MAIN PROJECT REPORT ON

Real-Time Emotion Detection from Face

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

Bachelor of Technology

In

Computer Science And Engineering

By

Sainjal Poly

Muhammad Abilaj K.E

Manu Varghese

Sachin Jacob

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY

(FISAT)®

ANGAMALY-683577, ERNAKULAM (DIST)

Affiliated to

MAHATMA GANDHI UNIVERSITY

Kottayam-686560

May 2018

FEDERAL INSTITUTE OF SCIENCE AND TECHNOLOGY (FISAT)®

Mookkannor(P.O), Angamaly-683577

CERTIFICATE

This is to certify that project report titled **Real-Time Emotion Detection from Face** is a bonafide work carried out by **Sainjal Poly(14004105)** in partial fulfilment for the award of Bachelor of Technology in Computer Science and Engineering from Mahatma Gandhi University, Kottayam, Kerala during the academic year 2017-2018.

Dr. Prasad J C

Staff-In-charge **Head of the Department**

Place:

Date:

Internal Examiner **External Examiner**

ABSTRACT

The existing system for Emotion Recognition uses explicit method for Feature Extraction purpose with is really challenging.So our proposed system is capable of implicitly extracting the features from the training data using a conventional neural network.In Conventional network automatically extract the feature from training data set.The Project can be used to identify different emotions from face.It mainly considers 7 emotions-consists of angry,sad,happy,disgusted,surprise,fear and neutral.Real-time system captures the frames from web camera and identifies the face coordinates and detect the emotion from face.We expect a good accuracy for the prediction model.

CONTRIBUTION OF THE AUTHOR

Manu Varghese : CNN used for the classification of images.Construction of CNN layers was done for the image property training.Facial localization is done using facial texture information and facial grey scale
Sachin Jacob : Splitted the entire application modules to two.Training module for training and testing module for the correctness of the model built.Downloaded the emotion data set from internet.Parameters for loading data and images has been adjusted
Sainjal Poly : Even after the building of the model,changed the parameters associated with the model constantly for the improvement in efficiency.Because too much variance in the parameters lead to overfitting and was resulted in poor prediction.Testing of the model was done constantly for the accurate image prediction model.
Muhammed Abilaj K E : To extract the hidden properties of the images applied normalization.So image processing tools like Adaptive histogram Equalization was applied on all the images before their processing.Only after that they were ready for the training and testing purpose.

ACKNOWLEDGEMENT

Apart from the efforts put in by us, the success of this project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project:

Mr.Paul Mundadan, chairman, FISAT Governing Body, who provided us with the vital facilities required by the project right from inception to completion.

Dr. George Issac, Principal, FISAT for the amenities he provided, which helped us in the fulfillment of our project.

Dr. Prasad J.C, HOD(CSE Dept), FISAT who always guided us and rendered his help in all phases of our project.

Mr.Pankaj Kumar G, for his constant encouragement and enthusiastic supervision and for guiding us with patience in all the stages.Without his help and inspiration, this would not have been materialized.

Mrs. Divya John ,Mrs Resmi R ,Mrs. Preethi N P and Mr. Paul P Mathai and for their guidance and constant supervision as well as for providing necessary information regarding the project and also for their support in completing the project. The faculty of the CSE Dept., FISAT and Lab Instructors for providing us with the necessary Lab facilities and helping us throughout this project.

Our families who inspired, encouraged and fully supported us in every trial that came our way. Also, we thank them for giving us not just financial, but moral and spiritual support.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

The success of service robotics decisively depends on a smooth robot to user interaction. Thus, a robot should be able to extract information just from the face of its user, e.g. identify the emotional state or deduce gender. Interpreting correctly any of these elements using machine learning (ML) techniques has proven to be complicated due the high variability of the samples within each task . This leads to models with millions of parameters trained under thousands of samples [3]. Furthermore, the human accuracy for classifying an image of a face in one of 7 different emotions is 65% + 5% [4]. One can observe the difficulty of this task by trying to manually classify the FER-2013 dataset images in within the following classes {"angry", "disgust", "fear", "happy", "sad", "surprise", "neutral"}. In spite of these difficulties, robot platforms oriented to attend and solve household tasks require facial expressions systems that are robust and computationally efficient. More-over, the state-of-the-art methods in image-related tasks such as image classification [1] and object detection are all based on Convolutional Neural Networks (CNNs). These tasks require CNN architectures with millions of parameters; therefore, their deployment in robot platforms and real-time systems becomes unfeasible. The proposed system implement a general CNN building framework for designing real-time CNNs. The implementations have been validated in a real-time facial expression system that provides face-detection, gender classification and that achieves human-level performance when classifying emotions. Furthermore, CNNs are used as black-boxes and often their learned features remain hidden, making it complicated to establish a balance between their classification accuracy and unnecessary parameters. Therefore, implemented a real- time visualization of the guided-gradient back-propagation proposed by Springenberg [11] in order to validate the features learned by the CNN.

Most of the face recognition (FR) approaches have focused on the use of two dimensional images. Since FR is still an unsolved problem under the different conditions, such as pose, illumination or database size. The expression of emotions and the recognition of a persons affective state are abilities indispensable for natural human interaction and social integration. The study of emotions has attracted interest of researchers from very diverse areas, ranging from psychology to the applied sciences. Face and emotion features detection is the currently very active area of research in the computer vision field as different kinds of face detection application are currently used such as image database management system, monitoring and surveillance analysis, biomedical image, smart rooms intelligent robots, human computer interfaces and drivers alertness system.

Facial recognition plays a vital rule in human computer interaction . A Face recognition system can be either verification or an identification system depending on the context of an application. The verification system authenticates a person identity by comparing the captured image with his/her own templates stored in the system. It performs a one to one comparison to determine whether the person presenting himself/herself to the system is the person he/she claims to be. An identification system recognizes a person by checking the entire template database for a match. It involves a one to many searches. The system will either make a match or subsequently identify the person or it will fail to make a match.

Voila Jones is the oldest and most recognized face algorithm available for the face detection from the image. The basic principle of the Viola-Jones algorithm is to scan a sub-window capable of detecting faces across a given input image. The standard image processing approach would be to rescale the input image to different sizes and then run the fixed size detector through these images. This approach turns out to be rather time consuming due to the calculation of the different size images. Contrary to the standard approach Viola-Jones rescale the detector instead of the input image and run the detector many times through the image each time with a different size. At first one might suspect both approaches to be equally time consuming, but Viola-Jones have devised a scale invariant detector that requires the same number of calculations whatever the size. This detector is constructed using a integral image and some simple rectangular features reminiscent of Haar wavelets.

Chapter 2

Related Works

2.1 Deep Learning using Linear Support Vector Machines

[13] Recently, fully-connected and convolutional neural networks have been trained to achieve state-of-the-art performance on a wide variety of tasks such as speech recognition, image classification, natural language processing, and bioinformatics. For classification tasks, most of these deep learning models employ the softmax activation function for prediction and minimize cross-entropy loss. Here it demonstrates a small but consistent advantage of replacing the softmax layer with a linear support vector machine. Learning minimizes a margin-based loss instead of the cross-entropy loss. While there have been various combinations of neural nets and SVMs in prior art, results using L2-SVMs show that by simply replacing softmax with linear SVMs gives significant gains on popular deep learning datasets MNIST, CIFAR-10, and the ICML 2013 Representation Learning Workshops face expression recognition challenge.Switching from softmax to SVMs is incredibly simple and appears to be useful for classification tasks. Further research is needed to explore other multiclass SVM formulations and better understand where and how much the gain is obtained.

2.2 Xception: Deep learning with depthwise separable convolution

[1] It presents an interpretation of Inception modules in convolutional neural networks as being an intermediate step in-between regular convolution and the depthwise separable convolution operation (a depthwise convolution followed by a pointwise convolution). In this light, a depthwise separable convolution can be understood as an Inception module with a maximally large number of towers. This observation leads us to propose a novel deep convolutional neural network architecture inspired by Inception, where Inception modules have been replaced with depthwise separable convolutions. It shows that this architecture, dubbed Xception, slightly outperforms Inception V3 on the ImageNet dataset (which Inception V3 was designed for), and significantly outperforms Inception V3 on a larger image classification dataset comprising 350 million images and 17,000 classes. Since the Xception architecture has the same number of parameters as Inception V3, the performance gains are not due to increased capacity but rather to a more efficient use of model parameters.Inception modules being an intermediate point in between. This observation has led to us to propose replacing Inception modules with depthwise separable convolutions in neural computer vision architectures. It presents a novel architecture based on this idea, named Xception, which has a similar parameter count as Inception V3. Compared to Inception V3, Xception shows small gains in classification performance on the ImageNet dataset and large gains on the JFT dataset. It expects depthwise separable convolutions to become a cornerstone of convolutional neural network architecture design in the future, since they offer similar properties as Inception modules, yet are as easy to use as regular convolution layers.

2.3 An Efficient Method to Face and Emotion Detection

[24] Face detection and emotion selection is the one of the current topic in the security field which provides solution to various challenges. Beside traditional challenges in captured

facial images under uncontrolled settings such as varying poses, different lighting and expressions for face recognition and different sound frequencies for emotion recognition. For the any face and emotion detection system database is the most important part for the comparison of the face features and sound Mel frequency components. For database creation features of the face are calculated and these features are store in the database. This database is then use for the evaluation of the face and emotion by using different algorithms.Here it implements an efficient method to create face and emotion feature database and then this will be used for face and emotion recognition of the person. For detecting face from the input image we are using Viola-Jones face detection algorithm and to evaluate the face and emotion detection KNN classifier is used.

For detecting the face from the image use the well known Viola Jones face detection method and for detecting voice features we used Mel frequency components of the human voice. By using KNN classifier algorithm is used to face and emotion reorganization of the person. Experimental results show the efficiency of the proposed face and emotion reorganization system is 94.5% to 97%.Most of the face recognition (FR) approaches have focused on the use of two dimensional images. Since FR is still an unsolved problem under the different conditions, such as pose, illumination or database size. The expression of emotions and the recognition of a persons affective state are abilities indispensable for natural human interaction and social integration. The study of emotions has attracted interest of researchers from very diverse areas, ranging from psychology to the applied sciences.

