improves the maximum localization of the salient patches on face images. Here, this paper proposes a framework for expression recognition by using the appearance features of selected facial patches. Facial patches are extracted, which are active during emotion elicitation, depending on the position of facial landmarks. These active patches are curated to obtain the salient patches, which comprise essential features for classification of each pair of expressions, thereby choosing different facial patches as salient for different pairs of expression classes. The one-against-one classification method is used using these features. Moreover, an automated learning-free facial landmark detection technique is proposed, where it results in a similar performance as that of other modern landmark detection methods, yet requires significantly less execution time. The proposed method is found to perform well consistently in different environments, hence, providing a solution for expression recognition in low-resolution images. Experiments on CK+ and JAFFE facial expression databases show the effectiveness of the proposed system.

   Advantages: Improves the localization of the salient patches on face images. Predicts expression recognition in low resolution images.
   Disadvantages: Requires huge data storage. Requires huge computational power.

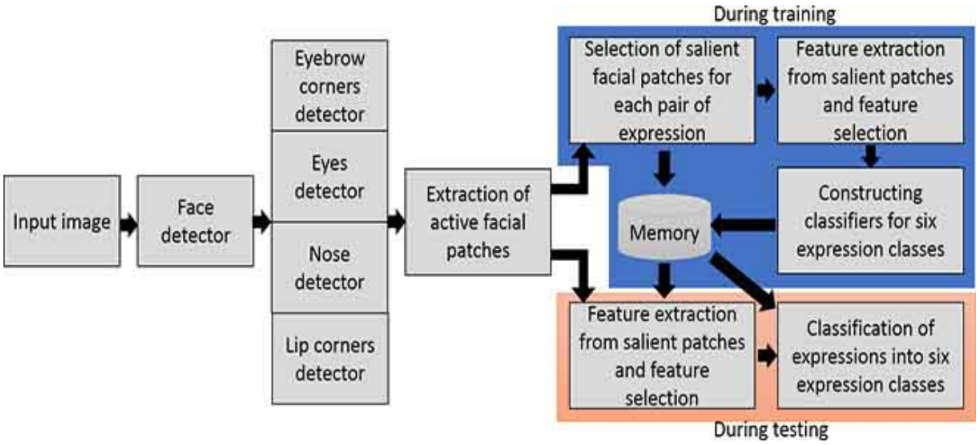

Figure 4.1: Overview of classification using salient patches

2. This paper presents a fuzzy logic based emotion recognition system. The system is comprised of an image processing stage, followed by an emotion recognition stage. In the image processing stage, the subject's facial features (eyes, mouth, etc.) are extracted. Next, the identifying points strategically placed on the face are extracted from each facial feature. In the emotion recognition stage, the identifying points are used to fuzzify and determine the strength of different facial actions. These strengths are then used to determine the subject's displayed emotion. The Japanese Female Facial Expression database was used to evaluate the system's performance resulting in an overall successful detection rate of 78.8 percent.
   Advantages: Improves the accuracy of emotion prediction . Improves its advantage of phycology.
   Disadvantages: Accuracy is very less. Requires huge fuzzy sets to predict emotions.

3. An emotion recognition method from a set of face images is proposed in this paper, which can recognize seven emotions of humans, i.e., six basic expressions in addition to neutral (anger, sad, surprised, disgust, fear, happy). The proposed method uses the GLCM approach for feature extraction and the nearest neighbor (NN) for classification. The fuzzy Euclidean distance is used with GLCM providing the texture characteristics of an input image through second-order statistical measurements. Because of the existence of vagueness and uncertainty in the discriminant features extracted from various emotional face images, the fuzzy

measure is involved in the NN classifier to recognize the emotions of faces with more accuracy. The experiments show excellent efficiency compared to some other feature extraction and facial emotion recognition methods.
Advantages: Good efficiency in facial emotion classification. Increase efficiency in feature extraction.
Disadvantages: Accuracy is very less. Requires huge fuzzy sets to predict emotions.

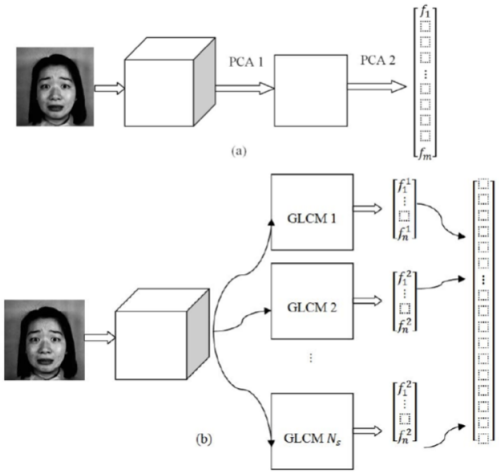

Figure 4.2: Two approaches for facial extraction (a) with feature reduction (b)without feature reduction

4. This paper proposes a method for detecting facial regions by combining a Gabor filter and a convolutional neural network. Gabor filter is used in the first stage, which extracts the most intrinsic facial features. As a result of this Gabor-two-filters transformation, we obtain four sub-images. The second stage of the method uses the application of the convolutional neural network to these four images. This approach yields better classification performance in comparison to the results obtained by the convolutional neural network alone.

   Advantages: Improve better classification performance of facial detection. Increase efficiency of extracts intrinsic facial features.

   Disadvantages: Not suitable for emotion detection. Can't consider the all features to for face detection.

5. Here they present a novel and efficient Deep Fusion Convolutional Neural Network (DF-CNN) for multi-modal 2D+3D Facial Expression Recognition (FER). DF-CNN uses a feature extraction subnet, a feature fusion subnet, and a softmax layer. In particular, each textured 3D face scan is represented as six types of 2D facial attribute maps, all of which are jointly fed into DF-CNN for feature learning and fusion learning, which results in a highly concentrated facial representation (32- dimensional). Prediction is calculated in two ways: 1) learning linear SVM classifiers using the 32-dimensional fused deep features; 2) directly performing softmax prediction using the 6-dimensional expression probability vectors. Very Different from existing 3D FER methods, DF-CNN combines feature learning and fusion learning into a single end-to-end training framework. To demonstrate the effectiveness of DF-CNN, conducted comprehensive experiments to compare the performance of DFCNN with handcrafted features, pre-trained deep features, fine-tuned deep features, and on three 3D face datasets. Usually, in most of the cases, DF-CNN consistently achieved the best results. This is the first work of introducing deep CNN to 3D FER and deep learning-based feature-level fusion for multi-modal 2D+3D FER.
   Advantages: Improves the localization of the salient patches on face images. Predicts expression recognition in low resolution images.
   Disadvantages: Requires huge data storage. Requires huge computational power.

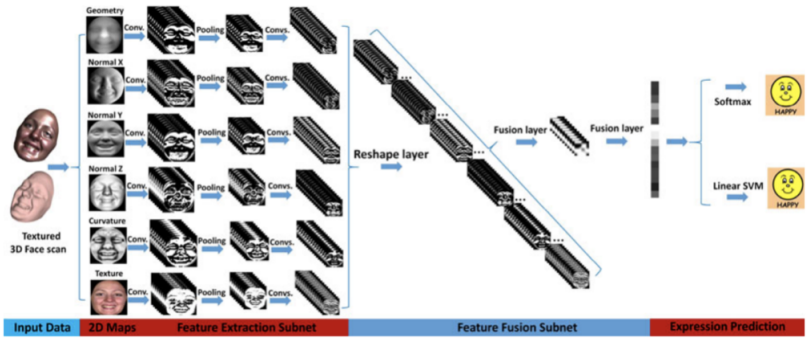

Figure 4.3: Pipeline of the proposed DF-CNN based multimodal 2D+3D FER approach

6. Due to the different modalities emotions could be expressed, automatic affect recognition might be a challenging task. Applications can be

found in various domains, including multimedia retrieval as well as human-computer interaction. Past few years, deep neural networks have had great success in determining emotional states. Due to this success, the paper proposes an emotion recognition system using auditory and visual modalities. To capture the emotional content for different styles of robust, speaking features have to be extracted. We use a CNN (convolutional neural network) to extract features from the speech, while for the visual modality, a deep residual network of fifty layers is employed.
Additionally, to the importance of feature extraction, a machine learning algorithm needs also to be insensitive to outliers while having the ability to model the context. Long short term memory networks are utilized in order to tackle this problem. The system is then trained in an end-to-end fashion where—making use of an advantage of the correlations of each of the streams— that we manage to significantly outperform, on the lines of concordance coefficient of correlation, traditional approaches supported visually handcrafted as well as auditory features for the prediction of spontaneous and natural emotions on the RECOLA database of the AVEC 2016 research challenge on emotion recognition.

