##### MUSIC PLAYBACK USING EMOTIONS

##### A MINOR PROJECT REPORT

###### ***Submitted by***

##### SHUJAATULLAH KHAN [RA1411008010103]

##### VATSALYA SINGHI [RA1411008010104]

##### KRISHNAPRASADH R [RA1411008010137]

##### JEGADESH B S [RA1411008010157]

##### ARJUN M [RA1411008010283]

*Under the Guidance of*

**MS KIRTHIGA DEVI T**

(Asst Professor OG, Department of Information Technology)

*in partial fulfillment for the award of the degree*

***of***

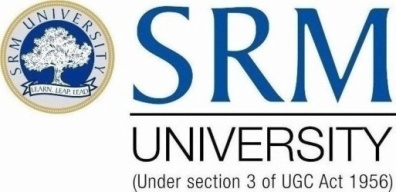
**BACHELOR OF TECHNOLOGY**

*in*

# INFORMATION TECHNOLOGY

*of*

**FACULTY OF ENGINEERING AND TECHNOLOGY**



S.R.M Nagar, Kattankulathur, Kancheepuram District

MAY 2017

SRM UNIVERSITY

KATTANKULATHUR

**BONAFIDE CERTIFICATE**

Certified that this Minor project report **“MUSIC PLAYBACK USING EMOTION RECOGNITION”**is the bonafide work of “**SHUJAATULLAH KHAN”** who carried out the project work under my supervision at SRM University, IT Department, Kattankulathur.

**SIGNATURE SIGNATURE**

Ms. KIRTHIGA DEVI T Dr.G.VADIVU

Asst. Prof (O.G)

**GUIDE HEAD OF THE DEPARTMENT**

INTERNAL EXAMINER

**DECLARATION**

I, SHUJAATULLAH KHAN (RA1411008010103) studying in III year B.Tech Information Technology program at, SRM University, Kattankulathur, Chennai, hereby declare that this project is an original work of mine and I have not verbatim copied / duplicated any material from sources like internet or from print media, excepting some vital company information / statistics and data that is provided by the company itself.

Signature of the Student

Date:

Place: SRM UNIVERSITY

**ACKNOWLEDGEMENT**

The success and the final outcome of this project required guidance and assistance from different sources and we feel extremely fortunate to have got this all along the completion of our project. Whatever we have done is largely due to such guidance and assistance and we would not forget to thank them.

We express our sincere thanks to the Head of the Department, Department of Information Technology, Dr.G.Vadivu, for all the help and infrastructure provided to us to complete this project successfully and his valuable guidance.

We owe our profound gratitude to our project guide Ms.Kirthiga Devi T, who took keen interest in our project work and guided us all along, till the completion of our project work by providing all the necessary information for developing a good system.

We are thankful and fortunate enough to get constant encouragement, support and guidance from all the Teaching staff of the Department of Information Technology which helped us in successfully completing our minor project work. Also, we would like to extend our sincere regards to all the non-teaching staff of the department of Information Technology for their timely support.

SHUJAATULLAH KHAN

RA1411008010103

TABLE OF CONTENTS

**CHAPTER NO. TITLE PAGE NO.**

ABSTRACT 6

LIST OF FIGURES 7

**1. INTRODUCTION 8**

1.1 EXISTING SYSTEM 8

1.2 DISADVANTAGES OF EXISTING SYSTEM 9

1.3 PROPOSED SYSTEM 9

1.4 ADVANTAGES OF PROPOSED SYSTEM 10

**2. REQUIREMENT ANALYSIS 11**

2.1 OBJECTIVE OF THE PROJECT 11

2.2 HARDWARE REQUIREMENTS 12

2.3 SOFTWARE REQUIREMENTS 12

**3. DESIGN 14**

3.1 INTRODUCTION 14

3.2 WORKFLOW 16

**4. IMPLEMENTATION 18**

**5. TESTING 20**

**6. CONCLUSION 25**

**APPENDIX 32**

**REFERENCES 49**

# ABSTRACT

Human emotion recognition plays an important role in the interpersonal relationship. The automatic recognition of emotions has been an active research topic from early eras. Therefore, there are several advances made in this field. Emotions are reflected from speech, hand and gestures of the body and through facial expressions. Hence extracting and understanding of emotion has a high importance of the interaction between human and machine communication. Music is considered as one of the best healing methodologies, as it soothes not only our mind, but also helps in the relaxation of our body and muscles. The main objective of this project is the real time implementation of emotion recognition system, and uses it to heal the psychology of the human mind using Music.

**LIST OF FIGURES PAGE NO**

Fig. 3.1: Overview of the Network architecture of the final model 14

Fig. 3.2: Overall architecture when compared to RNN 15

Fig. 3.3: Workflow of our System 16

Fig. 4.1: Emotion Matrix 19

Fig. 4.2: Number of images per emotion in the initial training set 19

Fig. 4.3: Number of images per emotion in the final training set 20

Fig. 5.1: Fear Face 22

Fig. 5.2: Disgusted Face 22

Fig. 5.3: Angry Face 23

Fig. 5.4: Surprised Face 23

**CHAPTER 1**

**INTRODUCTION**

Emotion recognition is the process of identifying human emotion, most typically from facial expressions. This is both something that humans do automatically but computational methodologies have also been developed.

Socially intelligent software tools can be accomplished. It serves as a Measurement systems for behavioural science. It allows a robot to understand the expressions of humans in turn enhancing its effectiveness in performing various tasks. As Human-Robot Interaction is increasing its attention nowadays. In order to put some limelight on socializing robots with human, Understanding the facial gestures and visual cues of an individual is a need.

**1.1 Existing Systems**

Emotion recognition is used for a variety of reasons:

* Affectiva uses it to help advertisers and content creators to sell their products more effectively. Affectiva also makes a Q-sensor that gauges the emotions of autistic children.
* Emotient was a startup company which utilized artificial intelligence to predict "attitudes and actions based on facial expressions".Apple indicated its intention to buy Emotient in January 2016.
* Vison provides real-time emotion recognition for web and mobile applications through a real-time API.
* Visage Technologies AB offers emotion estimation as a part of their Visage SDK for marketing and scientific research and similar purposes.
* Eyeris is an emotion recognition company that works with embedded system manufacturers including car makers and social robotic companies on integrating its face analytics and emotion recognition software; as well as with video content creators to help them measure the perceived effectiveness of their short and long form video creative.

Emotion recognition and emotion analysis are being studied by companies and universities around the world.

**1.2 Disadvantages of the Exisiting System**

There are various disadvantages in the existing emotion recognition system. Few of them are as follows:

* Pose and Frequent head movements
* Presence of structural components
* Occlusion
* Image orientation
* Subtle facial deformation
* Ambiguity and uncertainty in face motion measurement

**1.3 Proposed System**

In our proposed system we use various datasets and live video streaming to predict and recognize emotions using **Convulational Neural Networks**.We use complex modules such as **Tensor flow**, **Pyimage Search**.The basis of recognizing emotions in our case is that we use depths and areas of peaked facial lines and predict emotions according to their facial structure.Using these emotions we control music and enhance the psychological mindset of people living in a society. Thus reducing negativity and causes of suicide.