Face and emotion features detection is the currently very active area of research in the computer vision field as different kinds of face detection application are currently used such as image database management system, monitoring and surveillance analysis, biomedical image, smart rooms intelligent robots, human computer interfaces and drivers alertness system. Facial recognition plays a vital rule in human computer interaction. A Face recognition system can be either verification or an identification system depending on the context of an application. The verification system authenticates a person identity by comparing the captured image with his/her own templates stored in the system. It performs a one to one comparison to determine whether the person presenting himself/herself to the system is the person he/she claims to be. An identification system recognizes a person by checking the entire template database for a match. It involves a one to many searches. The system will either make a match or subsequently identify the person or it will fail to make a match.

2.4 MobileNets: Efficient CNN for Mobile Vision Applications

[2]MobileNets are based on a streamlined architecture that uses depthwise separable convolutions to build light weight deep neural networks. It introduces two simple global hyperparameters that efficiently trade off between latency and accuracy. These hyperparameters allow the model builder to choose the right sized model for their application based on the constraints of the problem. This present extensive experiments on resource and accuracy tradeoffs and show strong performance compared to other popular models on ImageNet classification. It then demonstrate the effectiveness of MobileNets across a wide range of applications and use cases including object detection, finegrain classification, face attributes and large scale geo-localization.They proposed a new model architecture called MobileNets based on depthwise separable convolutions. System investigated some of the important design decisions leading to an efficient model. System then demonstrated how to build smaller and faster MobileNets using width multiplier and resolution multiplier by trading off a reasonable amount of accuracy to reduce size and latency.Compared different MobileNets to popular models demonstrating superior size, speed and accuracy characteristics. Finally it demonstrates MobileNets effectiveness when applied to a wide variety of tasks. As a next step to help adoption and exploration of MobileNets, and plan on releasing models in Tensor Flow.

Convolutional neural networks have become ubiquitous in computer vision ever since AlexNet popularized deep convolutional neural networks by winning the ImageNet Challenge: ILSVRC 2012 . The general trend has been to make deeper and more complicated networks in order to achieve higher accuracy , these advances to improve accuracy are not necessarily making networks more efficient with respect to size and speed. In many real world applications such as robotics, self-driving car and augmented reality, the recognition tasks need to be carried out in a timely fashion on a computationally limited platform

2.5 Automatic emotion detection model from facial expression

[20]The human face plays a prodigious role for automatic recognition of emotion in the field of identification of human emotion and the interaction between human and computer for some real application like driver state surveillance, personalized learning, health monitoring etc. Most reported facial emotion recognition systems, however, are not fully considered subject-independent dynamic features, so they are not robust enough for real life recognition tasks with subject (human face) variation, head movement and illumination change. The sysytem tries to design an automated framework for emotion detection using facial expression. For human computer interaction facial expression makes a platform for non-verbal communication. The emotions are effectively changeable happenings that are evoked as a result of impelling force. So in real life application, detection of emotion is very challenging task.

Facial expression recognition system requires to overcome the human face having multiple variability such as color, orientation, expression, posture and texture so on. In this framework it takes frame from live streaming and processed it using Grabor feature extraction and neural network. To detect the emotion facial attributes extraction by principal component analysis is used and a clusterization of different facial expression with respective emotions. Finally to determine facial expressions separately, the processed feature vector is channeled through the already learned pattern classifiers.Till today all of the existing vision system for facial muscle action detection deal only with the frontal-view face images and cannot handle the temporal dynamics of facial actions. Also for some human being, it shows their emotion and mental state by facial expression, for this kind of situation the proposed model significantly fails to recognize the emotion and provides false positive result.The experimental confirmation shows that the proposed framework for automatic emotion detection can be well appertained to real time facial expression and emotion characterization task.

Emotion recognition is a two steps procedure which involves extraction of significant features and classification. Feature extraction determines a set of independent attributes, which together can portray an expression of facial emotion. For classification in emotion recognition the features are mapped into either of various emotion classes like anger, happy, sad, disgust, surprise, etc . For the effectiveness calculation of a facial expression identification model both the group of feature attributes which have been taken for feature extraction and the classifier that is responsible classification are equivalently significant. For a badly picked collection of feature attributes, in some cases, even a smart classification mechanism is not able to produce an ideal outcome. Thus, for getting the high classification accuracy and qualitative outcome, picking of superior features will play a major role.

The circumflex model by Russell and recognition of six basic emotions are having remarkable contribution in the field of emotion recognition. Other than this, the work by Kudiri M. Krishna, Said Abas Md, Nayan M Yunus , where they tried to detect emotion by using the concept of sub-image based features through facial expression. Silva C. DE Liyanage, Miyasato Tsutomu, Nakatsu Ryohei, they formed a model for emotion recognition with the help of multimodal information. Maja Pantic, Iounnis Patras, they implemented an approach for recognition facial action and temporal segment from fare profile image sequences by considering the dynamic property of facial action. Li Zhang, Ming Jiang, Dewan Farid, M.A. Hassain , they modelled an intelligent system for automatic emotion recognition. Happy S L, Routray Aurobinda , created an automatic emotion recognition system by using salient features. This article is greatly influenced by all of this contribution in the field of emotion recognition.

2.6 Deep Residual Learning for Image Recognition

[6] Deeper neural networks are more difficult to train. The system presents a residual learning framework to ease the training of networks that are substantially deeper than

those used previously. It explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. It provides comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers deeper than VGG nets but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. It also present analysis on CIFAR-10 with 100 and 1000 layers. The depth of representations is of central importance for many visual recognition tasks. Solely due to extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset.

In image recognition, VLAD is a representation that encodes by the residual vectors with respect to a dictionary, and Fisher Vector can be formulated as a probabilistic version of VLAD. Both of them are powerful shallow representations for image retrieval and classification . For vector quantization, encoding residual vectors is shown to be more effective than encoding original vectors. In low-level vision and computer graphics, for solving Partial Differential Equations (PDEs), the widely used Multigrid method reformulates the system as subproblems at multiple scales, where each subproblem is responsible for the residual solution between a coarser and a finer scale. An alternative to Multigrid is hierarchical basis preconditioning which relies on variables that represent residual vectors between two scales. It has been shown that these solvers converge much faster than standard solvers that are unaware of the residual nature of the solutions. These methods suggest that a good reformulation or preconditioning can simplify the optimization.

2.7 DEX: Deep Expectation of apparent age from a single image

[9]The convolutional neural networks (CNNs) use the VGG-16 architecture and are pretrained on ImageNet for image classification. In addition, due to the limited number of apparent age annotated images, it explore the benefit of finetuning over crawled Internet

face images with available age. We crawled 0.5 million images of celebrities from IMDB and Wikipedia that we make public. This is the largest public dataset for age prediction to date. it poses the age regression problem as a deep classification problem followed by a softmax expected value refinement and show improvements over direct regression training of CNNs. The proposed method, Deep EXpectation (DEX) of apparent age, first detects the face in the test image and then extracts the CNN predictions from an ensemble of 20 networks on the cropped face. The CNNs of DEX were finetuned on the crawled images and then on the provided images with apparent age annotations. DEX does not use explicit facial landmarks.The convolutional neural networks (CNNs) use the VGG-16 architecture and are pretrained on ImageNet for image classification. In this way we benefit from the representation learned to discriminate object categories from images. This representation is not capable of good age estimation. Finetuning the CNN on training images with apparent age annotations is a necessary step to benefit from the representation power of the CNN. The 524,230 face images crawled from IMDB and Wikipedia websites form new dataset, the IMDB-WIKI dataset.

2.8 Deep Sparse Rectifier Neural Networks

[5]While logistic sigmoid neurons are more biologically plausible than hyperbolic tangent neurons, the latter work better for training multi-layer neural networks. This shows that rectifying neurons are an even better model of biological neurons and yield equal or better performance than hyperbolic tangent networks in spite of the hard non-linearity and non-differentiability at zero, creating sparse representations with true zeros, which seem remarkably suitable for naturally sparse data. Even though they can take advantage of semi-supervised setups with extra-unlabeled data, deep rectifier networks can reach their best performance without requiring any unsupervised pre-training on purely supervised tasks with large labeled datasets. Hence, these results can be seen as a new milestone in the attempts at understanding the difficulty in training deep but purely supervised neural networks, and closing the performance gap between neural networks learnt with and without unsupervised pre-training.Sparsity and neurons operating mostly in a linear

regime can be brought together in more biologically plausible deep neural networks. Rectifier units help to bridge the gap between unsupervised pre-training and no pre-training, which suggests that they may help in finding better minima during training. This finding has been verified for four image classification datasets of different scales and all this in spite of their inherent problems, such as zeros in the gradient, or ill-conditioning of the parametrization. Rather sparse networks are obtained (from 50 to 80the best generalizing models, whereas the brain is hypothesized to have 95% to 99% sparsity), which may explain some of the benefit of using rectifiers.

While logistic sigmoid neurons are more biologically plausible than hyperbolic tangent neurons, the latter work better for training multi-layer neural networks.it shows that rectifying neurons are an even better model of biological neurons and yield equal or better performance than hyperbolic tangent networks in spite of the hard non-linearity and non-differentiability at zero, creating sparse representations with true zeros, which seem remarkably suitable for naturally sparse data. Even though they can take advantage of semi-supervised setups with extra-unlabeled data, deep rectifier networks can reach their best performance without requiring any unsupervised pre-training on purely supervised tasks with large labeled datasets. Hence, these results can be seen as a new milestone in the attempts at understanding the difficulty in training deep but purely supervised neural networks, and closing the performance gap between neural networks learnt with and without unsupervised pre-training.

Chapter 3

System Analysis

3.1 Proof of concept

This proposed system has a concise description of the theoretical background needed to understand the project in depth. In this project, the use of CNN techniques for the task of Emotion Detection. CNN can be used to implement image properties related projects

3.1.1 Need of proof of concept

3.1.1.1 Convolutional Neural Network

In machine learning, a convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field.

3.1.1.2 Why CNN?

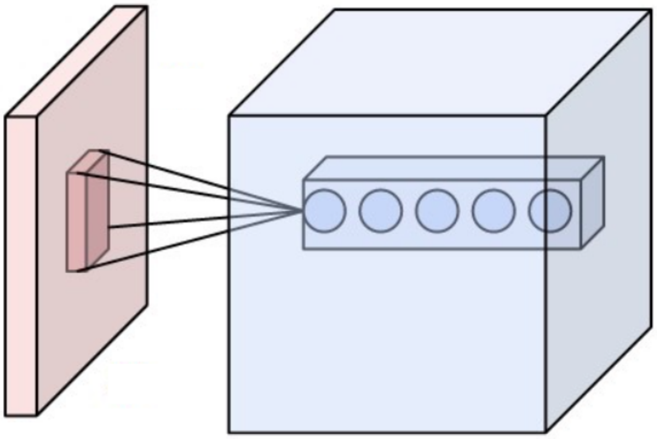


Figure 3.1. Convolutional layer and their receptive field

While traditional multilayer perceptron (MLP) models were successfully used for image recognition, due to the full connectivity between nodes they suffer from the curse of dimensionality, and thus do not scale well to higher resolution images.