7. An unsolved problem in computer vision is the Automated emotion recognition from facial images. Although Recent methods achieve a near-human accuracy under controlled scenarios, the popularity of emotions within the wild remains a challenging problem. Looking at the advances in Deep learning has supposed a breakthrough in many computer vision tasks, including facial expression analysis. Notably, the Deep Convolutional Neural Networks usage has reached the most straightforward ends up in the recent public challenges. This state of the art algorithms brings out the usage of ensembles of CNNs, which can outperform an individual CNN classifier. Two key considerations that are a significant influence on the results are: (i) the ultimate classification rule that assembles the results of the committee, and (ii) the planning of CNN's involves the adjustment of parameters that allow diversity and complementarity within the partial classification results. At the time of this paper, they propose enhancing the committee assembling with the introduction of supervised learning for the ensemble computation. The validation reveals an accuracy five percent higher with regards to the previous state of the art results supported averaging classifiers and a four percent to the bulk voting rule.

8. One of the first critical features of a human being is facial expressions, which can be used to determine the emotional state at a particular moment. The paper uses the Deep Neural Network and CNN to develop a facial emotion recognition model that categorizes a facial expression into seven various emotions: Surprised, Sad, Neutral, Happy, Disgusted, Angry, and Afraid. This paper compares the performance of 2 existing deep neural network architectures with our proposed architecture, the Venturi Architecture on the bases of training testing loss, training-loss, testing accuracy, and accuracy. The paper makes use of the Karolinska Directed Emotional Faces dataset containing a collection of 4900 pictures of human facial expressions.Two layers of feature maps were used to convolute the features from the images, and then it was passed on to the deep neural network with up to 6 hidden layers. The proposed Venturi architecture shows significant accuracy improvement compared to the modified triangular architecture and rectangular architecture.

9. Here, in this paper, they have used a method of finding out the coefficients describing elements of facial expressions. The features could also be taken from a 3D model using a Microsoft Kinect sensor. The classification of the features were performed using K-NN and MLP Neural network. Light conditions and changes of the head position are the main factors that affect the quality of the emotion recognition systems, therefore, methods in which 3D models are used are far more promising. In this experiment, Microsoft Kinect is used for 3D face modelling which has a infrared emitter and two cameras (one records visible light and the other operates in IR which measure the depth). The Kinect projects 121 strategically located points on the face which characterises the positions of the muscles on the face which is responsible for emotions. These are called the Action Units. A total of six action units are provided which detects the muscle movements. The distance between each AUs tells us the type of emotions expressed.

ES	neutral	joy	surprise	anger	sadness	fear	disgust
							
AU0	0.21	0.77	-0.10	0.30	0.17	-0.11	0.91
AU1	-0.06	0.09	0.60	-0.07	-0.04	0.20	0.13
AU2	-0.25	1.00	-0.49	0.06	-0.37	-0.60	0.88
AU3	-0.21	0.00	-0.13	0.04	-0.09	-0.17	0.00
AU4	-0.04	-0.47	0.58	-0.19	-0.02	0.28	-0.32
AU5	-0.23	-0.30	0.10	-0.34	-0.27	-0.02	-0.39

Figure 4.4: Facial Expressions and corresponding AU

Chapter 5

Architecture and Design

5.1 System Overview

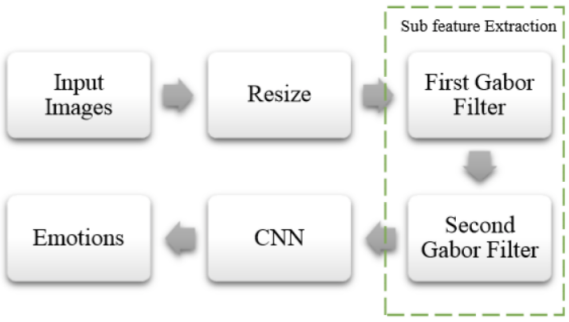

Figure 5.1: Block diagram of the process

The progress of the system is first to apply a Gabor filter to the images and then dispatch the output results as inputs to the neural network. The output of the Gabor filter is given to the convolutional neural network.

The Gabor filters extract the essential features by plotting down points at strategic locations on the face and measuring the distance between these points to determine the emotion expressed ultimately.

5.2 System Architecture

5.2.1 DataFlow Diagram

Level 0

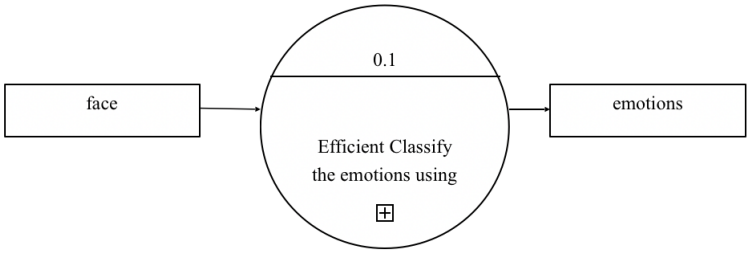

Figure 5.2: Level 0 Describes the overall process of this project. we are passing face as a input the system will classify the emotions using the CNN

Level 1

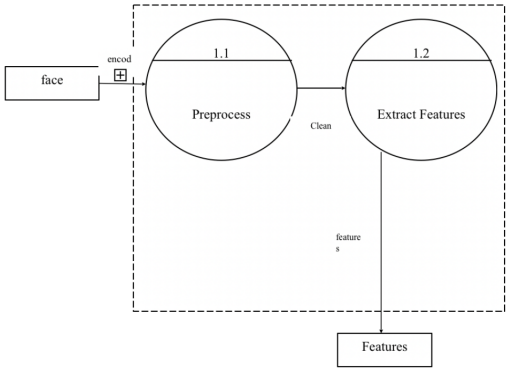

Figure 5.3: Level 1 Describes the first stage process of this project. we are passing face as a input the system will perform the preprocess and extract the important features

Level 2

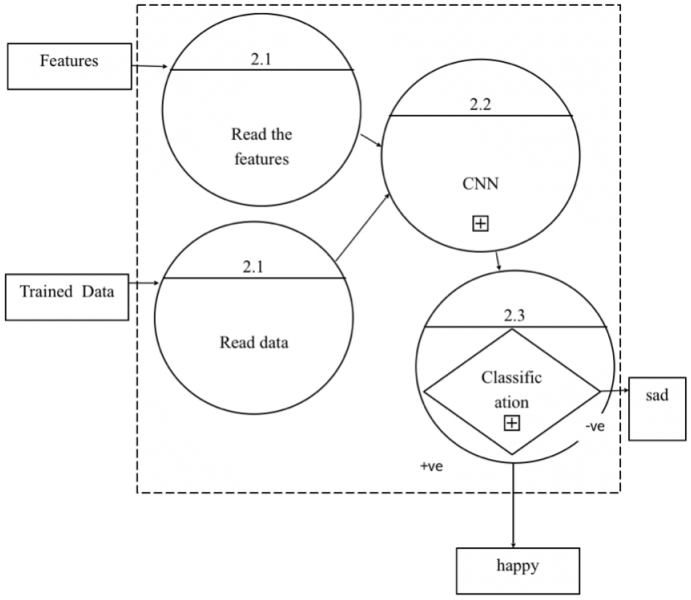

Figure 5.4: Level 2 Describes the final stage process of this project. we are passing extracted features from level 1and trained data as a input the system will classify emotions using CNN.

5.2.2 Class Diagram

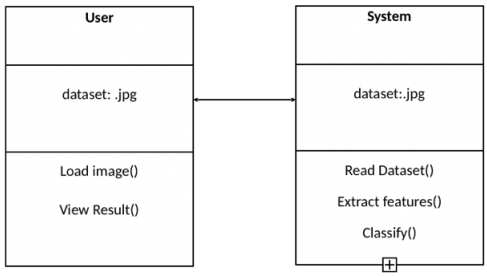

Figure 5.5: There are two classes, 'User' and 'System' which communicates with each other. The System perform the necessary functions to read datasets, extract features using there Gabor Filter and classify the image using the CNN.