**1.4 Advantages of the Proposed System**

The various advantages of our proposed system are as follows:

* We use live video streaming to recognize emotions, thus eradicating any chance of faking emotions using images.
* Since we use deep learning algorithms there is no chance of ambiguity or uncertainity in face motion measurement.
* Occlusion results in loss of discriminative information, particularly with the case of lower face occlusion, the mouth region where most of the emotions are expressed. Hence a novel block based approach to deal with expression recognition in the presence of partial occlusion has been examined to confirm the portion of the face that holds the foremost discriminative part for emotion classification using neural networks.

**CHAPTER 2**

**REQUIREMENT ANALYSIS**

**2.1 OBJECTIVE OF THE PROJECT**

Neural networks, and deep networks in particular, are known for their need for large amounts of training data. Moreover, the choice of images used for training are responsible for a big part of the performance of the eventual model. This implies the need for a both high qualitative and quantitative dataset. For emotion recognition, several datasets are available for research, varying from a few hundred high resolution photos to tens of thousands smaller images.

The use of the Haar Feature-Based Cascaded Classiﬁer inside the OpenCV framework, all data is preprocessed. For every image, only the square part containing the face is taken, rescaled, and converted to an array with 48x48 grey-scale values.

The networks are programmed with use of the TFLearn library on top of TensorFlow, running on Python. This environment lowers the complexity of the code, since only the neuron layers have to be created, instead of every neuron. The program also provides real-time feedback on training progress and accuracy, and makes it easy to save and reuse the model after training.

The use of Python as a coding platform serves us greatly as, it being high-level programming language and being platform independent. The execution is easy as most of the code uses libraries and modules, which can be imported easily.

**2.2 HARDWARE REQUIREMENTS**

To implement the emotion recognition as an application, we use raspberry Pi3 as, it serves as a modular platform for execution of IoT based codes, which can be executed using Xming or PuTTY software. The basic needs in hardware include, a Raspberry Pi3 and a camera, which on physical implementation, a CamPi can be used, fir appropriate placement.

**2.3 SOFTWARE REQUIREMENTS**

**2.3.1 OpenCV**

OpenCV (Open Source Computer Vision Library) is an open-source BSD-licensed library that includes several hundreds of computer vision algorithms. The document describes the so-called OpenCV 2.x API, which is essentially a C++ API, as opposite to the C-based OpenCV 1.x API. The latter is described in opencv1x.pdf. OpenCV has a modular structure, which means that the package includes several shared or static libraries.

**2.3.2 h5py**

The h5py package is a Python interface to the HDF5 binary data format. It lets you store huge amounts of numerical data, and easily manipulate that data from NumPy. For example, you can slice into multi-terabyte datasets stored on disk, as if they were real NumPy arrays. Thousands of datasets can be stored in a single file, categorized and tagged however you want.

**2.3.3 TensorFlow**

TensorFlow is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API. TensorFlow was originally developed by researchers and engineers working on the Google Brain Team within Google's Machine Intelligence research organization for the purposes of conducting machine learning and deep neural networks research, but the system is general enough to be applicable in a wide variety of other domains as well.

**2.3.4 TFlearn**

TFlearn is a modular and transparent deep learning library built on top of Tensorflow. It was designed to provide a higher-level API to TensorFlow in order to facilitate and speed-up experimentations, while remaining fully transparent and compatible with it. TFLearn introduces a High-Level API that makes

neural network building and training fast and easy. This API is intuitive and fully compatible with Tensorflow.

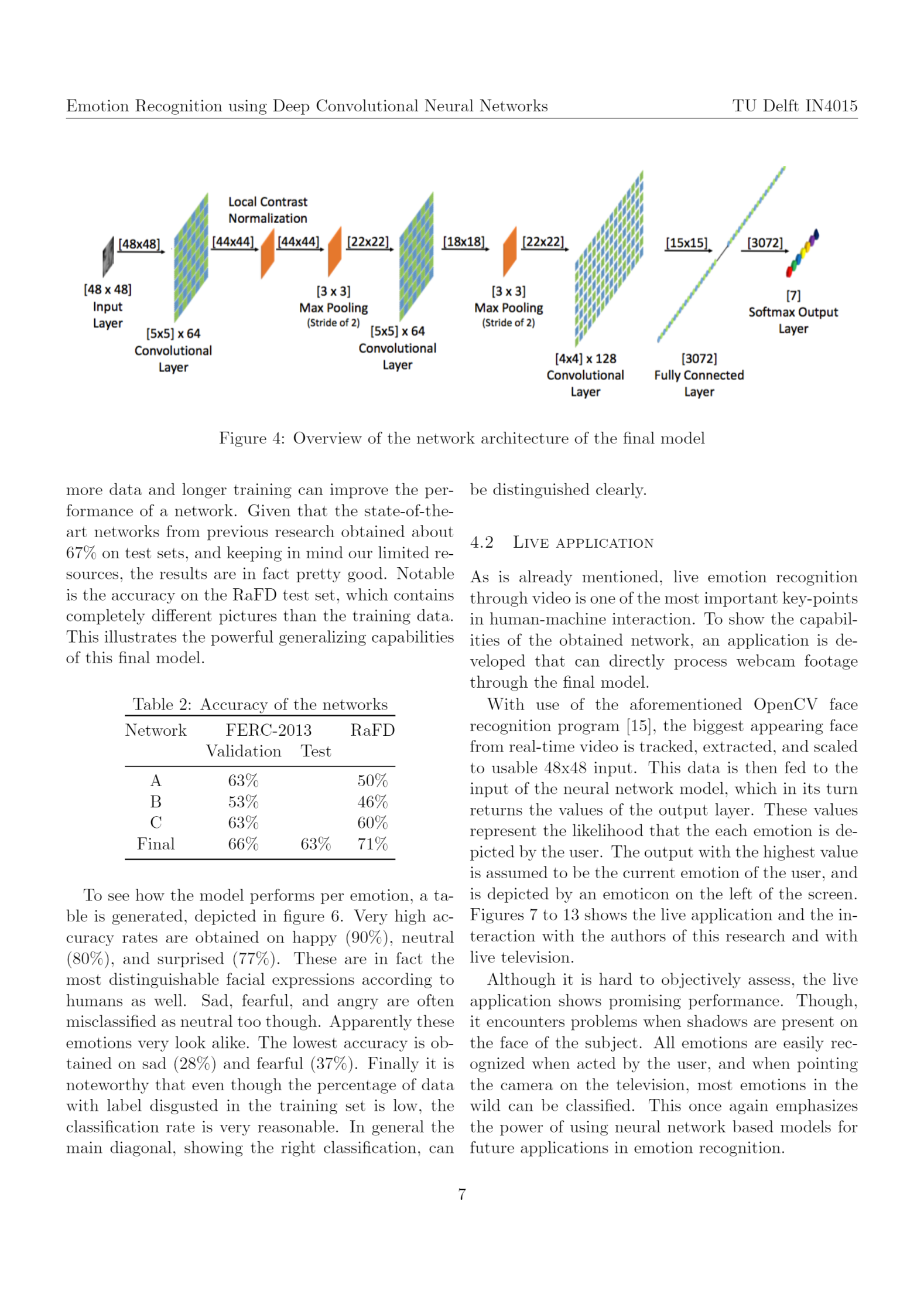
**2.3.5 NumPy**

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, useful linear algebra, Fourier transform, and random number capabilities.