Also, such network architecture does not take into account the spatial structure of data, treating input pixels which are far apart in the same way as pixels that are close together. Thus, full connectivity of neurons is wasteful for purposes such as image recognition that are dominated by spatially local input patterns.

As opposed to MLPs, CNNs have the following distinguishing features:

3D volumes of neurons. The layers of a CNN have neurons arranged in 3 dimensions: width, height and depth. The neurons inside a layer are connected to only a small region of the layer before it, called a receptive field . Distinct types of layers, both locally and completely connected, are stacked to form a CNN architecture. Local connectivity: following the concept of receptive fields, CNNs exploit spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers. The architecture thus ensures that the learnt "filters" produce the strongest response to a spatially local input pattern. Stacking many such layers leads to non-linear filters that become increasingly global (i.e. responsive to a larger region of pixel space) so that the network first creates representations of small parts of the input, then from them assembles representations of larger areas. Shared weights: In CNNs,as shown in figure 3.1, each filter is replicated across the entire visual field. These replicated units share the same parameterization (weight vector and bias) and form a feature map. This means that all the neurons in a given convolutional layer respond to the same feature within their specific response field. Replicating units in this way allows for features to be detected regardless of their position in the visual field, thus constituting the property of translation invariance. Together, these properties allow CNNs to achieve better generalization on vision problems. Weight sharing dramatically reduces the number of free parameters learned, thus lowering the memory requirements for running the network and allowing the training of larger, more powerful networks.

3.1.2 Design of proof of concept

Facial expressions play a very important role in human communication. The human face is the richest source of emotions. As society continues to make more use of human-machine interactions, it is important for machines to be able to interpret facial expressions in order to improve their authenticity or to make them less machine and more human. Our brains make vision seem easy. It doesn't take any effort for humans to tell apart a lion and a tiger, read an article, or recognizing a human's face. But these are actually hard problems to solve with a computer: they only seem easy because our brains are incredibly good at understanding images. There are seven types of human emotions shown to be universally recognizable across different cultures: anger, disgust, fear, happiness, sadness, surprise, contempt. Interestingly, even for complex expressions where a mixture of emotions could be used as descriptors, cross-cultural agreement is still observed. Therefore, a utility that detects emotion from facial expressions would be widely applicable. Such advancement could bring applications in medicine, marketing and entertainment. The task of emotion recognition is particularly difficult for two reasons: There does not exist a large database of training images and classifying emotions can be difficult depending on whether the input image is static or in a transition frame into a facial expression. The latter issue is particularly difficult for real-time detection where facial expressions vary dynamically. We propose and implement a general convolutional neural network (CNN) building framework for designing real-time CNNs. it validate our models by creating a real-time vision system which accomplishes the tasks of face detection, gender classification and emotion classification simultaneously in one blended step using our proposed CNN architecture. After presenting the details of the training procedure setup it proceed to evaluate on standard benchmark sets. It report accuracies of 66% in the FER-2013 emotion dataset. Along with this it also introduced the very recent real-time enabled guided back propagation visualization technique. Guided back-propagation uncovers the dynamics of the weight changes and evaluates the learned features. This argue that the careful implementation of modern CNN architectures, the use of the current regularization methods and the visualization of previously hidden features are necessary in order to reduce the gap between slow performances and real-time architectures.

3.1.3 Study of existing system

3.1.3.1 Face Detection using Haar Cascades

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other imagesf

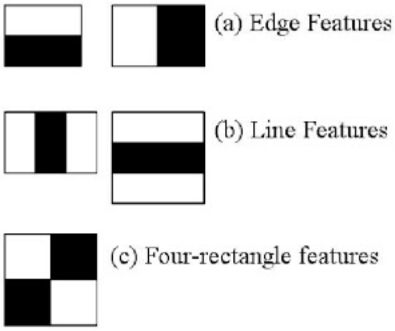


Figure 3.2. Haar Feature

Here it will work with face detection. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier.

Then we need to extract features from it. For this, Haar features shown in the figure 3.2 are used. They are just like our convolutional kernel. Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle.

This apply each and every feature on all the training images. For each feature, it finds the best threshold which will classify the faces to positive and negative. Obviously, there will be errors or misclassifications. We select the features with minimum error rate, which means they are the features that most accurately classify the face and non-face images. (The process is not as simple as this. Each image is given an equal weight in the beginning. After each classification, weights of misclassified images are increased. Then the same process is done. New error rates are calculated. Also new weights. The process is continued until the required accuracy or error rate is achieved or the required number of features are found).

The final classifier is a weighted sum of these weak classifiers. It is called weak because it alone can't classify the image, but together with others forms a strong classifier. The paper says even 200 features provide detection with 95% accuracy. Their final setup had around 6000 features.

3.1.4 Study of proposed system

Commonly used CNNs for feature extraction include a set of fully connected layers at the end. Fully connected layers tend to contain most of the parameters in a CNN. Specifically, VGG16 contains approximately 90% of all its parameters in their last fully connected layers. Recent architectures such as Inception V3 , reduced the amount of parameters in their last layers by including a Global Average Pooling operation. Global Average Pooling reduces each feature map into a scalar value by taking the average over all elements in the feature map. The average operation forces the network to extract global features from the input image. Modern CNN architectures such as Xception leverage from the combination of two of the most successful experimental assumptions in CNNs: the use of residual modules and depth-wise separable convolutions . Depth-wise separable convolutions reduce further the amount of parameters by separating the processes of feature extraction and combination within a convolutional layer.

Furthermore, the state-of-the-art model for the FER2-2013 dataset is based on CNN trained with square hinged loss . This model achieved an accuracy of 71% using approximately 5 million parameters. In this architecture 98% of all parameters are located in the last fully connected layers. The second-best methods presented in achieved an accuracy of 66% using an ensemble of CNN

3.1.5 Recommended system requirements

- **Dataset:**A massive collection of images taken form different camera models

- **CNN Classifier:**A convolutional neural network (CNN, or ConvNet) is a class of deep, feed-forward artificial neural networks that has successfully been applied to analyzing visual imagery.

  CNNs use a variation of multilayer perceptrons designed to require minimal preprocessing.They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics.

- **Keras:**Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or MXNet.Keras contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel.

3.1.6 Hardware And Software Requirements

3.1.6.1 Hardware Requirement

The hardware mostly used in this project are given below

- No of personal computers-1

- Processor-Pentium-4 or above
- Main Memory-512 MB or above
- Hard disk-60 GB or above
- Base memory-1 GB
- Keyboard-101 keys

3.1.6.2 Software Requirement

- **Operating Systems - Any Linux OS**

  Linux is a Unix-like computer operating system assembled under the model of free and open source software development and distribution. The defining component of Linux is the Linux, an operating system kernel first released October 5, 1991 by Linus Torvalds. Linux was originally developed as a free operating system for Intel x86 based personal computers. It has since been ported to more computer hardware platforms than any other operating system. It is a leading operating system on servers and other big iron systems such as mainframe computers and supercomputers. More than 90 percentage of todays top 500 supercomputers run some variant of Linux, including the 10 fastest. Linux also runs on embedded systems (devices where the operating system is typically built into the firmware and highly tailored to the system) such as mobile phones, tablet computers, network routers , televisions and video game consoles; the Android system in wide use on mobile devices is built on the Linux kernel. The development of Linux is one of the most prominent examples of free and open source. The primary difference between Linux and many other popular contemporary operating systems is that the Linux kernel and other components are free and open source software. Linux is not the only such operating system, although it is by far the most widely used. Some free and open source software licenses are based on the principle of copyleft, a kind of reciprocity: any work derived from a copyleft piece of software must also be copyleft itself. The most common free software license, the GNUGPL

- **Libraries Packages - Keras**

  Keras is an open source neural network library written in Python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, Theano, or MXNet. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is FranÃğois Chollet, a Google engineer

- **Tensorflow**

  TensorFlow is an open source software library for high performance numerical computation. Its flexible architecture allows easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. Originally developed by researchers and engineers from the Google Brain team within Google AI organization, it comes with strong support for machine learning and deep learning and the flexible numerical computation core is used across many other scientific domains.

- **Pandas**

  Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

- **Numpy**

  NumPy is the fundamental package for scientific computing with Python. It contains among other things: a powerful N-dimensional array object, sophisticated (broadcasting) functions,tools for integrating C/C++ and Fortran code and useful linear algebra, Fourier transform, and random number capabilities

- **Opencv2-Python**

  OpenCV-Python is a library of Python bindings designed to solve computer vision problems.Python is a general purpose programming language started by Guido van Rossum that became very popular very quickly, mainly because of its simplicity and code readability. It enables the programmer to express ideas in fewer lines of code without reducing readability.OpenCV now supports a multitude of algorithms related to Computer Vision and Machine Learning and is expanding day by day.

  OpenCV supports a wide variety of programming languages such as C++, Python, Java, etc., and is available on different platforms including Windows, Linux, OS X, Android, and iOS. Interfaces for high-speed GPU operations based on CUDA and OpenCL are also under active development.OpenCV-Python is the Python API for OpenCV, combining the best qualities of the OpenCV C++ API and the Python language.

  OpenCV-Python makes use of Numpy, which is a highly optimized library for numerical operations with a MATLAB-style syntax. All the OpenCV array structures are converted to and from Numpy arrays. This also makes it easier to integrate with other libraries that use Numpy such as SciPy and Matplotlib.

- **Python**

  The primary impetus for Python was to make a language that was powerful and capable and at the same time very easy to use and learn.Guido named Python after the British comedy program Monty Pythons Flying Circus. Perhaps this name was chosen to communicate the groundbreaking notion that programming should be fun and playful. Python is one of those rare languages which can claim to be both simple and powerful.Python features a dynamic type system and automatic memory management and supports multiple programming paradigms, including object-oriented, imperative, functional programming, and procedural styles. It has a large and comprehensive standard

Chapter 4

Design

4.1 Introduction

4.1.1 Elements of design

The components of an information system described during requirement analysis are the focal point in system design. Analysis must design the following elements.

