

**Masters Project Final Report**

**December 2015**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Project Title** |  | | | |
| **Student Name** |  | | | |
| **Registration No.**  **& Index No.** |  | | | |
| **Supervisor’s Name** | Mr. D.A.S Atukorale | | | |
| **Please Circle the appropriate** | **Master’s Program** | | **Type** | |
| **MIT** | **MCS** | **Research** | **Implementation** |

|  |
| --- |
| **For Office Use ONLY** |
|  |
|  |

Title of the project

**M. Mayura Prakash Wijeyaratne**

**2016**

## 

The title of the project

**A dissertation submitted for the Degree of Master of Computer Science**

**M. Mayura Prakash Wijeyaratne**

**University of Colombo School of Computing**

**2016**



DECLARATION

The thesis is my original work and has not been submitted previously for a degree at this or any other university/institute.

To the best of my knowledge it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: Mudiyanselage Mayura Prakash Wijeyaratne

Registration Number: 2012MCS068

Index Number: 12440681

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: Date: 30/12/2015

This is to certify that this thesis is based on the work of

Mr. Mudiyanselage Mayura Prakash Wijeyaratne

under my supervision. The thesis has been prepared according to the format stipulated and is of acceptable standard.

Certified by:

Supervisor Name: Mr. D.A.S. Atukorale

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Signature: Date: 30/12/2015

ABSTRACT

ACKNOWLEDGEMENTS

## TABLE OF CONTENTS

DECLARATION iii

ABSTRACT iv

ACKNOWLEDGEMENTS v

LIST OF FIGURES viii

ABBRIVIATIONS ix

CHAPTER 1: INTRODUCTION 1

1.1. Motivation 2

1.2. Problem Statement 2

1.3. Aims and Objectives 3

1.4. Scope and Limitations 3

1.5. Expected Contribution 4

Summary 4

CHAPTER 2: BACKGROUND 5

Summary 5

CHAPTER 3: METHODOLOGY 6

3.1. Prototype architecture 6

3.2. Algorithms used to calculate facial features from facial feature points 7

3.3. Machine learning techniques for human emotion identification 12

Summary 13

CHAPTER 4: IMPLEMENTATION 14

4.1. Selection of the coding languages and libraries 14

4.2. Collection of videos to create a dataset 16

4.3. Creating a software application to extract feature points 18

4.4. Pre-processing of facial feature values 20

4.5. Training K-means clustering model 21

Summary 22

CHAPTER 5: EVALUATION & RESULTS 23

5.1. Discussion on the feedback from the users participated for data collection 23

CHAPTER 6: CONCLUSION & FUTURE WORK 24

REFERENCES 25

APPENDIX 1 : Facial feature calculation mathematical formula in python I

1.1. Implementation of the mathematical formulae I

1.2. Calculate the value for each feature IV

LIST OF FIGURES

Figure 3.1: Prototype architecture of the system 6

Figure 3.2: Facial feature points [7] 7

Equation 1: Gradient of eyebrows 8

Equation 2: Distance between eyebrows and eyes ( f2 ) 8

Equation 3: Normalisation factor 8

Equation 4: Area between eyebrows ( f3 ) 9

Equation 5: Area of eyes ( f4 ) 9

Equation 6: Vertical to horizontal ratio of eyes ( f5 ) 9

Equation 7: Area of the circumference of a mouth ( f6 ) 10

Equation 8: Area of inner circumference of a mouth ( f7 ) 10

Equation 9: Vertical to horizontal ratio of the circumference of a mouth ( f8 ) 10

Equation 10: Vertical to horizontal ratio of the inner circumference of a mouth ( f9 ) 11

Equation 11: Vertical position of the corner of a mouth ( f10 ) 11

Figure 4.1: Video capture application 17

Figure 4.2: Output image from Luxand API 19

Figure 4.3: Luxand API not detecting the face 19

ABBRIVIATIONS

HRV Heart Rate Variability

BVP Blood Volume Pulse

PD Variations of Pupil Diameter

GSR Galvanic Skin Response

ECG Electrocardiogram

SVM Support Vector Machine

LDA Linear Discriminant Analysis

FACS Facial Action Coding System

CHAPTER 1: INTRODUCTION

Deadline! This is the one word every employee dreads of hearing. In the modern day where deadlines are inabundance, it has made employees work harder than ever before. Richard Boyatzis, Ph.D. has shown through Neuroimaging that deadlines often limit thinking, can lead to worse decision-making and can increase stress levels [1]. “Stress isn’t always bad, though. Stress within your comfort zone can help you perform under pressure, motivate you to do your best, even keep you safe when danger looms. But when stress becomes overwhelming, it can damage your health, mood, relationships, and quality of life” [2].