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

**CHAPTER 3**

**DESIGN**

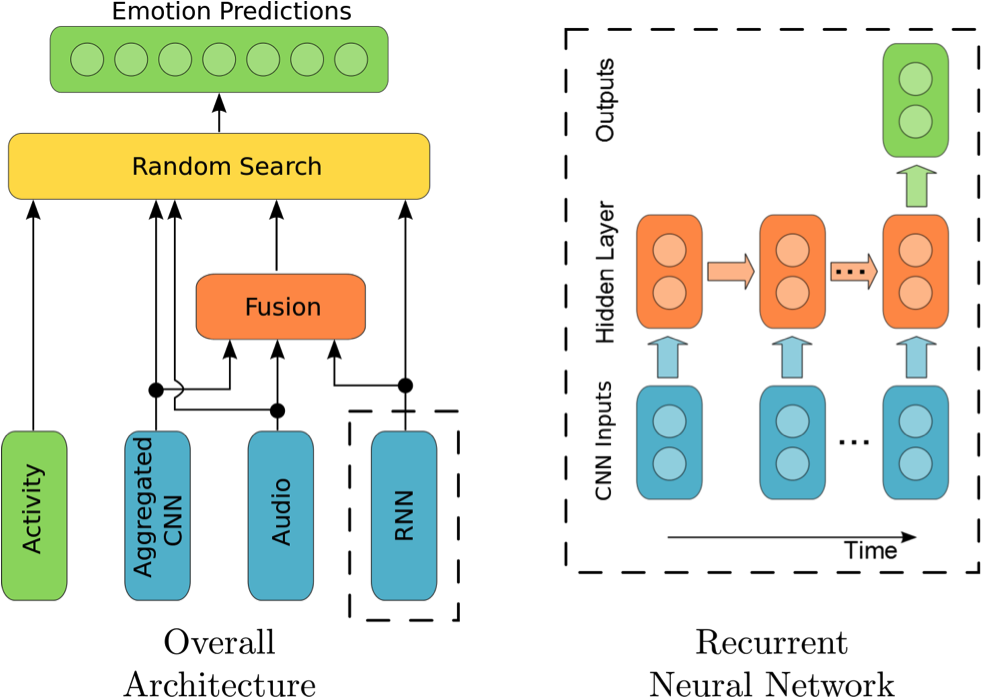


**Figure 3.1 : Overview of the Network architecture of the final model**

**3.1 INTRODUCTION**

Convolutional Neural networks (CNN) are very similat to ordinary neural neworks, they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer and all the tips/tricks we developed for learning regular Neural Networks still apply. An Overview of the Network architecture is depicted in Figure 3.1.

CNN is a specially designed multi-layer perceptron to identify two-dimension shapes. Therefore dimensional information retained in waveform points is effectively utilized by CNN. CNN model due to its characteristics of adaptive feature extraction, it is applied for emotion recognition. The overall architecture of CNN when compared to RNN is as given in Figure 3.2.

****

**Figure 3.2 : Overall architecture when compared to RNN**

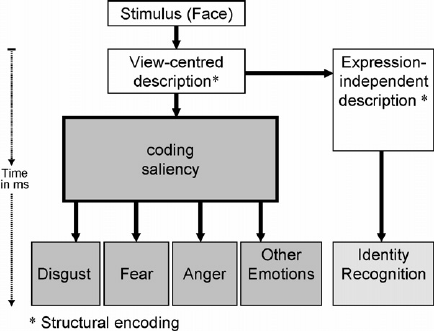
Thus helping in getting the various weights and biases of the the face and helping in predicting the emotion by comparing it with the data-sets and predicting a final output.

In the Input part is a prerequisite for face recognition system. Image acquisition operation is performed in this part. Live captured images are converted to digital data for performing image-processing computations. These captured images are sent to emotion detection algorithm.

We use various datasets, to refine the detection part of the system. The data-sets differ mainly on quantity, quality, and 'cleanness' of the images.

The networks are programmed with use of the TFLearn library on top of TensorFlow, running on Python. This environment lowers the complexity of the code, since only the neuron layers have to be created, instead of every neuron. The program also provides real-time feedback on training progress and accuracy, and makes it easy to save and reuse the model after training.

**3.3 WORKFLOW**

****

**Figure 3.3 : Workflow of our System**

Once the image is captured using a high resolution the view-centered description of image is sent further analysis. This is the input to a convolutional neural network. The first step in image analysis is often to perform some local filtering of the image to enhance edges in the image captured. This is done by taking the neighborhood of each pixel and convolves it with a certain mask which is default set of weights. Basically you compute a linear combination of those pixels. So if you have a positive weight on the center pixel and negative weights on the surrounding pixels you compute the difference between the center pixel and the surrounding, giving you a crude kind of edge detector.

In convolutional neural networks, every network layer acts as a detection filter for the presence of specific features or patterns present in the original data. The first layers in a CNN detect features that can be recognized and interpreted relatively easy. Later layers detect increasingly features that are more abstract and are usually present in many of the larger features detected by earlier layers. The last layer of the CNN is able to make an ultra-specific classification by combining all the specific features detected by the previous layers in the input data.

Using this depths and weights of the image captured can be classified into the emotions like disgust, fear, happy and anger.

**CHAPTER 4**

**IMPLEMENTATION**

With the use of the Haar Feature-Based Cascaded Classifier inside the OpenCV framework, all data is preprocessed.For every image, only the square part containing the face is taken, rescaled, and converted to an array with 48x48 grey-scale values. The networks are programmed with use of the TFLearn library on top of TensorFlow, running on Python. This environment lowers the complexity of the code, since only the neuron layers have to be created, instead of every neuron. The program also provides real-time feedback on training progress and accuracy, and makes it easy to save and reuse the model after training

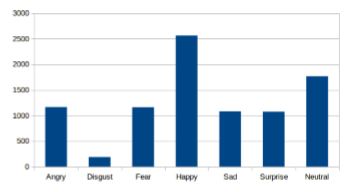
We use the FER-2013 We use the FER-2013 Faces Database , a set of 28,709 pictures of people displaying 7 emotional expressions (angry, disgusted, fearful, happy, sad, surprised and neutral).

To identify-faces, the computer uses the arrangement and shape of e.g. eyebrows and lips to determine the facial expression and hence the emotion of a person. One possible application for this lies in the area of surveillance and behavioural analysis by law enforcement. Furthermore such techniques are used in digital cameras to automatically take pictures when the user smiles.

The data-sets differ mainly on quantity, quality,and 'cleanness' of the images. The FERC-2013 set for example has about 32000 low resolution images, where the RaFD provides 8000 high resolution photos. Furthermore it can be noticed that the facial expressions in the CK+ and RaFD are posed (i.e.'clean').

****

**Figure 4.1: Emotion matrix**

****

**Figure 4.2 : Number of images per emotion in the initial training set**

All data-sets are converted in the form of arrays and the face emotion is recognized using lips, cheeks, eyebrows and the face depth. We have used OpenCV and TensorFlow for Face Detection and for the emotion recognition part, the data-sets comes into action. The programming language used is Python since it has premade modules and good to program.