- Data Flow
- Data Store
- Processes
- Procedures
- Controls
- Roles

Data flow analysis permits to isolate areas of interest in the organization and to study them by examining the data that enter the process and seeing how they change when they leave the process. In the data flow diagrams, the physical system is translated into a logical description that focuses on data and processes. It is advantageous to emphasize data and process in order to focus on the actual activities that occur and the resources needed to perform them, rather than on who performs the work.

4.1.2 Logical Design

Logical design is an abstract representation of inputs, outputs, databases and procedures of the system. A Data Flow Diagram is a graphical representation of the flow of data through an information system. A data flow diagram can also be used for the visualization of data processing. The idea behind the project is to detect various emotions from the face.

4.1.3 Physical Design

The physical design relates to the actual input and output processes of the system.This is explained in terms of how data is input into a system, how it is verified or authenticated, how it is processed, and how it is displayed. In physical design, the following requirements about the system are decided.

- **Input requirement:** The system requires a source image
- **Output requirement:** The system does the processing and detect the emotion from the face and shown to the user

4.1.4 Code Design

- Create a python program
- Link it with the dataset of images
- Create CNN using Keras fucntions
- Give these images as input to the input layer
- Generate the CNN model by training and save the model to a file
- Load this model file
- Open the camera and start capturing frame by frame
- Crop the face from the frame
- Find the face co-ordinate

- Compare face with the model
- Predict the emotion of the user

4.2 Design of Modules

4.2.1 Mini-Xception model

The second model is inspired by the Xception architecture. This architecture combines the use of residual modules and depth-wise separable convolutions. Residual modules modify the desired mapping between two subsequent layers, so that the learned features become the difference of the original feature map and the desired features. Consequently, the desired features H(x) are modified in order to solve an easier learning problem F (X) as shown in eqn (4.1)

$$H(x) = F(x) + x \tag{4.1}$$

Since initial proposed architecture deleted the last fully connected layer, we reduced further the amount of parameters by eliminating them now from the convolutional layers. This was done trough the use of depth-wise separable con volutions. Depth-wise separable convolutions are composed of two different layers: depth-wise convolutions and point-wise convolutions. The main purpose of these layers is to separate the spatial cross-correlations from the channel cross- correlations[1] . Depth-wise separable convolutions reduces the computation with respect to the standard convolutions by a factor of $\frac{1}{N} + \frac{1}{D^2}$ A visualization of the difference between a normal Convolution layer and a depth-wise separable convolution can be observed as shown in the figure 4.1 The final architecture is a fully-convolutional neural network that contains 4 residual depth-wise separable convolutions as shown in figure 4.2 where each convolution is followed by a batch normalization operation and a ReLU activation function. The last layer applies a global average pooling and a soft-max activation function to produce a prediction. This architecture has approximately 60, 000 parameters; which corresponds to a reduction of 10X wen compared to our initial naive implementation, and 80X when compared to the original CNN. Displays our complete final architecture which we refer to as mini-Xception.

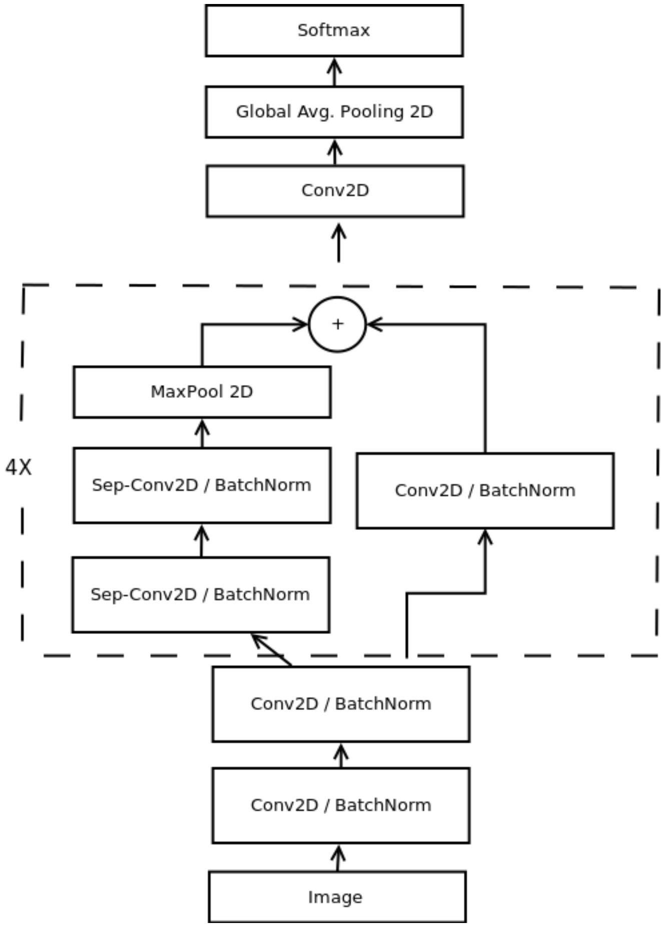


Figure 4.1. Proposed model for real-time classification

Here system tested this architecture in the FER-2013 dataset and we obtained the same accuracy of 66% for the emotion classification task. The final architecture weights can be stored in an 855 kilobytes file. By reducing our architectures computational cost it is able to join both models and use them consecutively in the same image without any serious time reduction. The complete pipeline including the openCV face detection module, the gender classification and the emotion classification takes 0.22 + 0.0003 ms on a i5-4210M CPU. This corresponds to a speedup of 1.5X when compared to the original architecture of Tang. We also added to our implementation a real-time guided back-propagation visualization to observe which pixels in the image activate an element of a higher-level feature map. Given a CNN with only ReLUs as activation functions for the intermediate layers, guided-back propagation takes the derivative of every element $(x, y)$of

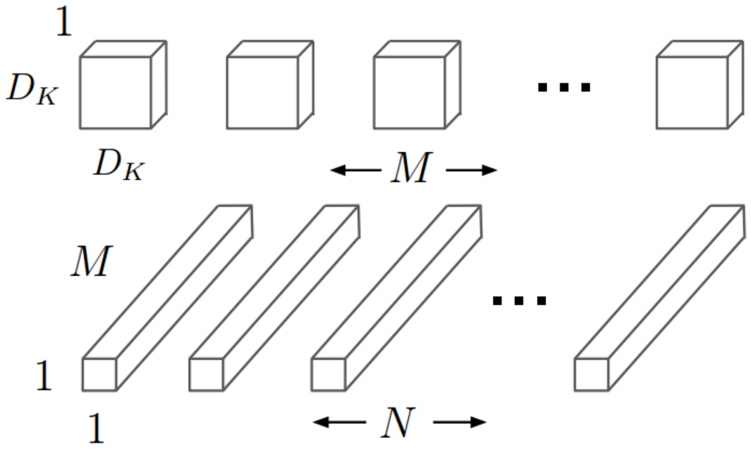


Figure 4.2. Standard convolutions vs Depth-wise separable convolutions

the input image I with respect to an element (i, j) of the feature map f L in layer L. The reconstructed image R filters all the negative gradients; consequently, the remaining gradients are chosen such that they only increase the value of the chosen element of the feature map. .