What is stress?

Stress is body’s way of responding to any threat by releasing stress hormones, including adrenaline to arouse to make an emergency reaction. This is also called “fight or flight”, this helps you to stay focused energetic and alert [2].

Why do we need stress at a certain level and why is it harmful if it you allow it to overwhelm you?

While, in emergency situations stress can save your life, however having stress all the time and more than your comfort zone will actually be detrimental to your mind and body [2]. Heart disease, sleep problems, depression, weight problems are some harmful effects of having stress for a long period of time at a level than a person can handle. "Over time, if you're constantly in fight-or-flight, if your heart muscles and valves are awash in the epinephrine, it causes changes in the arteries and in the way that cells are able to regenerate,” [3]. Due to the above-mentioned reasons, it is very important for people to maintain their stress level at healthy levels and not be stressful for long periods of time.

Nowadays many professionals interact with computers to do their work. Hence if the computer can detect stress of the employee while they work on the computer it will be very helpful to them.

If we could capture the facial features through the computer and monitor it over time, we could measure the stress level of the user. This would enable the system/computer to indicate to the user if the stress level is above normal healthy levels.

In the field of medical science there are numerous devices that could be used to detect human physiological features. A few of them are Heart Rate Variability (HRV), Blood Volume Pulse (BVP), Variations of Pupil Diameter (PD), Galvanic Skin Response (GSR), Fingertip skin temperature. These equipment need to be attached to the human body in order to gather information about the human body. Sun et.al in there research “Activity-aware Mental Stress Detection Using Physiological Sensors” have used Electrocardiogram (ECG), GSR, and accelerometer to gather physiological measurements to derive mental stress classification [13].

With the help of computer peripherals, we could gather some of the human physiological data in a non-invasive manner. Keyboard typing patterns, webcam footage of user/employee, variable pupil diameter using webcam are some of the better non-invasive techniques to measure stress.

In this thesis, I aim to use non-invasive physiological data gathering techniques to collect data from employees who are working in front of the computer for an extended period of time. With the collected data, I attempt to quantify the stress level of each user and indicate to the user if the stress level is rising with time.

1.1. Motivation

In today’s society stress has become a very big problem. People are getting many stress related illnesses due to working excessive amount of time under stress. In many of the professions the employee is more engaged with the computer to do their day to day work. Therefore the author was motivated to find the stress level of a employee without any interfering devices and to alert the user when the stress level is too high.

1.2. Problem Statement

Employees have so many tight deadlines to meet in their workplace. They are under constant stress to deliver the work on time. Therefore employees are working under duress for a long period of time and thereby are open to stress related illnesses easily. This has become a major problem in the current society.

In these workplaces, employees are often more engaged with computers to achieve their tight deadlines. They interact more time with the computer during their work hours. Author wants explore the possibility of gathering employee physiological data from a web camera fixed on the computer, and using data gathered the possibility of assessing the state of stress the user is in.

1.3. Aims and Objectives

The aims and objectives of this research is

1. Model stress index of an employee through non-invasive methods of physiological data gathering. Evaluate the stress index to find the fluctuations in stress over time
2. Investigate and review existing literature relating to non-invasive forms of data gathering, and stress related researches.
3. Using different machine learning techniques to find out what method would yield the better results in finding the person’s stress index.
4. Evaluate the research findings and conclude whether this method could be used the measure the stress level of a person effectively.

1.4. Scope and Limitations

The video footage used for this research comes from one workplace, and all the professionals are IT professionals. This is mainly due to the fact that, it was difficult to find any publicly available datasets with videos of professionals working in front of a computer.