****

**Figure 4.3 : Number of images per emotion in the final training set**

**CHAPTER 5**

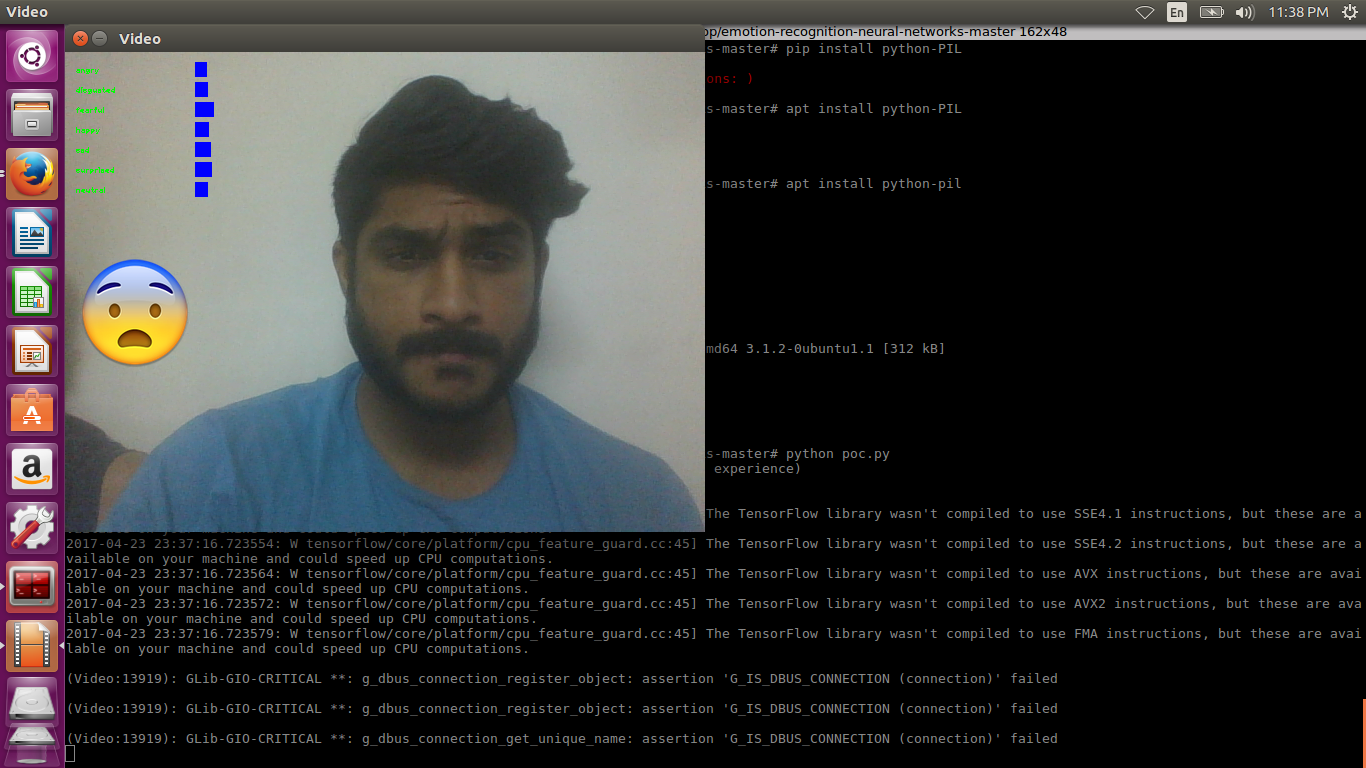
**TESTING**

Since the entire coding was done in Python as a coding platform, its more versatile and lightweight. The memory taken by the code is sufficiently low. Also, Python being one of the high-level coding languages, it offers n number of libraries which help in refining the code even more better.

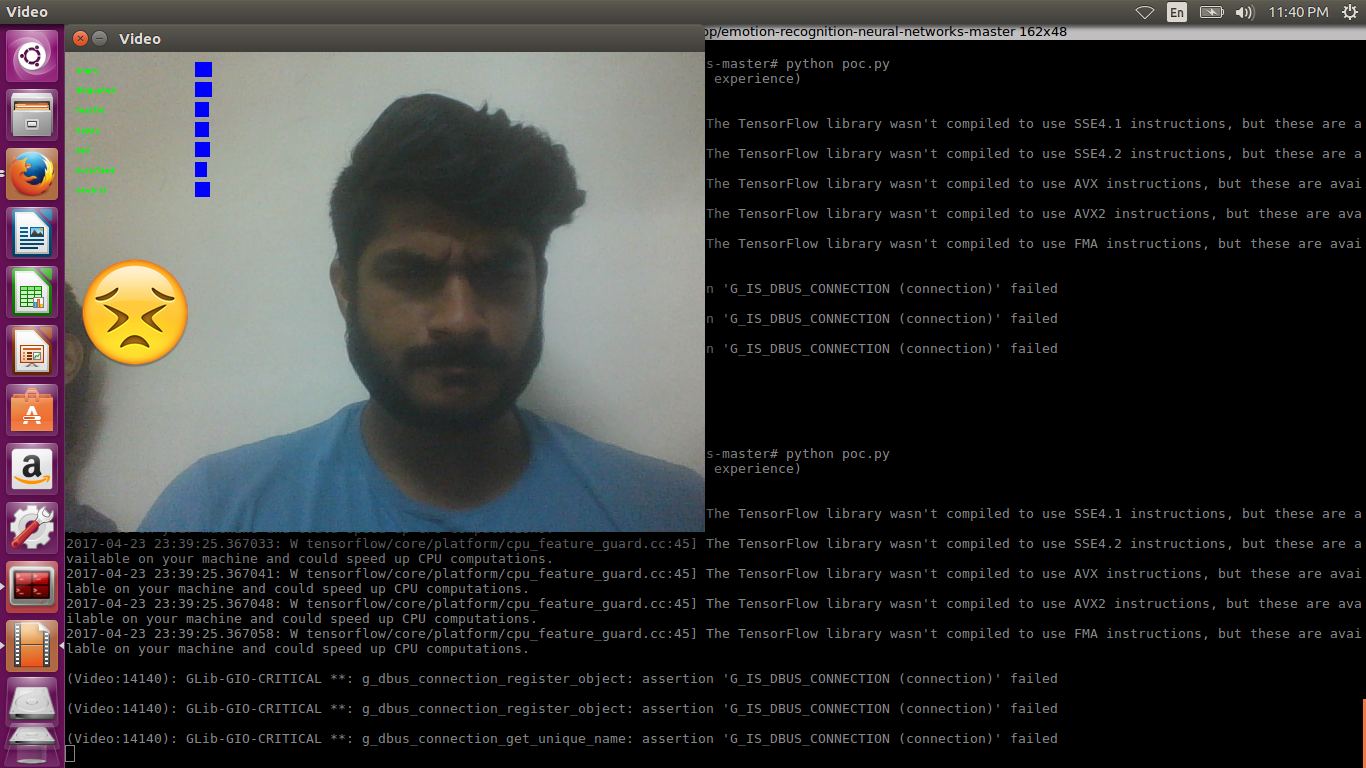
However, since the entire project is based on Haara-scades data-sets, its wide open to bugs. We observed that, Open CV uses a bunch load of memory and cores for processing the data-sets. It also requires a high ended graphics card for efficient processing.