4.3 Data Flow Diagram

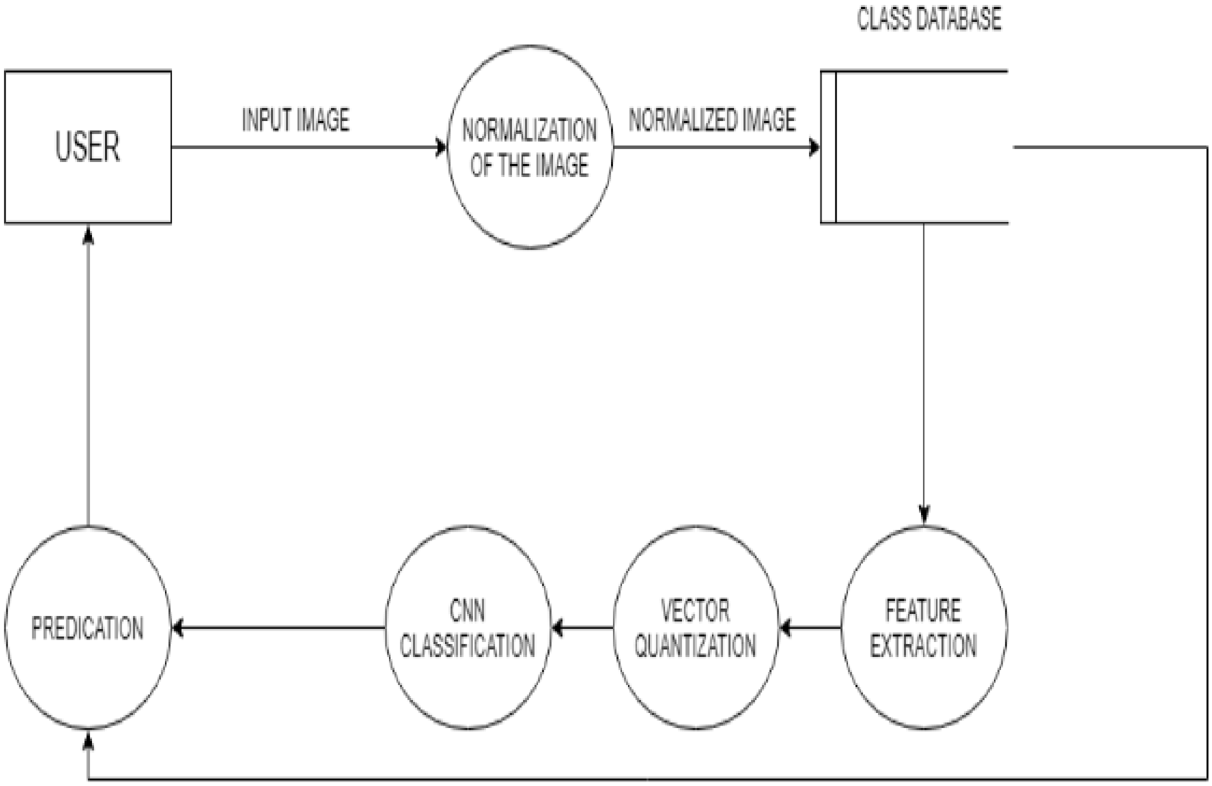


Figure 4.3. Data Flow Diagram

4.4 FlowChart

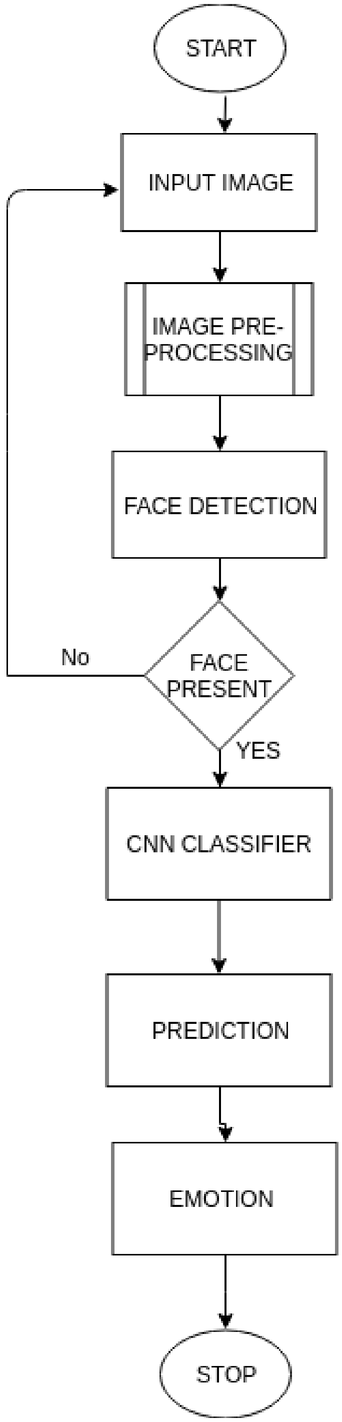


Figure 4.4. Flowchart

Chapter 5

Datasets

5.1 Facial Expression Recognition

The data consists of 48x48 pixel grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. The task is to categorize each face based on the emotion shown in the facial expression in to one of seven categories (0=Angry, 1=Disgust, 2=Fear, 3=Happy, 4=Sad, 5=Surprise, 6=Neutral).

train.csv contains two columns, "emotion" and "pixels". The "emotion" column contains a numeric code ranging from 0 to 6, inclusive, for the emotion that is present in the image. The "pixels" column contains a string surrounded in quotes for each image. The contents of this string a space-separated pixel values in row major order. test.csv contains only the "pixels" column and your task is to predict the emotion column.The format of the CSV file is as follows: usage, neutral, happiness, surprise, sadness, anger, disgust, fear, contempt, unknown, NF. Columns "usage" is the same as the original FER label to differentiate between training, public test and private test sets. The other columns are the vote count for each emotion with the addition of unknown and NF (Not a Face).

The training set consists of 28,709 examples. The public test set used for the leaderboard consists of 3,589 examples. The final test set, which was used to determine the winner of the competition, consists of another 3,589 examples.

This dataset was prepared by Pierre-Luc Carrier and Aaron Courville, as part of an

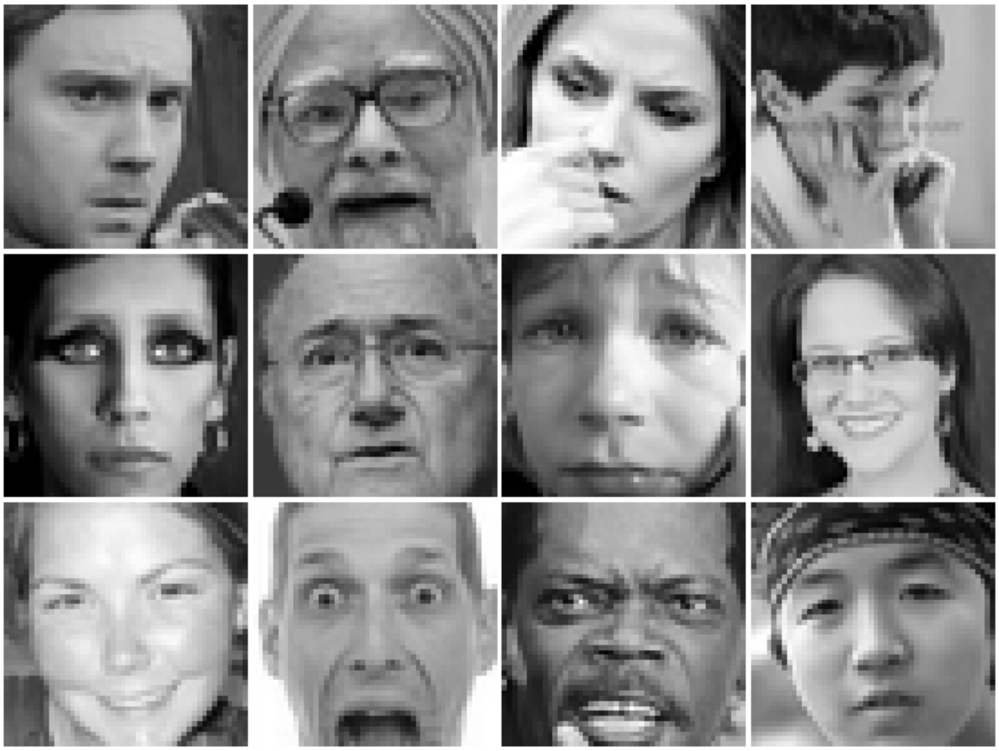


Figure 5.1. FER 2013 DataBase

ongoing research project. They have graciously provided the workshop organizers with a preliminary version of their dataset to use for this contest.

Chapter 6

Implementation

6.1 Running Procedure

The Emotion Prediction mainly consists of two important steps:

- **Training** :We used Fer-2013 as the database for classification of the images.These data were pre-prossessed and trained then validated.
- **Testing** :The Python program is executed and it captures the image from the Webcam and it will predict the Emotion.The system is connected the CNN model for the prediction purpose.

6.2 Implementation Plan

6.2.1 Frame Extraction and Face Detection

Initially the system have taken live video stream and extracts frames from the video. Then it tried to detect those frames that are having face. To detect face in a frame have used an existing well-known technique where for feature extraction Gabor method is used and for learning of neural network is used, and combinedly they are used for face detection[14] [15] [16].Send frame for further processing where the model is already learned for emotion detection. The feed forward architecture of Multi-Layer Perceptron (MLP) neural network [19][20][21]consists of three layer- input layer, a hidden layer, and an output layer as shown in fig 6.1. For an N dimensional input vector there exists N

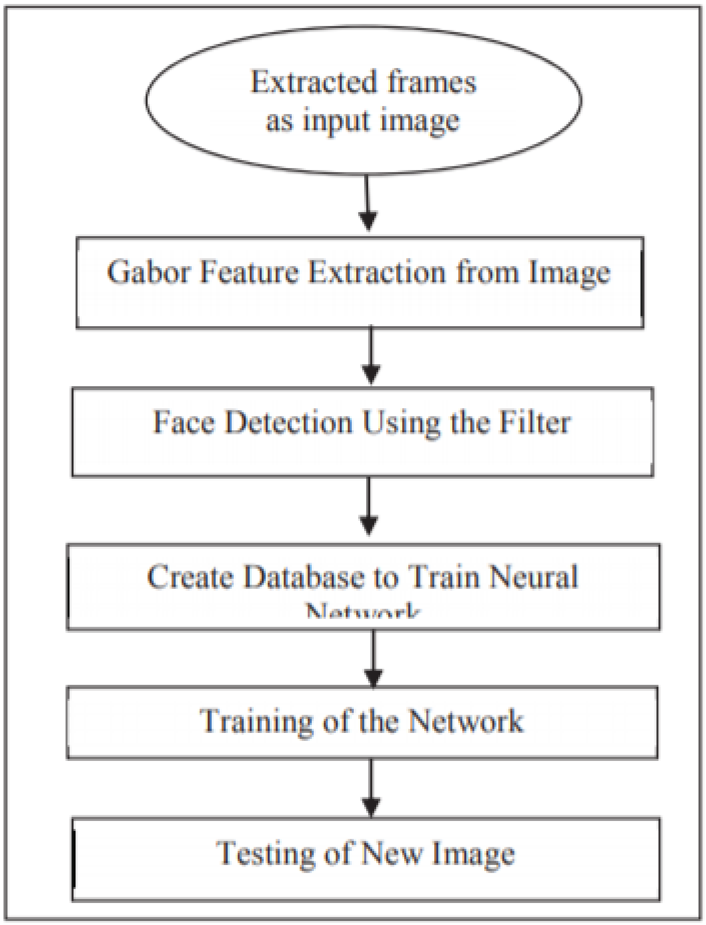

Extracted frames as input image
Gabor Feature Extraction from Image
Face Detection Using the Filter
Create Database to Train Neural Network
Training of the Network
Testing of New Image

Figure 6.1. Generalized Model for Face Detection

units in the input layer. The input units are fully cascaded to the I hidden layer units, which are in turn, connected to the J output layers units, where J is the number of output classes. A training data of 1 pairs (xi, yi) is assumed to be accessed where xi is the pattern vector, while yi is the corresponding pattern class.Can code yi as 1 and -1 in a 2-class task. MLP comprises of 3 layers, the input layer is a vector constituted having n^2 units of neurons (n x n pixel input images). The hidden layer contains n neurons, and the output layer contains a single neuron that is active to 1 when the face is presented and otherwise face is not presented.

6.2.2 Learning and Principal Component Analysis

In learning phase, firstly create clusters with respective emotion using PCA [18]. Now whenever a new frame will come its PCA will be generated we have compared that PCA in test phase. Here trained our system with different facial expression. Principal component analysis is a technique mainly used for dimensionality reduction, lossy data compression, and feature extraction.PCA can be described by orthogonally projecting the data onto a lower dimensional linear space called as principal subspace in order to maximize the variance of projected data [18].For maximum variance formulation consider a data set of observation xn where $n = 1, 2, 3, ..N$, and xn is a Euclidean value with dimensionality D. need to project the data onto a space having dimensionality M < D while maximizing the variance of the projected data. The value of M is assumed. For the projection have only considered the space having one dimension (M = 1). The direction of this space is defined by a vector u1 dimension D, then have considered a unit vector so that u1 $u1 = 1$. Each data point xn is then projected onto a scalar value u1 xn.The mean of the projected data is where x is the sample set mean given by eqn 6.1

$$X = \frac{1}{N}\sum_{n}^{N} x_n \tag{6.1}$$

6.2.3 Testing and Pattern Classification

Then during the test phase It has been taken any random image from the dataset and use pattern classification to find the emotion of that particular image. For pattern classification It has been used very well-known K-mean clustering with some modified approach. Here, instead of making clusters for each of the individual emotion by creating cluster centre for each of the emotion, we use their principal component as it is. Because for a same human being for similar emotion the output of principal component may vary up to certain extent. So for testing the emotion It has been considered two measure simultaneously

- Overlapping measure
- Minimum distance measure.