5.2.3 Sequence Diagram

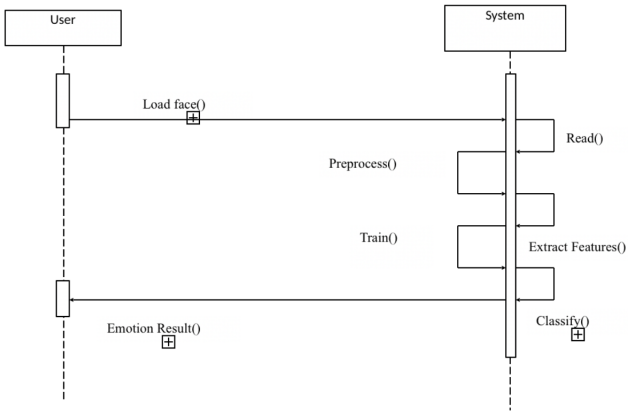

Figure 5.6: Sequence diagram

Chapter 6

Implementation

6.1 Implementation Platform

Hardware
Processor : Intel Core i5 - 2.4 Ghz
Hard Disk : 1 TB
Monitor : 13" LED
Input Devices : Keyboard, Mouse
Ram : 8 GB
System Type : 64-bit Operating System

Software
Operating system : Windows 10
Coding Language : Python
Operating Environment : Tensor flow

Here, the tensor flow environment is used where the codes written in python is executed in an anaconda prompt. Below is the screenshot of the exceution of the program in the anaconda prompt.

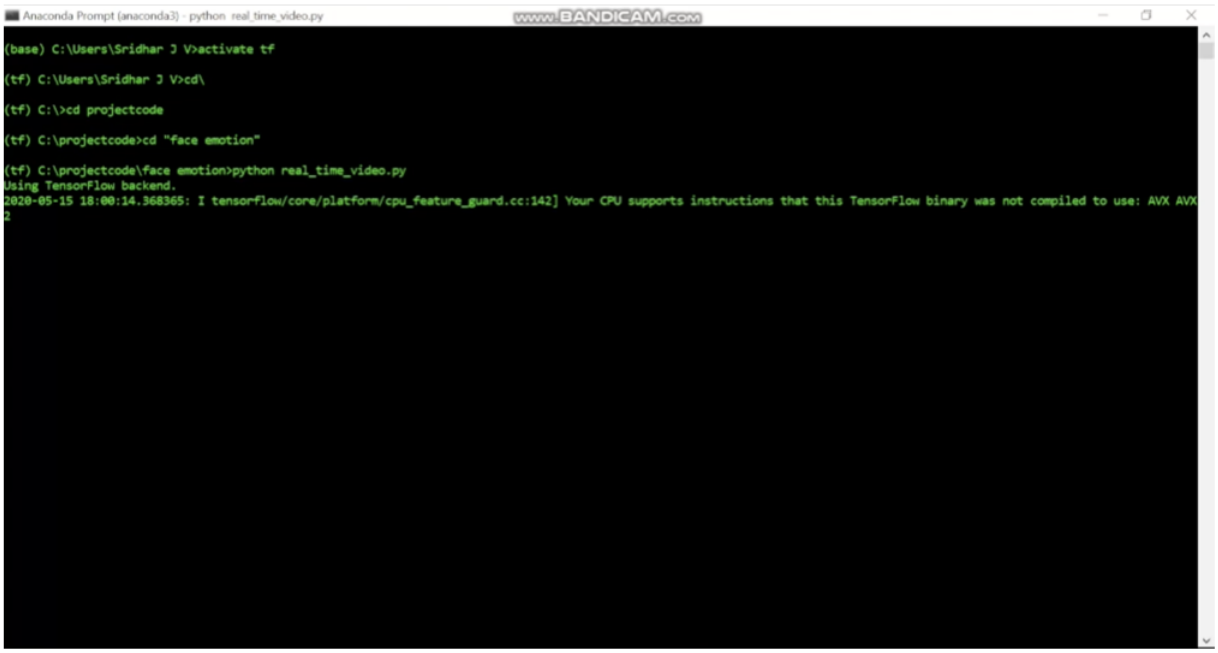

Figure 6.1: Anaconda Prompt

6.2 Methodology

6.2.1 Haar Cascade Classifier

Haar Cascade is a machine learning object detection algorithm used to identify objects in an image or video. The concept of features was proposed by Paul Viola and Michael Jones in their paper in 2001.

It is a machine learning algorithm based on the approach where a cascade function is trained from positive and negative images. Later afterward, it can be used to detect objects in other images.

The Haar Cascade algorithm has four stages:

1. Haar Feature Selection
2. Creating Integral Images
3. Adaboost Training
4. Cascading Classifiers

Haar Cascade is well known to detect faces and body parts in an image.

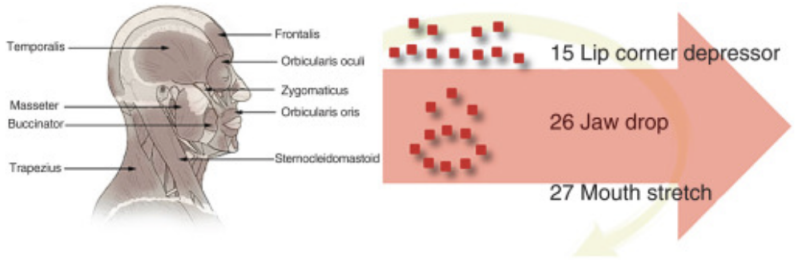

Figure 6.2: Muscles responsible for Emotions

Taking face detection as an example for a short demonstration, initially, the classifier needs a lot of positive images of faces. The negative images without faces is used to train the classifier. Combining these both images we then extract features from it. One of the main step is to collect the Haar Features. A Haar feature considers adjacent rectangular regions at a specific location in a detection window, sums up the pixel intensities in each region and calculates the difference between these sums.

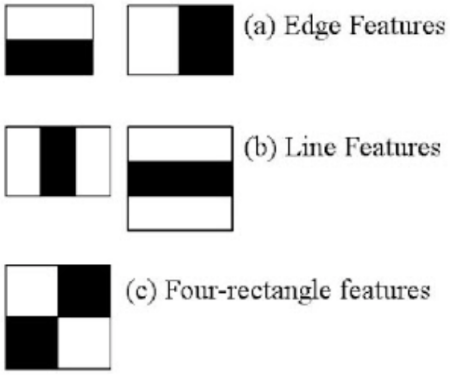

Figure 6.3: Haar Features

A fast way to used these windows is by using Intergral images.

For example, consider the image below. After the features are extracted, most of them will be irrelevant. Top rows good features. The first feature selected shows us that thr region of the eyes is often darker than the region of the nose and cheeks. The second, shows that it relies on the property that the eyes are darker than the bridge of the nose. But we can see that the same windows applying on any other part of the face is irrelevant.

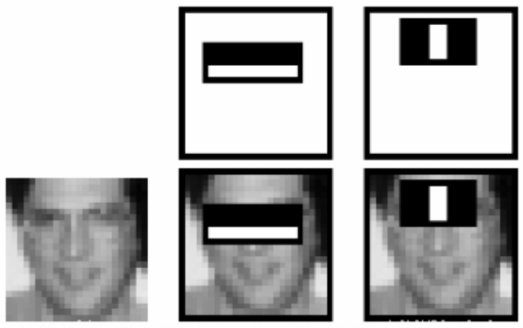

Figure 6.4: Relavent Haar Features

So out of 20000+ features extracted from the face, how do we select the best feature ?

This is achieved using Adaboost. It selects the best features and trains the classifiers that use them. Adaboost constructs a strong classifier as a linear combination of weighted simple weak classifiers. The process is as follows :
During the detection phase, a window of the target size is moved over the input image, and for each subsection of the image and Haar features are calculated. This difference is then compared to a learned threshold that separates non-objects from objects. Because each Haar feature is only a weak classifier, to describe an object with sufficient accuracy, a large number of Haar features are necessary to organized into cascade classifiers to form a strong classifier.