Since our system, was a bit on the low-end, with a 4GB RAM, and a minimal, 1GB AMD Radeon Graphics, which made the processing of data-sets slow. Thus, for showing a proper output, we had to rerun the code every-time for predicting a new emotion and

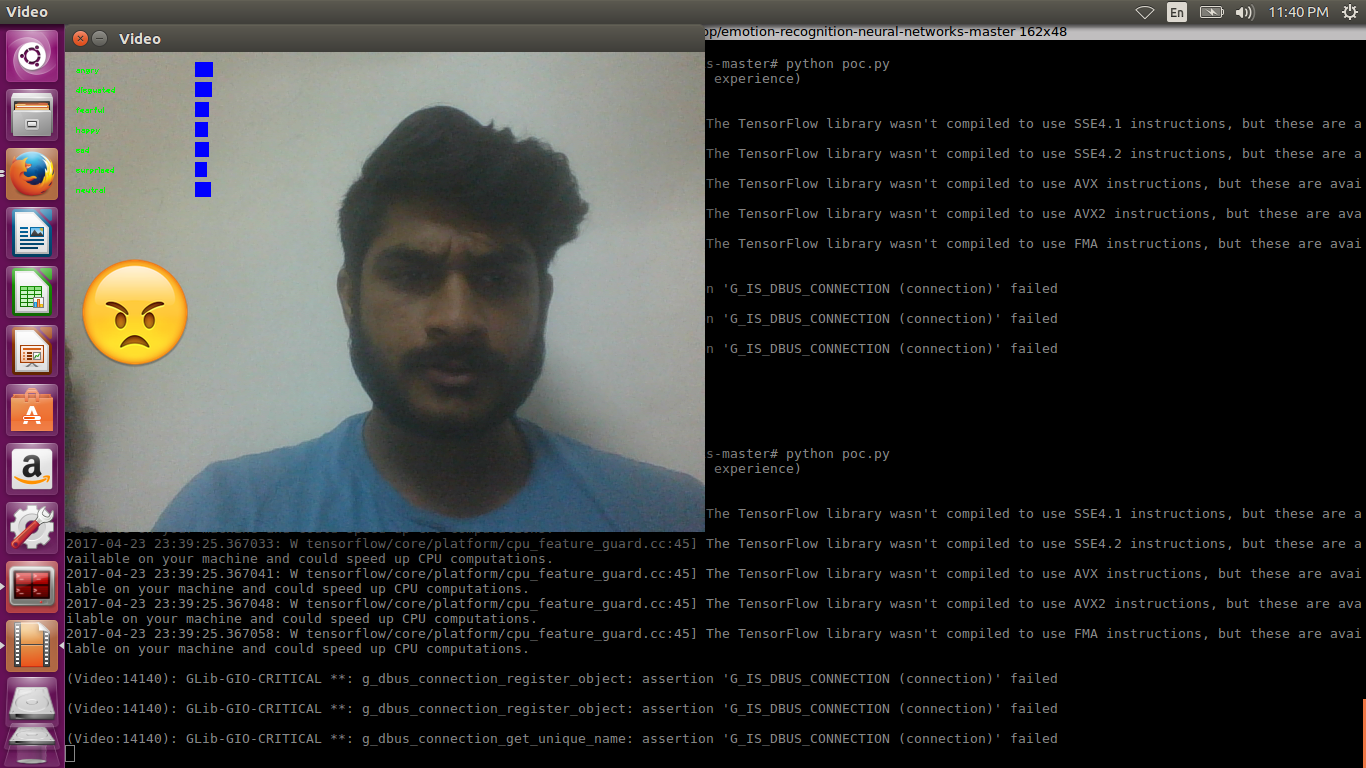
With use of the aforementioned Open CV face recognition program, the biggest appearing face from real-time video is tracked, extracted, and scaled to usable 48x48 input. This data is then fed to the input of the neural network model, which in its turn returns the values of the output layer. These values represent the likelihood that the each emotion is depicted by the user. The output with the highest value is assumed to be the current emotion of the user, and is depicted by an emoticon on the left of the screen. Although it is hard to objectively assess, the live application shows promising performance. Though, it encounters problems when shadows are present on the face of the subject. All emotions are easily recognized when acted by the user, and when pointing the camera on the television, most emotions in the wild can be classified



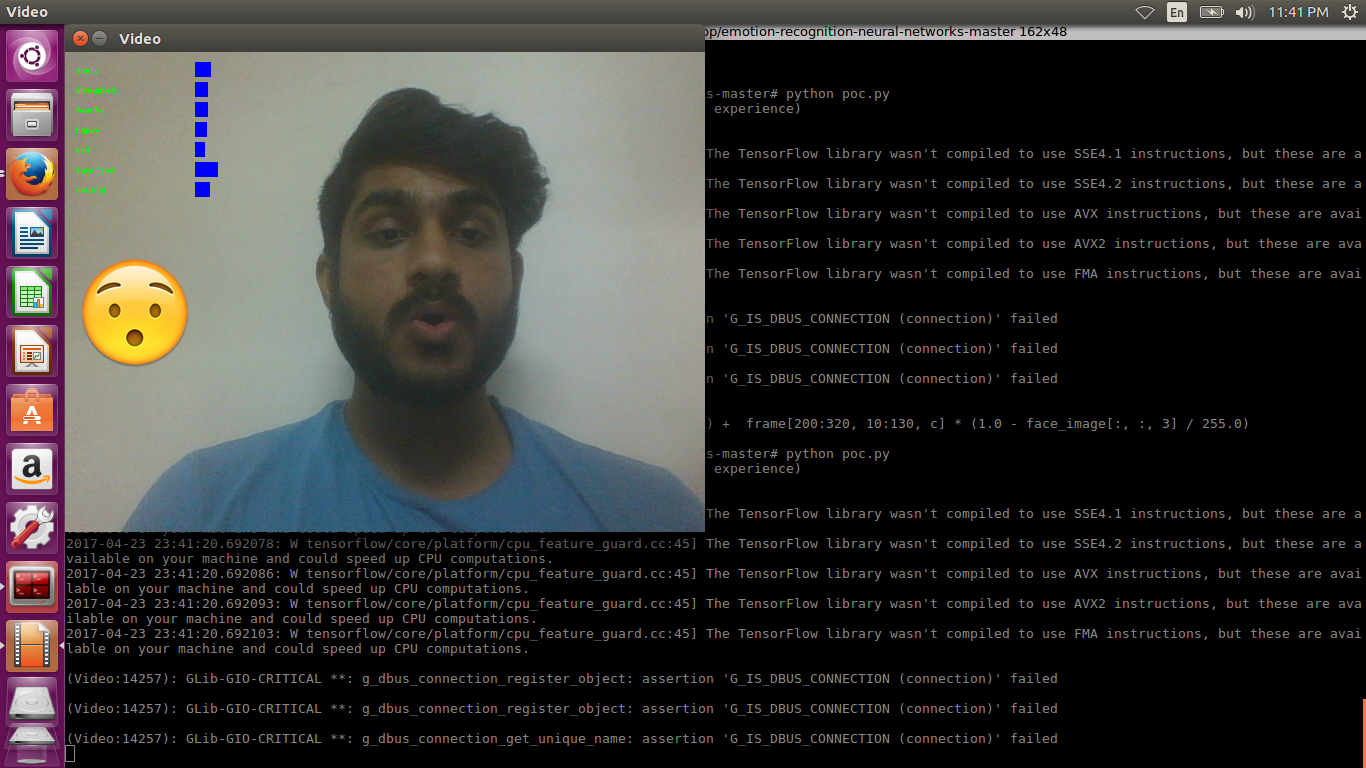
**Figure 5.1: Fear face**



**Figure 5.2: Disgusted face**



**Figure 5.3 : Angry face**



**Figure 5.4 : Surprised face**

**CHAPTER 6**

**CONCLUSION**

**6.1 INTRODUCTION**

Human emotions are one which can be predicted easily, or which can be faked. No one knows, what the mind of the other person truly feels. We thus, made an effort to bring out the emotions and resolve the negative emotions, by bringing the, to a happy state, using therapy.