In overlapping measure It has been tried to find amount of overlapping of principle component of the test image with the principle component of respective trained emotion, if overlapping occurs. In minimum distance measure, It has been used the Kmean approach.

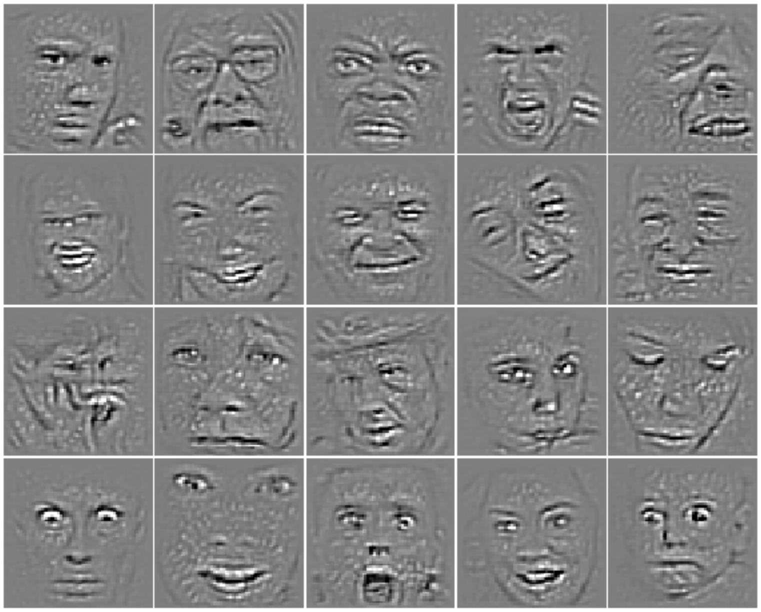


Figure 6.2. Visualization of our mini-Xception model

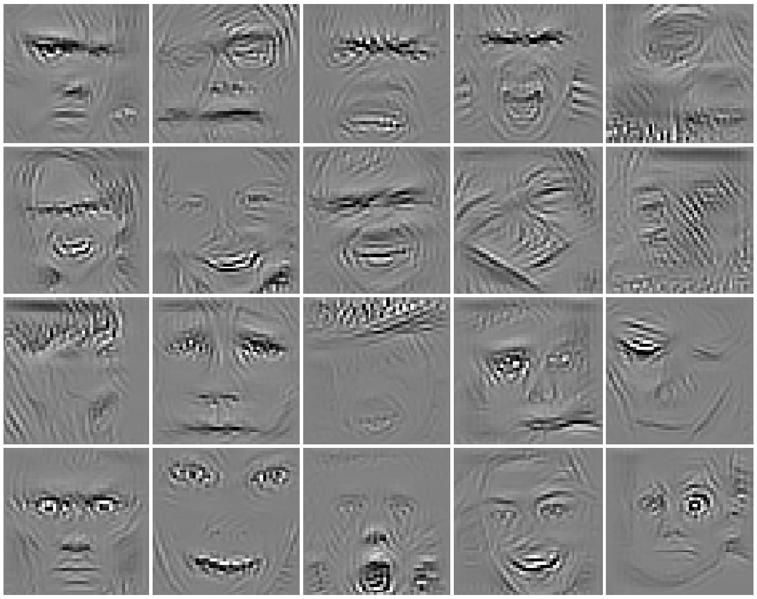


Figure 6.3. Visualization of our sequential fully CNN.

For any principal component of the test image It has been measured the distance with the principle component of different or same emotion and considered only the minimum one. Finally comparing these two result we are able to find the emotion of the test image.

6.3 Area of implementation

Robots that communicate with human have attracted much attention in the research field of robotics. In communication between human, almost all human recognize the subtleties of emotion in each other's facial expressions, voices, and motions. Robots can communicate more smoothly with human as they detect human emotions and respond with appropriate behaviors. Usually, almost all human express their own emotions with their facial expressions. In this paper, we propose an emotion detection system with facial features using a Bayesian network. In actual communication, it is possible that some parts of the face will be occluded by adornments such as glasses or a hat. In previous studies on facial recognition, these studies have been had the process to fill in the gaps of occluded features after capturing facial features from each image. However, not all occluded features can always be filled in the gaps accurately. Therefore, it is difficult for robots to detect emotions accurately in real-time communication. For this reason, we propose an emotion detection system taking into consideration partial occlusion of the face using causal relations between facial features.

Video games belong to the wide area of entertainment applications. Thus, assuming the existence of human emotions and in fact basing on them, they attempt to make the player to become emotionally attached with them. As the primary goal of a video game is to entertain the player, each video game try to allow the player to fulfill his or her dream. Standard video games try to do it in different ways depending on their genre and involving such elements as good gameplay, immersing storytelling, novelty, graphics and so on. Although video games belong to applications in which emotions naturally play an important role, only few of them try to incorporate their players affective state into the gameplay. Such games can be referred as affective or more properly affect-aware games. The importance of affect in delivering engaging experiences in entertainment and educational games is well recognized. Potential profits for affect-aware video games are not to be underestimated. Unfortunately, this affect-awareness is usually statically built-in the game at its development stage basing on the assumed model of so called representative player. There are two problems with such attitude. Firstly, each player differs in some way from that averaged model. Secondly, and more important, player's affect state can change even radically from session to session making almost impossible to predict the current user emotions at the development stage.

Chapter 7

Results

Results of the real-time emotion classification task in un- seen faces.Our complete real-time pipeline including: face detection and emotion detection have been fully integrated Provide the confusion matrix results of our emotion classification mini-Xception model.We can observe same common mis-classifications such as predicting sad instead of fear and predicting angry instead disgust.

Its has been observed that the features learned in our mini-Xception model are more interpret able than the ones learned from our sequential fully-CNN. Consequently the use of more parameters in our naive implementations leads to less robust features.

	Anger	**Sad**	**Happy**	**Neutral**	**Disgust**	**Fear**	**Surprise**
Anger	0.60	0.10	0.05	0.05	0.03	0.07	0.10
Sad	0.11	0.65	0.02	0.09	0.02	0.03	0.08
Happy	0.04	0.02	0.75	0.09	0.00	0.06	0.04
Neutral	0.15	0.08	0.05	0.63	0.02	0.05	0.02
Disgust	0.04	0.07	0.08	0.03	0.61	0.09	0.08
Fear	0.04	0.08	0.04	0.04	0.01	0.72	0.07
Surprise	0.04	0.06	0.05	0.08	0.06	0.04	0.67

Table 7.1. Confusion matrix of the emotion recognition system based on audio

Table 7.1 shows the confusion matrix of the emotion recognition system based on acoustic information, which gives details of the strengths and weaknesses of this system. The overall performance of this classifier was 70.9 percent. The diagonal components of table 7.1 reveal that all the emotions can be recognized with more than 64 percent of accuracy, by using only the features of the speech.However, table 1 shows that some pairs of emotions are usually confused more. Sadness is misclassified as neutral state (9%) and vise versa (9%). The same trend appears between happiness and anger, which are mutually confused (1% and 8%, respectively).

	Anger	**Sad**	**Happy**	**Neutral**	**Disgust**	**Fear**	**Surprise**
Anger	0.70	0.08	0.05	0.06	0.05	0.04	0.02
Sad	0.08	0.75	0.05	0.09	0.00	0.00	0.03
Happy	0.01	0.03	0.90	0.02	0.00	0.00	0.04
Neutral	0.03	0.09	0.04	0.75	0.09	0.00	0.00
Disgust	0.06	0.00	0.03	0.03	0.79	0.09	0.00
Fear	0.07	0.02	0.00	0.00	0.00	0.80	0.11
Surprise	0.05	0.09	0.03	0.06	0.00	0.01	0.76

Table 7.2. Confusion matrix of the facial expression classifier

Table 7.2 shows the confusion matrix of the facial expression classifier to analyze in detail the limitation of this emotion recognition system. The overall performance of this classifier was 85.1 percent. This table reveals that happiness is recognized with very high accuracy. The other three emotions are classified with 70 percent of accuracy, approximately. Table 7.2 also shows that in the facial expressions domain, anger is confused with sadness (8%) and neutral state is confused with happiness (4%). Notice that in the acoustic domain, sadness/anger and neutral /happiness can be separated with high accuracy, so it is expected that the bi modal classifier will give good performance for anger and neutral state. This table also shows that sadness is confused with neutral state (3%).However from the above two tables it has been understood that accuracy of facial expression classifier using CNN is much more than emotion recognition system based on audio.

Chapter 8

Conclusions And Future Works

We have proposed and tested a general building designs for creating real-time CNNs. Our proposed architectures have been systematically built in order to reduce the amount of parameters. We began by eliminating completely the fully connected layers and by reducing the amount of parameters in the remaining convolutional layers via depth-wise separable convolutions.The proposed models can be stacked for multi-class classifications while maintaining real-time inferences. Specifically, we have developed a vision system that performs face detection and emotion classification in a single integrated module.We had obtained a good accuracy in predicting the emotions.We have achieved human-level performance in our classifications tasks using a single CNN that leverages modern architecture constructs.

As increasing number of hidden layers resulted in a slight improvement in classifi- cation, but further increase in hidden layers however deteriorated the results.

References

[1] Francois Chollet. Xception: Deep learning with depth wise separable convolutions. CoRR, abs/1610.02357, 2016.

[2] Andrew G. Howard et al. Mobile nets: Efficient convolution neural networks for mobile vision applications. CoRR, abs/1704.04861, 2017.

[3] Dario Amodei et al. Deep speech 2: End-to-end speech recognition in English and mandarin. Corr, abs/1512.02595, 2015.

[4] Ian Good fellow et al. Challenges in Representation Learning: A report on three machine learning contests, 2013.

[5] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, pages 315 to 323, 2011.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770 to 778, 2016.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Interna- tional Conference on Machine Learning, pages 448to 465, 2015.

[8] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[9] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. International Journal of Computer Vision (IJCV), July 2016.

[10] Karen Simonyan and Andrew Zisserman. Very deep convolu- tional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[11] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. arXiv preprint arXiv:1412.6806, 2014.

[12] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 2818 to 2826, 2016.

[13] Yichuan Tang. Deep learnig using linear support vector machines. arXiv preprint arXiv:1306.0239, 2013.