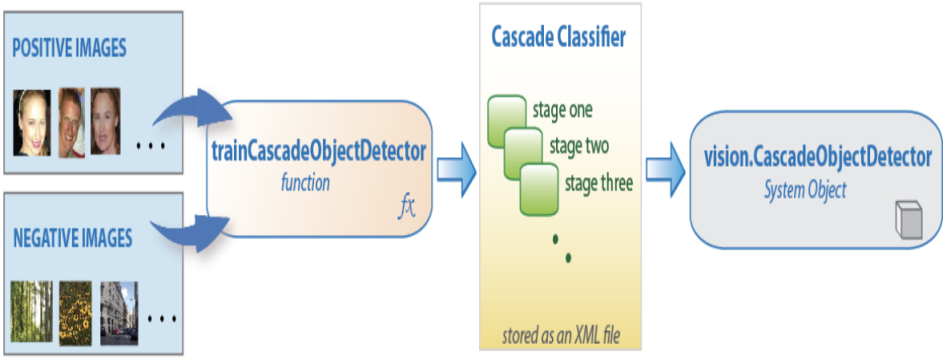

Figure 6.5: Adaboosting

The cascade classifier consists of a collection of stages. These stages are a collection of weak learners, which are called decision stumps. Each stage is trained using a technique called boosting. Boosting provides the ability to train a highly accurate classifier by taking a weighted average of the decisions made by the weak learners.

Each stage of the classifier labels the region defined by the current location of the sliding window as either positive or negative. Positive indicates that an object was found and negative indicates no objects were found. If the label is

negative, the classification of this region is complete, and the detector slides the window to the next location. If the label is positive, the classifier passes the region to the next stage. The detector reports an object found at the current window location when the final stage classifies the region as positive.

Negative samples as fast as possible, which is how each stage is designed. We assume that the vast majority of windows do not contain the object of interest. Conversely, true positives are rare and worth taking the time to verify.

1. true positive occurs when a positive sample is correctly classified.
2. false positive occurs when a negative sample is mistakenly classified as positive.
3. false negative occurs when a positive sample is mistakenly classified as negative.

To work well, each stage in the cascade must have a low false negative rate. If a stage incorrectly labels an object as negative, the classification stops, and you cannot correct the mistake. Vice versa, each stage can have a high false positive rate. If it is incorrectly labeled a nonobject as positive, another mistake is not made in following stages. Adding more stages has two consequences (1)reduces the overall false positive rate and (2)reduces the overall true positive rate.

Image Labeler can be used to label objects of interest with bounding boxes. The Image Labeler outputs a table to use for positive samples. A set of negative images can be provided from which the function generates negative samples automatically. To achieve acceptable detector accuracy, the number of stages, function parameters and feature types are set accordingly.

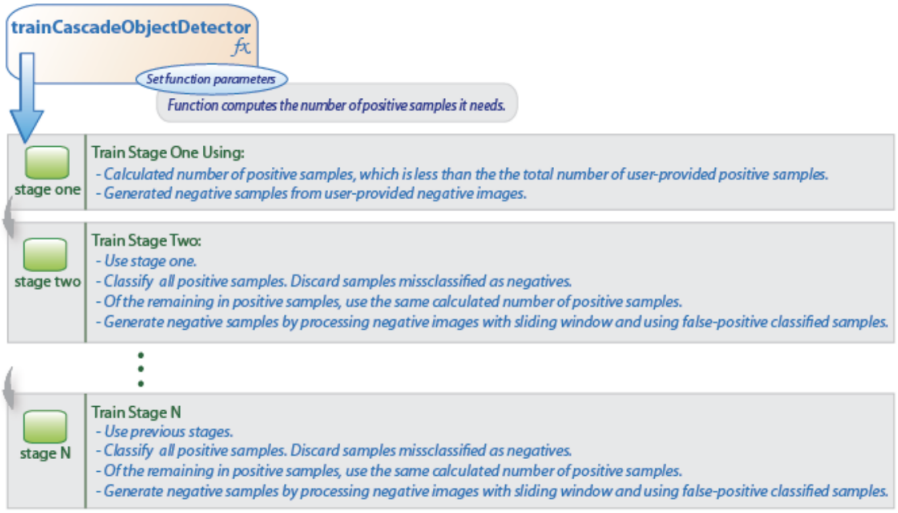

Figure 6.6: Brief Overview of the Stages of Haar Classifer

6.2.2 Gabor Filters

The proposed filter is shown in following Figure. The original image of the first Gabor filter is displayed and then the filtered image is passed again from the second Gabor filter.

Figure 6.7: Gabor Filter Stages

Gabor filter are used in texture analysis with edge detection and feature extraction. When it is applied to an image it returns the highest response at edges and points where texture changes.

The Gabor wavelet id used which is an equation maintaining a real part and an imaginary part. For obtaining 2 filters, we pass two different values each time.

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma)=\exp(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2})\exp(i(2\pi\frac{x'}{\lambda}+\psi))$$

in which the real part is:

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma)=\exp(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2})\cos(2\pi\frac{x'}{\lambda}+\psi)$$

and the imaginary part is:

$$g(x,y;\lambda,\theta,\psi,\sigma,\gamma)=\exp(-\frac{x'^2+\gamma^2y'^2}{2\sigma^2})\sin(2\pi\frac{x'}{\lambda}+\psi)$$

where:

$$x'=x\cos\theta+y\sin\theta$$

and

$$y'=-x\sin\theta+y\cos\theta$$

Figure 6.8: Gabor Wavelet Equation

Following set of images shows how the filtering works.
The following parameters are applied to the equation

$$(x, y) = (18,18),\ \sigma = 1.5,\ \theta = \pi / 4,\ \lambda = 5,\ \gamma = 1.5$$
$$\psi = 0$$

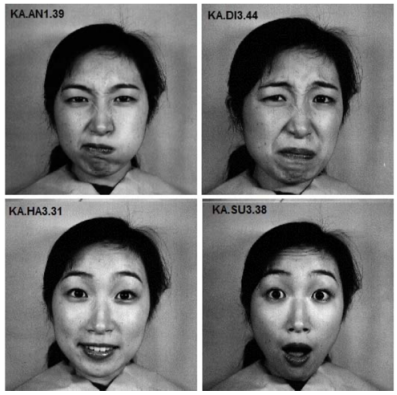

Figure 6.9: Original Images

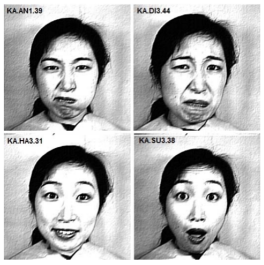

Figure 6.10: Image after applying first Gabor Filter

Now again, the following parameters are applied.

$$(x, y) = (18,18),\ \sigma = 1.5,\ \theta = 3\pi / 4,\ \lambda = 5,\ \gamma = 1.5$$
$$\psi = 0$$

Figure 6.11: Images after applying Second Gabor Filter

6.2.3 Convolutional Neural Network

For reference, the structure of the neural network is shown in the figure below. The image is received from the Gabor filter. The feature-extracted images is passed through different layers of the CNN and the learning process returns a vector with seven modes as output: Angry, Disgust, Fear, Happy, Neutral, Sad and Surprise.

In the first stage, the deep neural network applies a convolution of a 6 6 filter on the image. In the next step, MaxPooling reduces the dimensions to 128 128 6. Next, another convolutional network will be applied to the 16 16 filter size, and in the next step, using the MaxPooling function, we will reduce the size to 64 64 16. The next convolution is applied to the data with a 120 120 filter size. In the next step, using the Flatten function, all data is converted to a vector of the size 432000. Then, the vector is converted to a vector of length 84, and at the end, it is reduced to seven, which is the 7 categories of emotional states.