For the research component, the data was collected as the video recordings. Using standard machine learning techniques, part of the videos would be used for training while the other part would be used for testing purposes. Even though in this research prior recorded videos are used, it could be extended for live video capturing and displaying the level of stress to the user as future work.

To detect stress accurately precise and real-time information is needed on the individual which could be provided by GSR and HRV [28], however since this research focus non-invasive technologies, it can only give an indication of the individual is suffering from stress.

1.5. Expected Contribution

This research is expected to contribute to the field of computer science in many ways.

Firstly this research will add knowledge about different ways of identifying emotions of a person’s face through machine learning. The research component will be to calculate the stress index through the results that is achieved through emotion identification.

Summary

Stress has become a major problem in the society today. Most of the individuals encounter stress through frequently working with tough deadlines at work. Many of the professional nowadays engage with the computer to do their daily work, therefore identifying if a person is stressed through non-invasive methods and notifying them of it will enable them to take necessary actions to reduce their stress.

CHAPTER 2: BACKGROUND

Researches on stress detection using computers with the help of other devices have been done from the early 2000s. In 2006 a research with the title “Stress recognition using non-invasive technology” by Jing Zhai and Armando Barreto [4] has been done using invasive technologies like BVP, GSR, PD and Skin Temperature. The above mentioned technologies are could be categorised as minimal invasive ways of stress recognition. This is because the technology today has developed so much that we could have much more non-invasive methods of recognising stress. Some of those non-invasive techniques in the modern day are key stroke dynamics and pattern variations [3], mouse track movements, web cam footage.

In a research to measure stress on e-learning students [5], the information acquired from key strokes and mouse clicks. Some of the sources of information gathered in this research comprise of click accuracy, click duration, mouse movement and also key strokes. In this research they have made an observation that if the student is stressed backspace key and right shift key will pressed more often.

The above researches are minimal invasive and non-invasive methods of detecting stress. There have been researches done to find out if players of a particular game develop stress. One such research is “PokerMetrics: Stress and Lie Detection through Non-Invasive Physiological Sensing”[5]. In this research too, they have used minimal invasive stress recognition methods such as skin conductance peaks and HRV and non-invasive method of voice pitch variation in their research to find if poker players develop stress.

The author’s research is to recognise stress in a working environment with webcam footage. There have been researches done to recognise the emotions of people through the video footage. In the research “Unsupervised Emotional Scene Detection for Lifelog Video Retrieval Based on Gaussian Mixture Model (2013)” [6]

Summary

CHAPTER 3: METHODOLOGY

In this chapter the aspects such as design assumptions relating to the proof of concept, prototype architecture and discussion on the process flow will be featured. There will also be a critical analysis of the problem under investigation.

3.1. Prototype architecture

The Figure (3.1) illustrates the prototype architecture of the system.

Dataset of videos

Choose a video

Extract still pictures

Extract facial feature points

Output feature points

Calculate facial features for learning emotions

Screenshots

Cartesian points

Feature based clustering

Identified emotions

Detected stress patterns

Figure 3.1: Prototype architecture of the system

Figure 1 illustrates the overview of the prototype architecture of the system. From the set of videos, a single video is chosen at each time. Then in order to extract the facial feature points of the face, still pictures are considered and from each of still pictures facial markers/points are extracted as Cartesian points. These points are then used to calculate facial features, which will be the fed in as features for modelling machine learning based clustering component. Using clustering algorithm, the emotions will be identified, and thus detecting stress of a person.

3.2. Algorithms used to calculate facial features from facial feature points

Nomiya et.al has defined ten types of facial features using the facial feature points in order to detect discriminative movement of facial feature points in the appearance of various facial expressions [7].

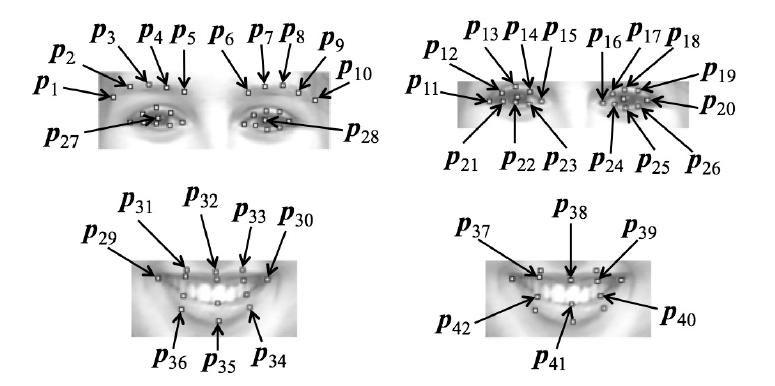
The facial feature points used for these facial feature points are show in Figure (3.2).