[14] D. Anurag,S. Ashim, A Comparative Study on different approaches of Real Time Human Emotion Recognition based on Facial Expression Detection, International Conference on Advances in Computer Engineering and Applications, 4/15/31.002015 IEEE.

[15] Kudiri M. Krishna, Said Abas Md, Nayan M Yunus, Emotion Detection Using Subimage Based Features Through Human Facial Expressions, International on Computer Information Science (ICCIS). 9781467319386/12/31.002012 IEEE.

[16] Pal Pritam, Iyer Ananth N. and Yantorno Robert E . Emotion Detection From Infant Facial Expression, Internationa Conference on Acoustics, Speech and Signal Processing.X/06/20.002016 IEEE

[17] Silva C. DE Lianage,Miyasato Tsutomu, Nakatsu Ryohei. Facial Emotion Recognition Using Multi-modal Information, International Conference on Information, Communication and Signal Processing, 3/97/10.001997 IEEE.

[18] Maja Pantic, Iounnis Patras. Dynamics of Facial Expression : Recognition of Facial Actions and their Temoporal Segments from fave profile Image Sequences, IEEE Transctions on System, Man and Cybernetis.10834419/20.002006 IEEE

[19] Songfan Yang, Bir Bhanu, Understanding Discrete Facial expressions in Video Using an Emotion Avatar Image, IEEE Tranjactins on Systems, Man and Cybernetis. 2012 IEEE

[20] Happy S L, Routray Aurobinda, Autamatic facial Expression Recognition Using Features Salient Facial Patterns, IEEE Tranjactions on Affective Computing.3045 2014 IEEE.

[21] Mohammad Soleymani , Sadjad Asghari Esfeden, Yunfu ,Maja Pantic, Analysis of EEG signals and facial Expressions for Continuous Emotion detection, IEEE Transctions on Affective Computing, 1949 3045 2015 IEEE.

[22] Leh Luoh, Chih- Chang Huang, Hsueh-Yenhiu. Image Processing Based Emotion Recognition, International Conference on System Science and Engineering. 64746/10/26.00 2010

[23] F. Abdat, C. Maaoui and A. Pruski, UKSIM 5th European Symposium on Computer Modeling and Simulation.

[24] Chung, A.J. [Adrian J.] Deligianni F. [Fani] Hu, X.P. [Xiao-eng], An Efficient Method to Face and Emotion Detection, IVC(23), No. 11 1 October 2005, pp. 999-1008.

Appendix A

Sample Code

Train.py

```
from keras.callbacks import CSVLogger, ModelCheckpoint, EarlyStopping
from keras.callbacks import ReduceLROnPlateau
from keras.preprocessing.image import ImageDataGenerator

from models.cnn import mini_XCEPTION
from utils.datasets import DataManager
from utils.datasets import split_data
from utils.preprocessor import preprocess_input

# parameters
batch_size = 32
num_epochs = 10000
input_shape = (64, 64, 1)
validation_split = .2
verbose = 1
num_classes = 7
patience = 50
```

```
base_path = '../trained_models/emotion_models/'

# 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)

# model parameters/compilation
model = mini_XCEPTION(input_shape, num_classes)
model.compile(optimizer='adam', loss='categorical_crossentropy',
              metrics=['accuracy'])
model.summary()



datasets = ['fer2013']
for dataset_name in datasets:
    print('Training␣dataset:', dataset_name)

    # callbacks
    log_file_path = base_path + dataset_name + '_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), verbose=1)
```

```
trained_models_path = base_path + dataset_name + '_mini_XCEPTION'
model_names = trained_models_path + '.{epoch:02d}-{val_acc:.2f}.hdf5'
model_checkpoint = ModelCheckpoint(model_names, 'val_loss', verbose=1,
                                                 save_best_only=True)
callbacks = [model_checkpoint, csv_logger, early_stop, reduce_lr]

# loading dataset
data_loader = DataManager(dataset_name, image_size=input_shape[:2])
faces, emotions = data_loader.get_data()
faces = preprocess_input(faces)
num_samples, num_classes = emotions.shape
train_data, val_data = split_data(faces, emotions, validation_split)
train_faces, train_emotions = train_data
model.fit_generator(data_generator.flow(train_faces, train_emotions,
                                        batch_size),
                    steps_per_epoch=len(train_faces) / batch_size,
                    epochs=num_epochs, verbose=1, callbacks=callbacks,
                    validation_data=val_data)
```

Video_Emotion.py

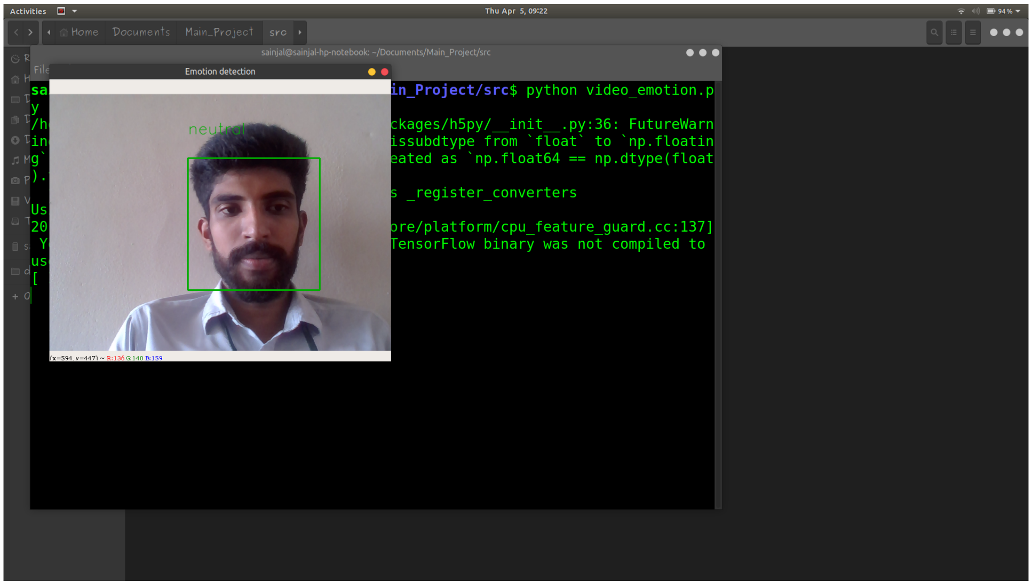
```
from statistics import mode

import cv2
from keras.models import load_model
import numpy as np

from utils.datasets import get_labels
from utils.inference import detect_faces
from utils.inference import draw_text
from utils.inference import draw_bounding_box
from utils.inference import apply_offsets
from utils.inference import load_detection_model
from utils.preprocessor import preprocess_input

# parameters for loading data and images
detection_model_path = '../trained_models/detection_models
/haarcascade_frontalface_default.xml'
emotion_model_path = '../trained_models/emotion_models
/fer2013_mini_XCEPTION.102-0.66.hdf5'
emotion_labels = get_labels('fer2013')

# hyper-parameters for bounding boxes shape
frame_window = 10
emotion_offsets = (20, 40)

# loading models
```

```
face_detection = load_detection_model(detection_model_path)
emotion_classifier = load_model(emotion_model_path, compile=False)

# getting input model shapes for inference
emotion_target_size = emotion_classifier.input_shape[1:3]

# starting lists for calculating modes
emotion_window = []

# starting video streaming
cv2.namedWindow('Emotion␣detection')
video_capture = cv2.VideoCapture(0)
# video_capture.set(3,1280)

# video_capture.set(4,1024)

while True:
    bgr_image = video_capture.read()[1]
    gray_image = cv2.cvtColor(bgr_image, cv2.COLOR_BGR2GRAY)
    rgb_image = cv2.cvtColor(bgr_image, cv2.COLOR_BGR2RGB)
    faces = detect_faces(face_detection, gray_image)

    for face_coordinates in faces:

        x1, x2, y1, y2 = apply_offsets(face_coordinates, emotion_offsets)
        gray_face = gray_image[y1:y2, x1:x2]
        try:
            gray_face = cv2.resize(gray_face, (emotion_target_size))
        except:
```

```
    continue

gray_face = preprocess_input(gray_face, True)
gray_face = np.expand_dims(gray_face, 0)
gray_face = np.expand_dims(gray_face, -1)
emotion_prediction = emotion_classifier.predict(gray_face)
emotion_probability = np.max(emotion_prediction)
emotion_label_arg = np.argmax(emotion_prediction)
emotion_text = emotion_labels[emotion_label_arg]
emotion_window.append(emotion_text)

if len(emotion_window) > frame_window:
    emotion_window.pop(0)
try:
    emotion_mode = mode(emotion_window)
except:
    continue

if emotion_text == 'angry':
    color = emotion_probability * np.asarray((255, 0, 0))
elif emotion_text == 'sad':
    color = emotion_probability * np.asarray((0, 0, 255))
elif emotion_text == 'happy':
    color = emotion_probability * np.asarray((255, 255, 0))
elif emotion_text == 'surprise':
    color = emotion_probability * np.asarray((0, 255, 255))
else:
    color = emotion_probability * np.asarray((0, 255, 0))
    color = color.astype(int)
    color = color.tolist()

    draw_bounding_box(face_coordinates, rgb_image, color)
    draw_text(face_coordinates, rgb_image, emotion_mode,
              color, 0, -45, 1, 1)

bgr_image = cv2.cvtColor(rgb_image, cv2.COLOR_RGB2BGR)
cv2.imshow('Emotion␣detection', bgr_image)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

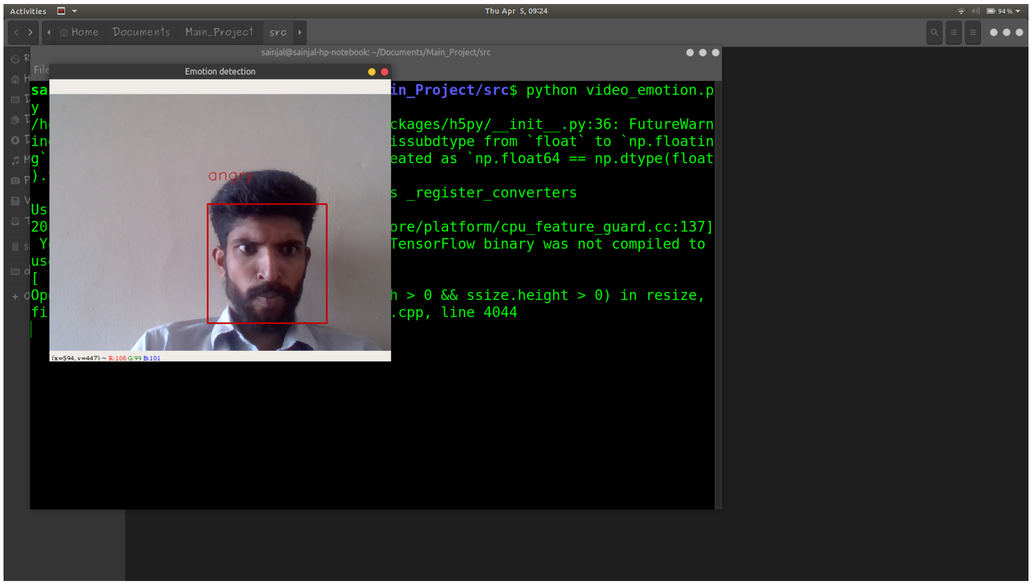
Appendix B

Screen Shots


Activities
Thu Apr 5, 09:22
94 %
Home
Documents
Main_Project
src
sainjal@sainjal-hp-notebook: ~/Documents/Main_Project/src
File
Emotion detection
neutral
(x=594, y=447) ~ R:136 G:140 B:159

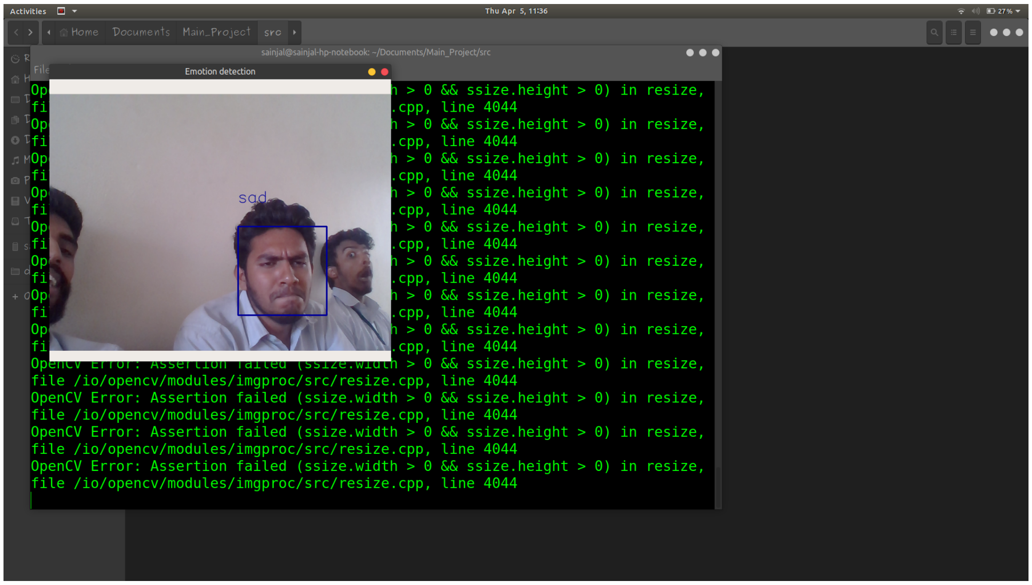
```
sa                                             in_Project/src$ python video_emotion.p
y
/h                                             ckages/h5py/__init__.py:36: FutureWarn
in                                             issubdtype from `float` to `np.floatin
g`                                             eated as `np.float64 == np.dtype(float
).
                                               s _register_converters