In brief, the main functions applied in the network are:

1. MaxPooling Function
2. Flatten Function
3. Softmax Function
4. Rectified Linear Unit (Relu) function

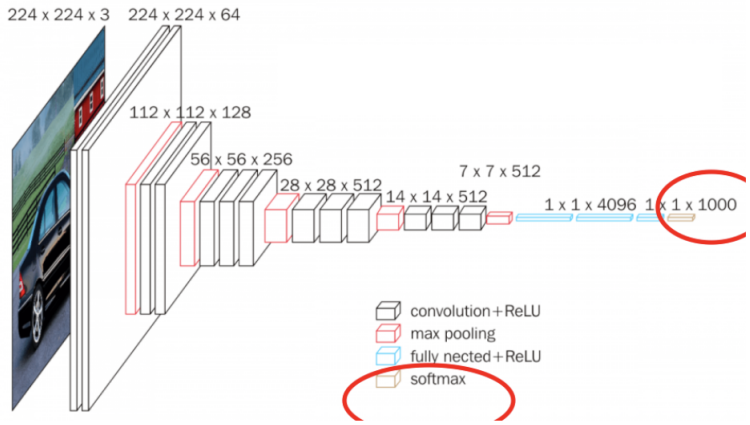

Figure 6.12: CNN Architecture

Layer type	Details	Output Shape
Conv	Conv (6x6)	256, 256, 6
Activation	Relu	256, 256, 6
MaxPooling	Pool size (2,2)	128, 128, 6
Conv	Conv (16x16)	128, 128, 16
Activation	Relu	128, 128, 16
MaxPooling	Pool size (2,2)	64, 64, 16
Conv	Conv (120x120)	60, 60, 120
Activation	Relu	60, 60, 120
Dropout	----------	60, 60, 120
Flatten	Flatten to a vector	432000
Dense	Input →84	84
Activation	Relu	84
Dropout	----------	84
Dense	Input → Classese Num =7	7
activation	softmax	7

Figure 6.13: CNN Architecture Method

Chapter 7

Testing

During the implementation when we run the main program in a tensor flow environment, two small windows pop up. One of which displays the probability of all the emotions expressed live. The other is the canvas window displayed from the webcam which displays our face and the emotion state expressed. A sample window is shown below.

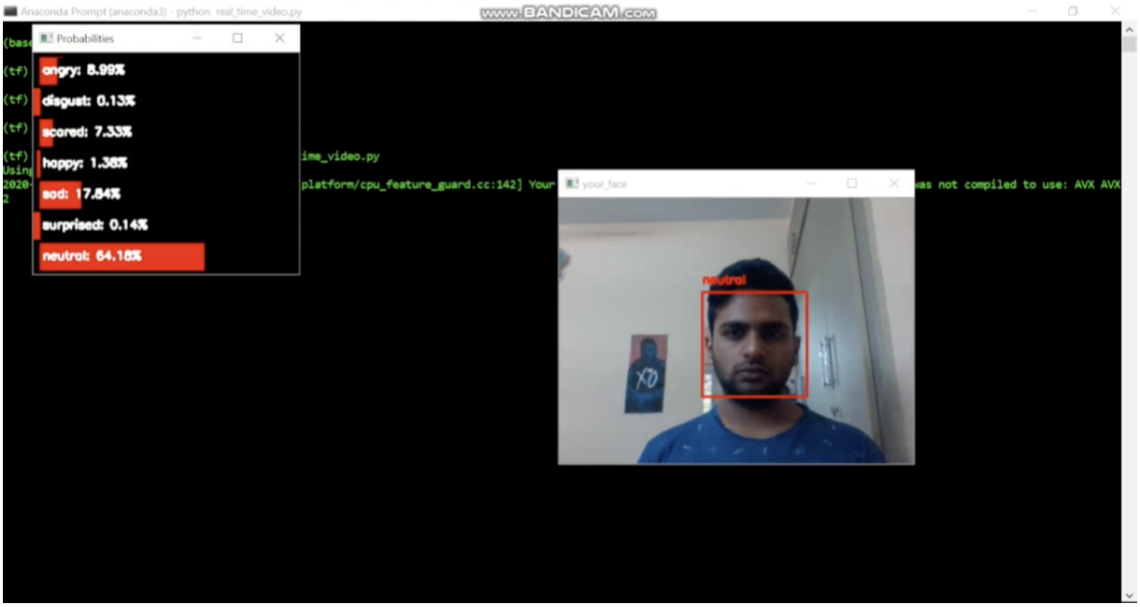

Figure 7.1: Output Window

Here we go about testing the various scenarios that the program is functioned to accomplish to satisfy the objectives.

These are presented one by one in a series of Test cases, They are :

1. Identify and detect a face.
2. Track the face successfully when moving around.
3. Identify all the 7 emotion states live and accurately.

Failure Cases to identify during:

1. No Face present in front of the webcam.
2. Low lighting conditions.
3. Wearing spectacles/mask.

We can run these same test without the use of Gabor Filter to compare the accuracy between CNNs with and without the use of a Gabor Filter and plot harts to analyse the data accordingly.

7.1 Test Cases

7.1.1 Test Case 1

Identify and detect a face which is captured through the inbuilt webcam and displayed on the screen with a box around the face and the respective emotion displayed.

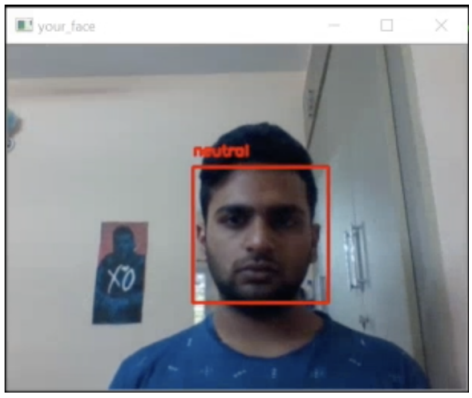

Figure 7.2: Test Case 1

Here, we can see that the test is passed by successfully detecting the face.

7.1.2 Test Case 2

Track the face successfully in motion when moving around in all the axes.

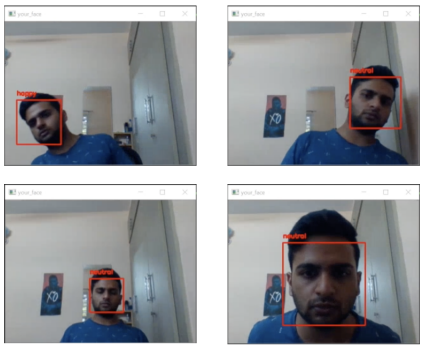

Figure 7.3: Test Case 2

Here, we can see that the test is passed by successfully tracking the face.

7.1.3 Test Case 3

Identify 7 emotions live one by one.

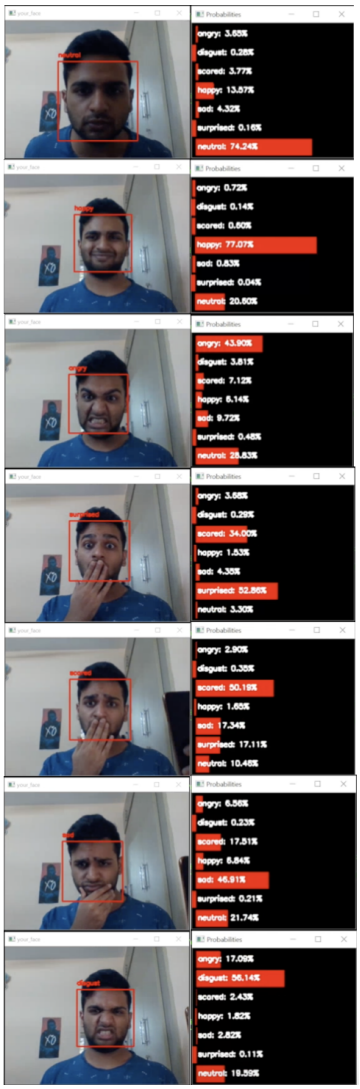

Figure 7.4: Test Case 3

Here, we see that all 7 states were recognised accurately which passes this test case.

7.1.4 Test Case 4

No face present in front of the Webcam.