Emotions are detected using Convolution Neural Networks, to better grasp the eccentric depths and weight pf the facial matrix. The depths are thus analyzed and compared with the existing data-sets to compare and contrast an emotion.

Our faces, thus say all. Each and every micro-projection is taken into consideration, and thus the emotion matrix was plotted. Music being one of the best healers, we control Music using emotions; based on the emotion detected, the music is made to change. Emotion recognition thus, in near future will prove its worth in the upcoming high-end technologies.

**6.2 FUTURE ENHANCEMENTS**

The project can be used in mobile, web applications in order to get a more user interactive and user refined experience. The entire project can be made lighter in terms of the memory and time units taken to execute the algorithms. It has a variety of applications in various fields.

**6.2.1 MARKETING**

It's safe to say that we should expect a great deal of change in the world of marketing moving forward. Gabi Zijderveld, CMO at Affectiva, provides a peek into the future by explaining the evolution of emotion recognition technology:

Initially, the technology was used to understand how consumers engage with their brand content and advertising, and how these emotions then influence brand awareness and purchase intent. Now the technology is also used to infuse consumer experiences, apps and interactive advertising with Emotion AI. This will help to transform the face of marketing and advertising by reading human emotions and then adapting consumer experiences to these emotions in real time. The technology gives marketers the power to truly delight and engage their customers with uniquely dynamic and personalized interactions.”

Web development is already on the path towards more personalization. As emotion recognition technology becomes more sophisticated and more deeply embedded in our array of devices, it will become expected that our computers and phones provide us with a continual progression of customized triggers and messaging. The technology will be found even in future car dashboards, refrigerator doors, and conference room walls -- essentially any surface will become a possible means for detection of emotions.

Social media will constantly focus on each user’s emotions. For example, in the future expect Facebook’s algorithm to focus just as much on one’s emotional reactions as it does to one’s historical click behavior, providing a unique social environment that goes far beyond prediction of the types of posts, pages, and ads one would like. Expect Facebook Ads to provide advertisers with the ability to hyper target not only based on age, geography, and job titles, but also on the individual’s emotional state or progression of emotional states.

Online marketing will likely evolve into sequential experiences, with deeper engagement upon recognition of positive emotional reactions. You can also expect more deeply embedded forms of marketing similar to product placement. Ultimately, expect emotion recognition to be just another core component of marketing, similar to how “digital marketing” is now really just “marketing.”Emotion recognition technology is clearly bringing about a revolution in marketing.

**6.2.2 TELEPSYCHIATRY**

The famous science fiction story and the movie it became — Minority Report — depicts a future world in which crimes are anticipated and prevented before they can occur. While we’ll likely never achieve true precognition within society, much less law enforcement, technology is even now being pioneered that vastly expands how we detect and process human beings’ emotional and mental states and responses. This technology may one day soon become the key to predictive mental health counseling, in which a client’s needs are anticipated in real time and his or her reactions to treatment are assessed as they occur.

Intriguing as that idea seems, when the technology is coupled to video conferencing, it has real implications for the future of telemedicine and telepsychiatry. Nina Ruhe, writing for MedCityNews, explains that “emotion recognition technology” has promise for the field of telehealth.

“Almost every Smartphone and tablet possess face recognition technology through a camera,” Ruhe points out. “New technology is now able to read your emotions based on different facial cues. In the world of telemedicine where patients are evaluated over mobile platforms, the ability for a medical professional to discern what the patient is feeling can be useful in the healing process.”

Perhaps much more importantly, that type of real-world feedback could not only help tele-psychiatrists better serve their clients’ needs, but might also serve as a very worthwhile gauge of the efficacy of the treatment itself. One of the biggest problems faced by all service providers when assessing a patient’s emotional state is that some degree of self-reporting is necessary to the process. There are clues, yes, but a patient must ultimately be honest about what he is thinking or feeling.

Emotion recognition technology could help providers verify that a client is providing honest feedback versus the cues that reveal how he or she is truly feeling and reacting during treatment.

“Telemedicine vendors with an interest in moving more into specialized areas like telepsychiatry,” writes Ruhe, can especially benefit from emotion recognition technology so they can understand what their patients are feeling even if the patient isn’t physically present or is not explaining their emotions to a psychiatrist or psychologist.”

Understanding an individual’s emotional state is extremely important to understanding them as a person. This technology, should it become the norm, would be of tremendous benefit to that process. While the outcome remains to be seen, the promise is real… and the coming years should see even more technological innovation along these same lines.

**6.2.3 EDUCATION AND e-EDUCATION**

There is lots of evidence, that some emotional states support learning processes and other suppress them [Hudlicka 2003, Picard 2003, Sheng et al. 2010]. The dis-tinction of the two groups of emotional states in some cases is not obvious, for ex-ample such positive mood as hilarity is not good for learning processes, while slightly negative emotional states foster critical thinking and are appropriate for analytical tasks [Landowska 2013]. Automatic emotion recognition algorithms can help to explore this phenomena by making assessments of learner emotional states more objective than typical questionnaire-based investigations.

**Scenario 1. Emotional templates of educational tasks.**

The purpose of this scenario is investigation on emotional states that occur dur-ing different types of educational tasks. This investigation aims at identification of emotional templates of educational tasks, that can be defined as distinguishable sets of effective and counter-productive emotional states for solving specific task types. To perform this investigation representative set of educational tasks should be prepared and both learners’ performance in task execution and his/her emotion-al state must be measured. Analysis of the correlation between performance and emotional states would enable to justify statements on effective and counter-productive emotions for specific task types, however a significant number of re-spondents should be engaged in order to make the thesis reliable. Information on effective emotional states can be then used in educational problems diagnosis, ed-ucational tool design or in further exploration of educational processes.

**Scenario 2. Emotional stereotypes of learners.**

Emotionality is one of the elements of human personality and may differ significantly based on in-born temper, previous experience and socialization process. However some emotional reactions are common for people living in one culture or having the same experience and similar characteristics is expected in educational processes. Learner affective stereotype is a definition of typical emotional states that might be observed in educational settings. It is expected, that novice learners will more frequently show symptoms of frustration, while more experienced ones could feel boredom. To support that thesis with evidence, emotional states of dif-ferent (novice/experienced) students will be measured and recognized while they perform the same tasks set of growing difficulty. Learners’ stereotypes can be then used in e-educational environments to adapt learning paths and/or interaction models, when no individual information on user is available.