Us
20                                             ore/platform/cpu_feature_guard.cc:137]
 Y                                             TensorFlow binary was not compiled to
us
[
```

Figure B.1. Screen Shot 1


Activities
Thu Apr 5, 09:24
Home
Documents
Main_Project
src
sainjal@sainjal-hp-notebook: ~/Documents/Main_Project/src
File
Emotion detection
angry
(x=594, y=447) ~ R:106 G:99 B:101

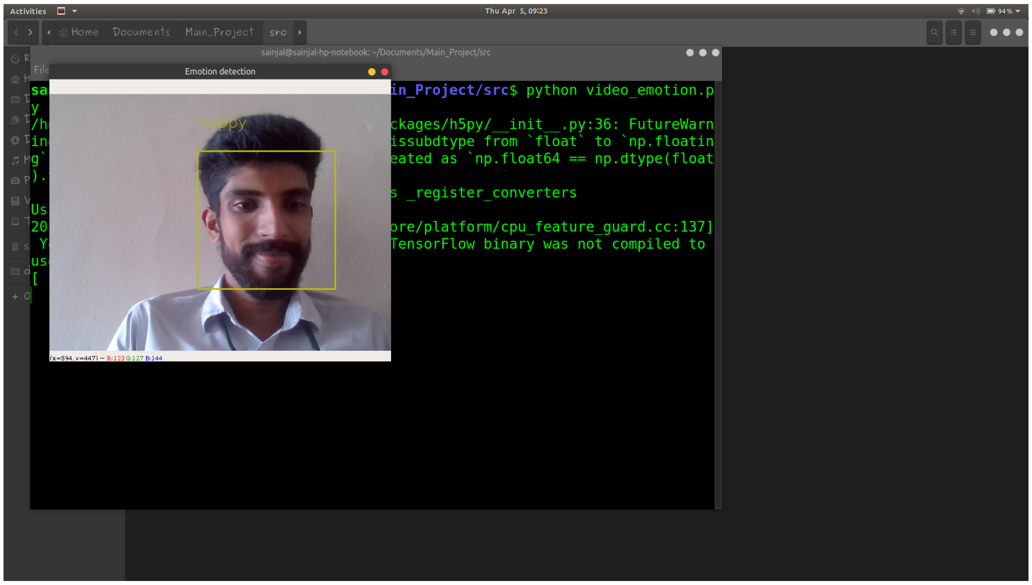
```
sa                                        in_Project/src$ python video_emotion.p
y
/h                                        ckages/h5py/__init__.py:36: FutureWarn
in                                        issubdtype from `float` to `np.floatin
g`                                        eated as `np.float64 == np.dtype(float
).
                                          s _register_converters

Us
20                                        ore/platform/cpu_feature_guard.cc:137]
 Y                                        TensorFlow binary was not compiled to
us
[
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
```

Figure B.2. Screen Shot 2



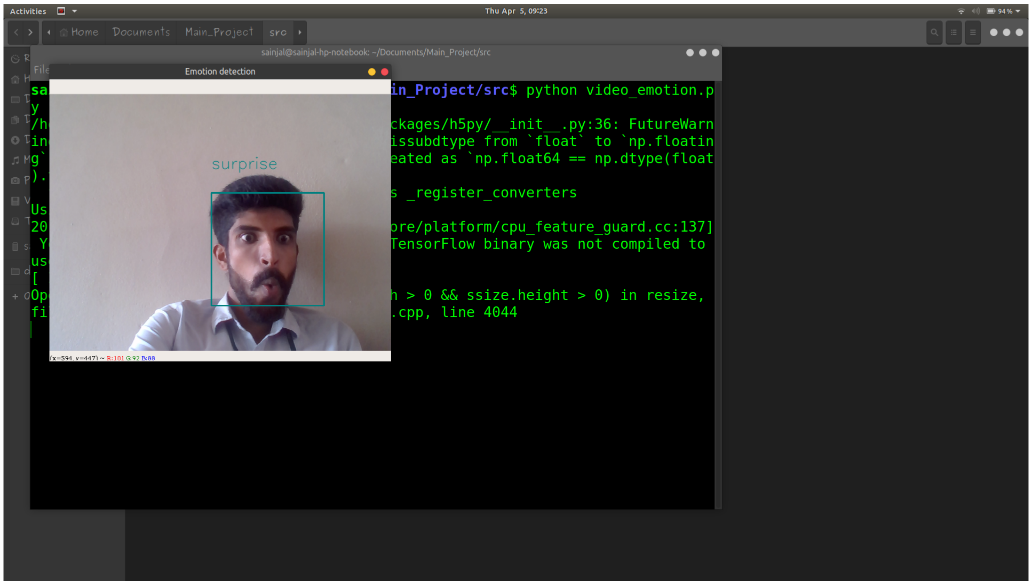
```
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
Op                                        h > 0 && ssize.height > 0) in resize,
fi                                        .cpp, line 4044
OpenCV Error: Assertion failed (ssize.width > 0 && ssize.height > 0) in resize,
file /io/opencv/modules/imgproc/src/resize.cpp, line 4044
OpenCV Error: Assertion failed (ssize.width > 0 && ssize.height > 0) in resize,
file /io/opencv/modules/imgproc/src/resize.cpp, line 4044
OpenCV Error: Assertion failed (ssize.width > 0 && ssize.height > 0) in resize,
file /io/opencv/modules/imgproc/src/resize.cpp, line 4044
OpenCV Error: Assertion failed (ssize.width > 0 && ssize.height > 0) in resize,
file /io/opencv/modules/imgproc/src/resize.cpp, line 4044
```

Figure B.3. Screen Shot 3


Activities
Thu Apr 5, 09:23
Home
Documents
Main_Project
src
sainjal@sainjal-hp-notebook: ~/Documents/Main_Project/src
File
Emotion detection
happy
(x=594, y=447) ~ R:123 G:127 B:144

```
sa                               in_Project/src$ python video_emotion.p
y
/h                               ckages/h5py/__init__.py:36: FutureWarn
in                               issubdtype from `float` to `np.floatin
g`                               eated as `np.float64 == np.dtype(float
).
                                 s _register_converters
Us
20                               ore/platform/cpu_feature_guard.cc:137]
 Y                               TensorFlow binary was not compiled to
us
[
```

Figure B.4. Screen Shot 4


Activities
Thu Apr 5, 09:23
Home
Documents
Main_Project
src
sainjal@sainjal-hp-notebook: ~/Documents/Main_Project/src
File
Emotion detection
surprise
(x=594, y=447) ~ R:101 G:92 B:88

```
sa                               in_Project/src$ python video_emotion.p
y
/h                               ckages/h5py/__init__.py:36: FutureWarn
in                               issubdtype from `float` to `np.floatin
g`                               eated as `np.float64 == np.dtype(float
).
                                 s _register_converters
Us
20                               ore/platform/cpu_feature_guard.cc:137]
 Y                               TensorFlow binary was not compiled to
us
[
Op                               h > 0 && ssize.height > 0) in resize,
fi                               .cpp, line 4044
```

Figure B.5. Screen Shot 5