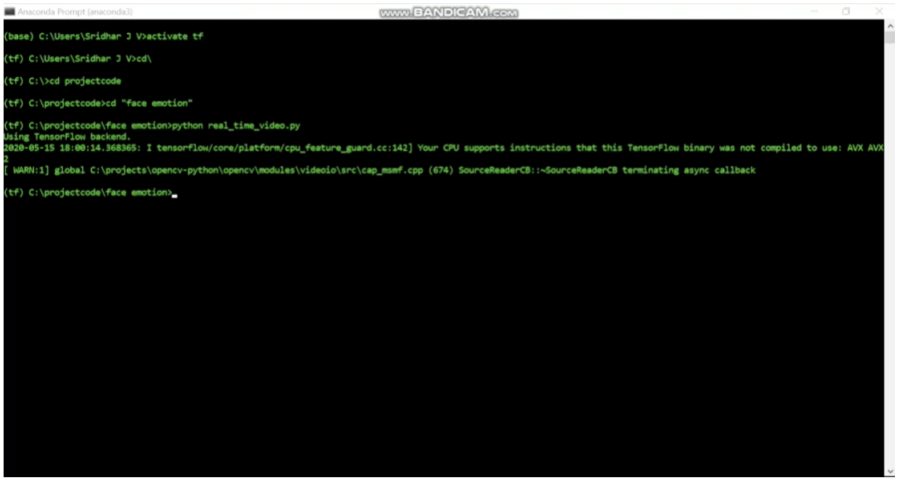

Figure 7.5: Test Case 4

Here, we see that the prompt produced an error which fails the condition of at least 1 face to be present in front of the webcam.

7.1.5 Test Case 5

Recognise all the emotions in low lighting conditions.

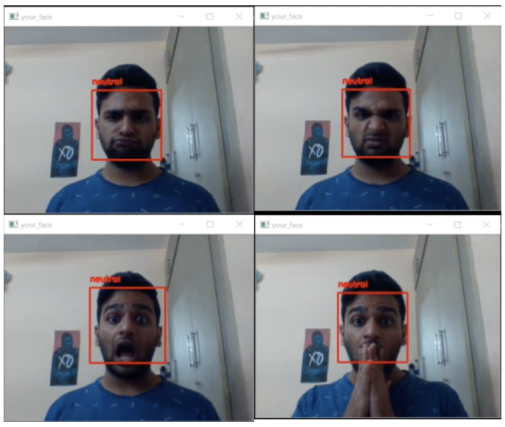

Figure 7.6: Test Case 5

Here, we see that in low lighting conditions, the program has a hard time to recognise the strategic spots on the face which disrupts the readings and produces inaccurate results.

7.1.6 Test Case 6

When the person wears spectacles or anything on the face.

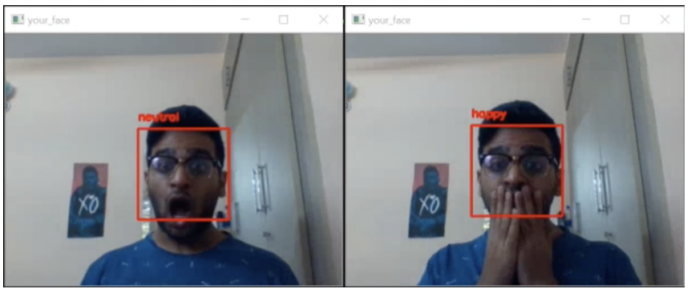

Figure 7.7: Test Case 6

Here, we can see that by wearing foreign objects, the computer has a hard time to read our face. The computer categories these objects as a part of our face but cannot infer natural features of the face like eyes and eyebrows which disrupts the reads and lowers the accuracy.

Chapter 8

Results

8.1 2GF+CNN vs CNN

After the testing phase, we collect the related data like the time to recognise, accuracy, prediction etc. A similar test is conducted in another paper but here the Gabor filter is not used. The data without the Gabor filter is analysed and plotted to derive the following conclusions.

The experiment is conducted and the number of trials is converted into 'epochs' where at certain number of trails epochs are taken.

Epoch	CNN Method Accuracy	2Gabor + CNN Method
1	0.1050	0.1326
10	0.5138	0.8619
15	0.7348	0.9227
20	0.8343	0.9558
25	0.9006	0.9779
30	0.9116	0.9716

Figure 8.1: Comparison Accuracy

Accuracy	CNN Method		2Gabor + CNN	
	Epoch	Time(s)	Epoch	Time(s)
10-15%	1	42	1	41
50-55%	9	330	6	232
70-73%	14	521	8	308
86-87%	18	683	10	390
91-92%	30	1189	14	541

Figure 8.2: Comparison of Speed

The accuracy of 70-73 percent is achieved after 8 epochs with 308 seconds using the proposed approach. It takes 521 seconds using the convolutional method. In addition, the accuracy of 91-92 percent takes 541 seconds to be achieved using the proposed approach and it takes 1189 seconds using the convolutional method.

A graph of accuracy vs epoch of 2Gabor filter+CNN and only CNN can illuminate the difference between the two methods.

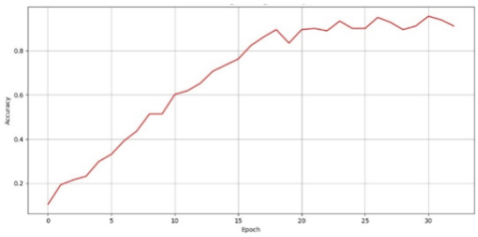

Figure 8.3: Comparison Accuracy

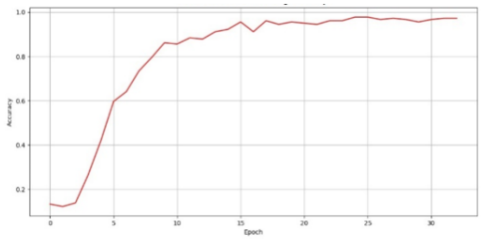

Figure 8.4: Comparison of Speed

Therefore, the 2GF+CNN needs less time than the convolutional approach to achieve the same accuracy. Increase of running time in the proposed method is expected because the sub-features extracted by two consecutive Gabor filters simplify the learning process (feature extraction and classification) of the CNN. The first used Gabor filter extracts texture information in different scales and directions while the second one extracts more sub features from the previous Gabor feature gap.

8.2 Confusion of Emotions

From the experiment we could see that few emotions were confused with the other emotion. We could say this because the probability of these two emotions were almost the same. Therefore the computer had a hard time figuring which emotion to be displayed accurately. The following information briefly addresses that topic.

In order to determine which emotions are the easiest and which the most difficult to distinguish, it is necessary to calculate the confusion matrices. Here each emotion is paired up with each other to determine its accuracy and riddle out the 'true positives'. From the confusion matrices of classification in pairs, it is evident that :

1. Sadness was confused with neutral emotion.
2. Fear was confused with surprised emotion.

Emotions	neutral	joy	surprise	anger	sadness	fear	disgust
neutral	-	0.86	0.94	0.85	**0.71**	0.93	0.85
joy	0.86	-	0.98	0.84	0.84	0.98	0.83
surprise	0.94	0.98	-	0.98	0.94	**0.75**	0.95
anger	0.85	0.84	0.98	-	0.87	0.97	0.84
sadness	**0.71**	0.84	0.94	0.87	-	0,89	0.86
fear	0.93	0.98	**0.75**	0.97	0.89	-	0.94
disgust	0.85	0.83	0.95	0.84	0.86	0.94	-

Figure 8.5: The Classification of Accuracy in pairs

8.3 Effect of Head Roation

Head orientation also played an important role with the accuracy and classification of the emotions. Changing the head orientation in relation to x and y axis may cause that a part of the user's face is not visible to the Webcam.

We examined the effect of changes of head orientation. For this purpose extra session was registered, during which the examined I was not sitting in front of the webcam, but at a certain angle.

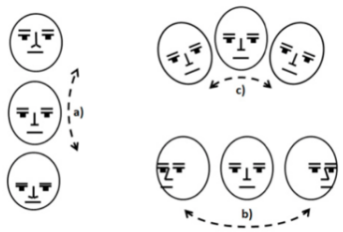

Figure 8.6: Rotation of head in a)Y- axis b)X-axis and c)Z-axis

We could conclude that the effect of head orientation will alter the accuracy of the emotion. Change in the orientation of the head in any axes would lower the accuracy of the classified emotion.

Chapter 9

Conclusion

After the Gabor filter applied, the system learning became faster and the accuracy has improved. The learning speed of the convolutional neural network has increased immensely. This is because the Gabor filter actually extracts the image sub feature and gives the neural network. By doing this, the convolutional neural network receives a number of sub feature and takes one step further in extracting the emotions from the faces.

Accuracy was influenced by the way users play facial expressions.

Certainly, the classification accuracy was influenced by the way users play specific facial expressions. In real conditions the classification accuracy can be affected by many additional factors. When you feel real emotions, facial expressions can vary greatly - may be exposed to a greater or lesser extent.