**Scenario 3. Evaluation of educational resources.**

The goal of this scenario is evaluation of educational resources, especially those prepared for self-learning. In distance and electronic education one of the critical success factors is learner discipline in following provided learning path. When one fails to deal with fluctuation of motivation and attention, learning pro-cesses are paused or even abandoned. One of the frequently launched cause for course resignation is: “Boring resources”. In this scenario observation of student’s interaction with resources is combined with monitoring his/her emotional state in order to identify parts of resources that cause boredom. That information may be then used to remove weak points and improve overall resource quality. A set of different types of educational resources including recorded lectures, screencasts and interactive materials will be investigated. This scenario might be also used for quality evaluation of resources provided in virtual universities and other distance learning environments.

**Scenario 4. Usability testing of educational tools.**

In this scenario usability of educational tools is evaluated. Software usability tests are usually based on eye-tracking techniques and we propose to extend it with user emotion recognition, which can be a valuable information while evaluat-ing user experience [Kołakowska et al. 2013]. Typical tasks performed with edu-cational tools include: educational tool access, resource search, resource launch, performing interactive tasks, viewing results or feedback information, communi-cation with teachers or class mates and more. More specific task description for the scenario will be performed using cognitive walkthrough method [Blackmon et al. 2002]. Then representative group of students will be asked to perform tasks in controlled environment that will additionally record and recognize their emotional states. Information on affect and its fluctuations (especially identification of frustration) can help to improve software products that are designed to support learning processes.

**APPENDIX**

**poc.py**

import cv2

import sys

from constants import \*

from emotion\_recognition import EmotionRecognition

from os.path import join

import numpy as np

import matplotlib.pyplot as plt

# Load Model

network = EmotionRecognition()

network.build\_network()

images = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_IMAGES\_FILENAME))

labels = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_LABELS\_FILENAME))

images = images.reshape([-1, SIZE\_FACE, SIZE\_FACE, 1])

labels = labels.reshape([-1, len(EMOTIONS)])

print '[+] Loading Data'

data = np.zeros((len(EMOTIONS),len(EMOTIONS)))

for i in xrange(images.shape[0]):

result = network.predict(images[i])

data[np.argmax(labels[i]), result[0].index(max(result[0]))] += 1

#print x[i], ' vs ', y[i]

# Take % by column

for i in range(len(data)):

total = np.sum(data[i])

for x in range(len(data[0])):

data[i][x] = data[i][x] / total

print data

print '[+] Generating graph'

c = plt.pcolor(data, edgecolors = 'k', linewidths = 4, cmap = 'Blues', vmin = 0.0, vmax = 1.0)

def show\_values(pc, fmt="%.2f", \*\*kw):

from itertools import izip

pc.update\_scalarmappable()

ax = pc.get\_axes()

ax.set\_yticks(np.arange(len(EMOTIONS)) + 0.5, minor = False)

ax.set\_xticks(np.arange(len(EMOTIONS)) + 0.5, minor = False)

ax.set\_xticklabels(EMOTIONS, minor = False)

ax.set\_yticklabels(EMOTIONS, minor = False)

for p, color, value in izip(pc.get\_paths(), pc.get\_facecolors(), pc.get\_array()):

x, y = p.vertices[:-2, :].mean(0)

if np.all(color[:3] > 0.5):

color = (0.0, 0.0, 0.0)

else:

color = (1.0, 1.0, 1.0)

ax.text(x, y, fmt % value, ha = "center", va = "center", color = color, \*\*kw)

show\_values(c)

plt.xlabel('Predicted Emotion')

plt.ylabel('Real Emotion')

plt.show()

**emotion\_recognition.py**

from \_\_future\_\_ import division, absolute\_import

import re

import numpy as np

from dataset\_loader import DatasetLoader

import tflearn

from tflearn.layers.core import input\_data, dropout, fully\_connected, flatten

from tflearn.layers.conv import conv\_2d, max\_pool\_2d, avg\_pool\_2d

from tflearn.layers.merge\_ops import merge

from tflearn.layers.normalization import local\_response\_normalization

from tflearn.layers.estimator import regression

from constants import \*

from os.path import isfile, join

import random

import sys

class EmotionRecognition:

def \_\_init\_\_(self):

self.dataset = DatasetLoader()

def build\_network(self):

# Smaller 'AlexNet'

# https://github.com/tflearn/tflearn/blob/master/examples/images/alexnet.py

print('[+] Building CNN')

self.network = input\_data(shape = [None, SIZE\_FACE, SIZE\_FACE, 1])

self.network = conv\_2d(self.network, 64, 5, activation = 'relu')

#self.network = local\_response\_normalization(self.network)

self.network = max\_pool\_2d(self.network, 3, strides = 2)

self.network = conv\_2d(self.network, 64, 5, activation = 'relu')

self.network = max\_pool\_2d(self.network, 3, strides = 2)

self.network = conv\_2d(self.network, 128, 4, activation = 'relu')

self.network = dropout(self.network, 0.3)

self.network = fully\_connected(self.network, 3072, activation = 'relu')

self.network = fully\_connected(self.network, len(EMOTIONS), activation = 'softmax')

self.network = regression(self.network,

optimizer = 'momentum',

loss = 'categorical\_crossentropy')

self.model = tflearn.DNN(

self.network,

checkpoint\_path = SAVE\_DIRECTORY + '/emotion\_recognition',

max\_checkpoints = 1,

tensorboard\_verbose = 2

)

self.load\_model()

def load\_saved\_dataset(self):

self.dataset.load\_from\_save()

print('[+] Dataset found and loaded')

def start\_training(self):

self.load\_saved\_dataset()

self.build\_network()

if self.dataset is None:

self.load\_saved\_dataset()

# Training

print('[+] Training network')

self.model.fit(

self.dataset.images, self.dataset.labels,

validation\_set = (self.dataset.images\_test, self.dataset.\_labels\_test),

n\_epoch = 100,

batch\_size = 50,

shuffle = True,

show\_metric = True,

snapshot\_step = 200,

snapshot\_epoch = True,

run\_id = 'emotion\_recognition'

)

def predict(self, image):

if image is None:

return None

image = image.reshape([-1, SIZE\_FACE, SIZE\_FACE, 1])

return self.model.predict(image)

def save\_model(self):

self.model.save(join(SAVE\_DIRECTORY, SAVE\_MODEL\_FILENAME))

print('[+] Model trained and saved at ' + SAVE\_MODEL\_FILENAME)

def load\_model(self):

if isfile(join(SAVE\_DIRECTORY, SAVE\_MODEL\_FILENAME)):

self.model.load(join(SAVE\_DIRECTORY, SAVE\_MODEL\_FILENAME))

print('[+] Model loaded from ' + SAVE\_MODEL\_FILENAME)

def show\_usage():

# I din't want to have more dependecies

print('[!] Usage: python emotion\_recognition.py')

print('\t emotion\_recognition.py train \t Trains and saves model with saved dataset')

print('\t emotion\_recognition.py poc \t Launch the proof of concept')

if \_\_name\_\_ == "\_\_main\_\_":

if len(sys.argv) <= 1:

show\_usage()

exit()

network = EmotionRecognition()

if sys.argv[1] == 'train':

network.start\_training()

network.save\_model()

elif sys.argv[1] == 'poc':

import poc

else:

show\_usage()