9.1 Future Works

The proposed system of emotion classification has already founds its niche in the market in several industry. But there is still more innovative and motivating ways to incorporate these into growing future technologically advanced areas. Few of ways where it could find its applications are:

1. Cars, as a safety feature to detect rash/sleepy drivers.
2. Music recommendation where the suitable music is played according to the persons emotion.
3. Social media filters like Instagram/Snapchat to use as animated GIFs and Images.
4. In psychological counselling sessions where patient can be diagnosed from many sociological and emotional conditions.
5. In offices and commercial organisation to detect any hostile and corrosive environment. Etc.

References

[1] S. L. Happy and A. Routray, Automatic facial expression recognition using features of salient facial patches, in IEEE Transactions on Affective Computing, vol. 6, no. 1, pp. 1-12, 1 Jan.-March 2015, doi: 10.1109/TAFFC.2014.2386334.

[2] A. Nicolai and A. Choi, Facial Emotion Recognition Using Fuzzy Systems, 2015 IEEE International Conference on Systems, Man, and Cybernetics, Kowloon, 2015, pp. 2216-2221, doi: 10.1109/SMC.2015.387.

[3] M. Imani and G. A. Montazer, GLCM features and fuzzy nearest neighbor classifier for emotion recognition from face, 2017 7th International Conference on Computer and Knowledge Engineering (ICCKE), Mashhad, 2017, pp. 8-13, doi: 10.1109/ICCKE.2017.8167879.

[4] Kwolek B. (2005) Face Detection Using Convolutional Neural Networks and Gabor Filters. In: Duch W., Kacprzyk J., Oja E., Zadrożny S. (eds) Artificial Neural Networks: Biological Inspirations – ICANN 2005. ICANN 2005. Lecture Notes in Computer Science, vol 3696. Springer, Berlin, Heidelberg.

[5] H. Li, J. Sun, Z. Xu and L. Chen, Multimodal 2D+3D Facial Expression Recognition With Deep Fusion Convolutional Neural Network, in IEEE Transactions on Multimedia, vol. 19, no. 12, pp. 2816-2831, Dec. 2017, doi: 10.1109/TMM.2017.2713408.

[6] Tzirakis, Panagiotis Trigeorgis, George Nicolaou, Mihalis Schuller, Björn Zafeiriou, Stefanos. (2017). End-to-End Multimodal Emotion Recognition Using Deep Neural Networks. IEEE Journal of Selected Topics in Signal Processing. PP. 10.1109/JSTSP.2017.2764438.

[7] G. Pons and D. Masip, Supervised Committee of Convolutional Neural Networks in Automated Facial Expression Analysis, in IEEE Transactions on Affective Computing, vol. 9, no. 3, pp. 343-350, 1 July-Sept. 2018, doi: 10.1109/TAFFC.2017.2753235.

[8] A. Verma, P. Singh and J. S. Rani Alex, Modified Convolutional Neural Network Architecture Analysis for Facial Emotion Recognition, 2019 International Conference on Systems, Signals and Image Processing (IWSSIP), Osijek, Croatia, 2019, pp. 169-173, doi: 10.1109/IWSSIP.2019.8787215.

[9] Tarnowski, Paweł Kołodziej, Marcin Majkowski, Andrzej Rak, Remigiusz. (2017). Emotion recognition using facial expressions. Procedia Computer Science. 108. 1175-1184. 10.1016/j.procs.2017.05.025.

[10] M. M. Taghi Zadeh, M. Imani and B. Majidi, Fast Facial emotion recognition Using Convolutional Neural Networks and Gabor Filters, 2019 5th Conference on Knowledge Based Engineering and Innovation (KBEI), Tehran, Iran, 2019, pp. 577-581, doi: 10.1109/KBEI.2019.8734943.

[11] M. H. Abbasi, B. Majidi, and M. T. Manzuri, Deep cross altitude visual interpretation for service robotic agents in smart city, in 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), 2018, pp. 79-82.

[12] M. H. Abbasi, B. Majidi, and M. T. Manzuri, Glimpse-gaze deep vision for Modular Rapidly Deployable Decision Support Agent in smart jungle, in 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), 2018, pp. 75-78.

[13] A. Fadaeddini, M. Eshghi, and B. Majidi, A deep residual neural network for low altitude remote sensing image classification, in 2018 6th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), 2018, pp. 43-46.

[14] T. Wu, M. S. Bartlett, and J. R. Movellan, Facial expression recognition using Gabor motion energy filters, in 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, 2010, pp. 42-47.

[15] S. Li, W. Deng, and J. Du, Reliable Crowdsourcing and Deep Locality Preserving Learning for Expression Recognition in the Wild, in 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2584-2593.

[16] Ratliff M. S., Patterson E., Emotion recognition using facial expressions with active appearance models, Proceedings of the Third IASTED International Conference on Human Computer Interaction, ACTA Press, Anaheim, CA, USA, 2008, 138–143.

[17] Tian Y. I., Kanade T., Cohn J. F., Recognizing action units for facial expression analysis, IEEE Transactions on Pattern Analysis and Machine Intelligence, 23 (2001), no. 2, 97–115.

[18] Mao Q., Pan X., Zhan Y., Shen X., Using Kinect for real-time emotion recognition via facial expressions, Frontiers Inf Technol Electronic Eng, 16 (2015), no. 4, 272–282.

[19] Li B. Y. L., Mian A. S., Liu W., Krishna A., Using Kinect for face recognition under varying poses, expressions, illumination and disguise, 2013 IEEE Workshop on Applications of Computer Vision (WACV), 2013, 186–192.

[20] Koelstra S., Patras I., Fusion of facial expressions and EEG for implicit affective tagging, Image and Vision Computing, 31 (2013), no. 2, 164–174.

Appendix A

Code

real_time_video.py

```
from keras.preprocessing.image import img_to_array
import imutils
import cv2
from keras.models import load_model
import numpy as np

# parameters for loading data and images
detection_model_path = 'haarcascade_files/haarcascade_frontalface_default.xml'
emotion_model_path = 'models/_mini_XCEPTION.106-0.65.hdf5'

# hyper-parameters for bounding boxes shape
# loading models
face_detection = cv2.CascadeClassifier(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)
EMOTIONS = ["angry" ,"disgust","scared", "happy", "sad", "surprised",
 "neutral"]


# starting video streaming
cv2.namedWindow('your_face')
camera = cv2.VideoCapture(0)
while True:
    frame = camera.read()[1]
    #reading the frame
```

```
frame = imutils.resize(frame,width=400)
gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
faces = face_detection.detectMultiScale(gray,scaleFactor=1.1,minNeighbors=5,

canvas = np.zeros((250, 300, 3), dtype="uint8")
frameClone = frame.copy()
if len(faces) > 0:
    faces = sorted(faces, reverse=True,
    key=lambda x: (x[2] - x[0]) * (x[3] - x[1]))[0]
    (fX, fY, fW, fH) = faces
                # Extract the ROI of the face from the grayscale image, resi
        # the ROI for classification via the CNN
    roi = gray[fY:fY + fH, fX:fX + fW]
    roi = cv2.resize(roi, (48, 48))
    roi = roi.astype("float") / 255.0
    roi = img_to_array(roi)
    roi = np.expand_dims(roi, axis=0)


    preds = emotion_classifier.predict(roi)[0]
    emotion_probability = np.max(preds)
    label = EMOTIONS[preds.argmax()]


for (i, (emotion, prob)) in enumerate(zip(EMOTIONS, preds)):
            # construct the label text
            text = "{}: {:.2f}%".format(emotion, prob * 100)
            w = int(prob * 300)
            cv2.rectangle(canvas, (7, (i * 35) + 5),
            (w, (i * 35) + 35), (0, 0, 255), -1)
            cv2.putText(canvas, text, (10, (i * 35) + 23),
            cv2.FONT_HERSHEY_SIMPLEX, 0.45,
            (255, 255, 255), 2)
            cv2.putText(frameClone, label, (fX, fY - 10),
            cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)
            cv2.rectangle(frameClone, (fX, fY), (fX + fW, fY + fH),
                          (0, 0, 255), 2)

cv2.imshow('your_face', frameClone)
cv2.imshow("Probabilities", canvas)
if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
        break

camera.release()
cv2.destroyAllWindows()
```

train_emotion_classifier.py

```
from keras.callbacks import CSVLogger, ModelCheckpoint, EarlyStopping
from keras.callbacks import ReduceLROnPlateau
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from keras.layers import Activation, Convolution2D, Dropout, Conv2D
from keras.layers import AveragePooling2D, BatchNormalization
from keras.layers import GlobalAveragePooling2D
from keras.models import Sequential
from keras.layers import Flatten
from keras.models import Model
from keras.layers import Input
from keras.layers import MaxPooling2D
from keras.layers import SeparableConv2D
from keras import layers
from keras.regularizers import l2
import pandas as pd
import cv2
import numpy as np

dataset_path = 'fer2013/fer2013.csv'
image_size=(48,48)
# parameters
batch_size = 32 #to perform any normalizing operations
num_epochs = 110
input_shape = (48, 48, 1) #frame
validation_split = .2
verbose = 1
num_classes = 7
patience = 50
base_path = 'models/'
l2_regularization=0.01


def load_fer2013():
    data = pd.read_csv(dataset_path) #reads the dataset
        pixels = data['pixels'].tolist() #enumerates a new list
        width, height = 48, 48 #frame size
faces = []
```

```
for pixel_sequence in pixels:
face = [int(pixel) for pixel in pixel_sequence.split(' ')]
face = np.asarray(face).reshape(width, height)
face = cv2.resize(face.astype('uint8'),image_size)
faces.append(face.astype('float32'))
faces = np.asarray(faces)
    faces = np.expand_dims(faces, -1) #expand shape of the array (a,axis)
emotions = pd.get_dummies(data['emotion']).as_matrix()
return faces, emotions

def preprocess_input(x, v2=True):
    x = x.astype('float32')
    x = x / 255.0
    if v2:
        x = x - 0.5
        x = x * 2.0
    return x



# data generator
data_generator = ImageDataGenerator(
                        featurewise_center=False,
                        featurewise_std_normalization=False,
                        rotation_range=10,
                        width_shift_range=0.1,
                        height_shift_range=0.1,
                        zoom_range=.1,
                                    horizontal_flip=True) #create modified versi

# model parameters/compilation
# model = mini_XCEPTION(input_shape, num_classes)
regularization = l2(l2_regularization)

# base
img_input = Input(input_shape)
x = Conv2D(8, (3, 3), strides=(1, 1), kernel_regularizer=regularization, use_bia
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = Conv2D(8, (3, 3), strides=(1, 1), kernel_regularizer=regularization, use_bia
x = BatchNormalization()(x)
```

```
x = Activation('relu')(x)

# module 1
residual = Conv2D(16, (1, 1), strides=(2, 2), padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(16, (3, 3), padding='same', kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(16, (3, 3), padding='same', kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 2
residual = Conv2D(32, (1, 1), strides=(2, 2), padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(32, (3, 3), padding='same', kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(32, (3, 3), padding='same', kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 3
residual = Conv2D(64, (1, 1), strides=(2, 2),padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(64, (3, 3), padding='same',kernel_regularizer=regularization
x = BatchNormalization()(x)
x = Activation('relu')(x)
x = SeparableConv2D(64, (3, 3), padding='same',kernel_regularizer=regularization
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])

# module 4
residual = Conv2D(128, (1, 1), strides=(2, 2),padding='same', use_bias=False)(x)
residual = BatchNormalization()(residual)
x = SeparableConv2D(128, (3, 3), padding='same',kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = Activation('relu')(x)
```

```
x = SeparableConv2D(128, (3, 3), padding='same',kernel_regularizer=regularizatio
x = BatchNormalization()(x)
x = MaxPooling2D((3, 3), strides=(2, 2), padding='same')(x)
x = layers.add([x, residual])
x = Conv2D(num_classes, (3, 3), padding='same')(x)
x = GlobalAveragePooling2D()(x)
output = Activation('softmax',name='predictions')(x)

model = Model(img_input, output)
model.compile(optimizer='adam', loss='categorical_crossentropy',metrics=['accura
model.summary()

# callbacks : function done during train,test,predict
log_file_path = base_path + '_emotion_training.log'
csv_logger = CSVLogger(log_file_path, append=False)
early_stop = EarlyStopping('val_loss', patience=patience)
reduce_lr = ReduceLROnPlateau('val_loss', factor=0.1, patience=int(patience/4),
trained_models_path = base_path + '_mini_XCEPTION'
model_names = trained_models_path + '.{epoch:02d}-{val_acc:.2f}.hdf5'
model_checkpoint = ModelCheckpoint(model_names, 'val_loss', verbose=1,save_best_
callbacks = [model_checkpoint, csv_logger, early_stop, reduce_lr]

# loading dataset
faces, emotions = load_fer2013()
faces = preprocess_input(faces)
num_samples, num_classes = emotions.shape
xtrain, xtest,ytrain,ytest = train_test_split(faces, emotions,test_size=0.2,shuf
model.fit_generator(data_generator.flow(xtrain, ytrain,
                                        batch_size),
                        steps_per_epoch=len(xtrain) / batch_size,
                        epochs=num_epochs, verbose=1, callbacks=callbacks,
                        validation_data=(xtest,ytest))
```

facial_emotion_image.py

```
    from keras.preprocessing.image import img_to_array
from keras.models import load_model
import imutils
import cv2
import numpy as np
import sys

# parameters for loading data and images
detection_model_path = 'haarcascade_files/haarcascade_frontalface_default.xml'
emotion_model_path = 'models/_mini_XCEPTION.106-0.65.hdf5'
img_path = sys.argv[1]

# hyper-parameters for bounding boxes shape
# loading models
face_detection = cv2.CascadeClassifier(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)
EMOTIONS = ["angry","disgust","scared", "happy", "sad", "surprised","neutral"]


#reading the frame
orig_frame = cv2.imread(img_path)
frame = cv2.imread(img_path,0)
faces = face_detection.detectMultiScale(frame,scaleFactor=1.1,minNeighbors=5,min

if len(faces) > 0:
    faces = sorted(faces, reverse=True,key=lambda x: (x[2] - x[0]) * (x[3] - x[1
    (fX, fY, fW, fH) = faces
    roi = frame[fY:fY + fH, fX:fX + fW]
    roi = cv2.resize(roi, (48, 48))
    roi = roi.astype("float") / 255.0
    roi = img_to_array(roi)
    roi = np.expand_dims(roi, axis=0)
    preds = emotion_classifier.predict(roi)[0]
    emotion_probability = np.max(preds)
    label = EMOTIONS[preds.argmax()]
    cv2.putText(orig_frame, label, (fX, fY - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.45
    cv2.rectangle(orig_frame, (fX, fY), (fX + fW, fY + fH),(0, 0, 255), 2)

cv2.imshow('test_face', orig_frame)
```

```
cv2.imwrite('test_output/'+img_path.split('/')[-1],orig_frame)
if (cv2.waitKey(2000) & 0xFF == ord('q')):
    sys.exit("Thanks")
cv2.destroyAllWindows()
```

csv_to_images.py

```
import os
import csv
import argparse
import numpy as np
import scipy.misc

parser = argparse.ArgumentParser()
parser.add_argument('-f', '--file', required=True, help="path of the csv file")
parser.add_argument('-o', '--output', required=True, help="path of the output di
args = parser.parse_args()

w, h = 48, 48
image = np.zeros((h, w), dtype=np.uint8)
id = 1

with open(args.file) as csvfile:
    datareader = csv.reader(csvfile, delimiter =',')
    next(datareader,None)

    for row in datareader:

        emotion = row[0]
        pixels = row[1].split()
        usage = row[2]
        pixels_array = np.asarray(pixels, dtype=np.int)

        image = pixels_array.reshape(w, h)
        #print image.shape

        stacked_image = np.dstack((image,) * 3)
        #print stacked_image.shape

        image_folder = os.path.join(args.output, usage)
        if not os.path.exists(image_folder):
            os.makedirs(image_folder)
        image_file =  os.path.join(image_folder , str(id)+'_'+emotion+'.jpg')
        scipy.misc.imsave(image_file, stacked_image)
```

```
        id += 1
        if id % 100 == 0:
            print('Processed {} images'.format(id))

print("Finished processing {} images".format(id))
```