**constants.py**

CASC\_PATH = './haarcascade\_files/haarcascade\_frontalface\_default.xml'

SIZE\_FACE = 48

EMOTIONS = ['angry', 'disgusted', 'fearful', 'happy', 'sad', 'surprised', 'neutral']

SAVE\_DIRECTORY = './data/'

SAVE\_MODEL\_FILENAME = 'Gudi\_model\_100\_epochs\_20000\_faces'

SAVE\_DATASET\_IMAGES\_FILENAME = 'data\_set\_fer2013.npy'

SAVE\_DATASET\_LABELS\_FILENAME = 'data\_labels\_fer2013.npy'

SAVE\_DATASET\_IMAGES\_TEST\_FILENAME = 'test\_set\_fer2013.npy'

SAVE\_DATASET\_LABELS\_TEST\_FILENAME = 'test\_labels\_fer2013.npy'

**dataset\_loader.py**

from os.path import join

import numpy as np

from constants import \*

import cv2

class DatasetLoader(object):

def \_\_init\_\_(self):

pass

def load\_from\_save(self):

self.\_images = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_IMAGES\_FILENAME))

self.\_labels = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_LABELS\_FILENAME))

self.\_images\_test = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_IMAGES\_TEST\_FILENAME))

self.\_labels\_test = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_LABELS\_TEST\_FILENAME))

self.\_images = self.\_images.reshape([-1, SIZE\_FACE, SIZE\_FACE, 1])

self.\_images\_test = self.\_images.reshape([-1, SIZE\_FACE, SIZE\_FACE, 1])

self.\_labels = self.\_labels.reshape([-1, len(EMOTIONS)])

self.\_labels\_test = self.\_labels.reshape([-1, len(EMOTIONS)])

@property

def images(self):

return self.\_images

@property

def labels(self):

return self.\_labels

@property

def images\_test(self):

return self.\_images\_test

@property

def labels\_test(self):

return self.\_labels\_test

@property

def num\_examples(self):

return self.\_num\_examples

**manual\_poc.py**

# Proof-of-concept

import cv2

import sys

import os

from constants import \*

from emotion\_recognition import EmotionRecognition

import numpy as np

def format\_image(image):

if len(image.shape) > 2 and image.shape[2] == 3:

image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

else:

image = cv2.imdecode(image, cv2.CV\_LOAD\_IMAGE\_GRAYSCALE)

faces = cv2.CascadeClassifier(CASC\_PATH).detectMultiScale(

image,

scaleFactor = 1.3,

minNeighbors = 5

)

# None is we don't found an image

if not len(faces) > 0:

return None

max\_area\_face = faces[0]

for face in faces:

if face[2] \* face[3] > max\_area\_face[2] \* max\_area\_face[3]:

max\_area\_face = face

# Chop image to face

face = max\_area\_face

image = image[face[1]:(face[1] + face[2]), face[0]:(face[0] + face[3])]

# Resize image to network size

try:

image = cv2.resize(image, (SIZE\_FACE, SIZE\_FACE), interpolation = cv2.INTER\_CUBIC) / 255.

while True:

cv2.imshow("frame", image)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

except Exception:

print("[+] Problem during resize")

return None

# cv2.imshow("Lol", image)

# cv2.waitKey(0)

return image

# Load Model

network = EmotionRecognition()

network.build\_network()

files = []

for f in os.listdir("./"):

ext = os.path.splitext(f)[1]

if ext.lower() in [".jpg"]:

files.append(f)

for f in files:

frame = cv2.imread(f)

# Predict result with network

result = network.predict(format\_image(frame))

if result is not None:

for index, emotion in enumerate(EMOTIONS):

print emotion, ': ', result[0][index]

print "Emotion: of ", f, "-", EMOTIONS[np.argmax(result[0])]

**plot\_emotion\_matrix.py**

import cv2

import sys

from constants import \*

from emotion\_recognition import EmotionRecognition

from os.path import join

import numpy as np

import matplotlib.pyplot as plt

# Load Model

network = EmotionRecognition()

network.build\_network()

images = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_IMAGES\_FILENAME))

labels = np.load(join(SAVE\_DIRECTORY, SAVE\_DATASET\_LABELS\_FILENAME))

images = images.reshape([-1, SIZE\_FACE, SIZE\_FACE, 1])

labels = labels.reshape([-1, len(EMOTIONS)])

print '[+] Loading Data'

data = np.zeros((len(EMOTIONS),len(EMOTIONS)))

for i in xrange(images.shape[0]):

result = network.predict(images[i])

data[np.argmax(labels[i]), result[0].index(max(result[0]))] += 1

#print x[i], ' vs ', y[i]

# Take % by column

for i in range(len(data)):

total = np.sum(data[i])

for x in range(len(data[0])):

data[i][x] = data[i][x] / total

print data

print '[+] Generating graph'

c = plt.pcolor(data, edgecolors = 'k', linewidths = 4, cmap = 'Blues', vmin = 0.0, vmax = 1.0)

def show\_values(pc, fmt="%.2f", \*\*kw):

from itertools import izip

pc.update\_scalarmappable()

ax = pc.get\_axes()

ax.set\_yticks(np.arange(len(EMOTIONS)) + 0.5, minor = False)

ax.set\_xticks(np.arange(len(EMOTIONS)) + 0.5, minor = False)

ax.set\_xticklabels(EMOTIONS, minor = False)

ax.set\_yticklabels(EMOTIONS, minor = False)

for p, color, value in izip(pc.get\_paths(), pc.get\_facecolors(), pc.get\_array()):

x, y = p.vertices[:-2, :].mean(0)

if np.all(color[:3] > 0.5):

color = (0.0, 0.0, 0.0)

else:

color = (1.0, 1.0, 1.0)

ax.text(x, y, fmt % value, ha = "center", va = "center", color = color, \*\*kw)

show\_values(c)

plt.xlabel('Predicted Emotion')

plt.ylabel('Real Emotion')

plt.show()

**REFERENCES**

1. Open Source Computer Vision. Face detection using haar cascades.

URL:http://docs.opencv.org/master/d7/d8b/tutorial\_py\_face\_detection.html

1. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classiﬁcation with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.
2. Speech emotion recognition using convolutional and Recurrent Neural Networks.

URL: http://ieeexplore.ieee.org/document/7820699/

1. Emotion recognition from multi-channel EEG data through Convolutional Recurrent Neural Network

URL:http://ieeexplore.ieee.org/document/7822545/

1. Self-Configuring Ensemble of Neural Network Classifiers for Emotion Recognition in the Intelligent Human-Machine Interaction

URL:http://ieeexplore.ieee.org/document/7376829/

1. An experimental study of speech emotion recognition based on deep convolutional neural networks

URL:http://ieeexplore.ieee.org/document/7344669/

1. Improvement of speech emotion recognition with neural network classifier by using speech spectrogram

URL:http://ieeexplore.ieee.org/document/7314180/

1. PyImageSearch by Adrian Rosebrock , provides a crash course for OpenCV3.

URL:<http://www.pyimagesearch.com/free-opencv-crash-course/>