Figure 3.2: Facial feature points [7]

Using these facial feature points the following facial features are calculated.

Gradient of eyebrows (*f1*)

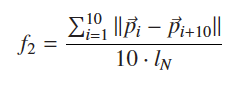
This feature value is based on the gradients al and ar of the two lines obtained from facial feature points on left and right eyebrows using least squares. This feature value is obtained through Equation (1).

al - gradient of points p1, p2, p3, p4 and p5

ar - gradient of points p6, p7, p8, p9 and p10

Equation 1: Gradient of eyebrows

Distance between eyebrows and eyes ( *f2* )

Using the mean distance between the facial feature points on eyebrows and those on the upper side of eyes, the value of this feature is obtained through Equation (2).

Equation 2: Distance between eyebrows and eyes ( *f2* )

Here, lN is a normalisation factor for the difference of the size of a face. It is defined as the distance between the centre points of left and right eyes, that is,

Equation 3: Normalisation factor

p27 and p28 being points 27 and 28.

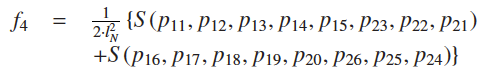
Area between eyebrows ( *f3* )

This feature value is given by Equation (4) as the area formed by connecting four facial feature points p5, p6, p16 and p15 located at the inner corners of eyebrows and eyes.

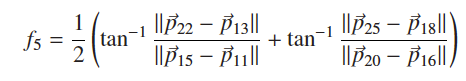
Here, S is the area of a polygon formed by connecting points p5, p6, p16 and p15.

Equation 4: Area between eyebrows ( *f3* )

Area of eyes ( *f4* )

By normalising the area of left and right eyes represented by two octagons, this feature value is defined by Equation (5).

Equation 5: Area of eyes ( *f4* )

Vertical to horizontal ratio of eyes ( *f5* )

Equation 6: Vertical to horizontal ratio of eyes ( *f5* )

Based on the ratio of the distance between top and bottom points to the distance between left and right points on an eye, this feature value is defined by Equation (6) .

Area of the circumference of a mouth ( *f6* )

This feature value is defined by Equation (7) as the area of an octagon formed by connecting eight facial feature points located on the circumference of a mouth.

Equation 7: Area of the circumference of a mouth ( *f6* )

Area of inner circumference of a mouth ( *f7* )

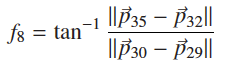
Similar to the sixth feature value, this feature value is defined by Equation (8) as the area of an octagon formed by connecting eight facial feature points located on the inner circumference of a mouth.

Equation 8: Area of inner circumference of a mouth ( *f7* )



Vertical to horizontal ratio of the circumference of a mouth ( *f8* )

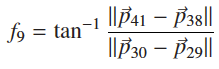
Equation 9: Vertical to horizontal ratio of the circumference of a mouth ( *f8* )



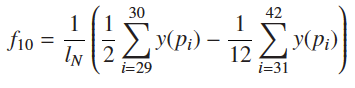
Based on the ratio of the distance between top and bottom points to the distance between left and right points on the circumference of a mouth, this feature value is defined by Equation (9) .

Vertical to horizontal ratio of the inner circumference of a mouth ( *f9* )

Equation 10: Vertical to horizontal ratio of the inner circumference of a mouth ( *f9* )

Similar to the eighth feature value, this feature value is defined by Equation (9) based on the ratio of the distance between top and bottom points to the distance between left and right points on the inner circumference of a mouth.

Vertical position of the corner of a mouth ( *f10* )



Equation 11: Vertical position of the corner of a mouth ( *f10* )

This feature value represents how high the position of the corner of a mouth is. It is defined by Equation (11).

Here, y (p) is the y -coordinate of a facial feature point p . If the mean value of the y -coordinate of the facial feature points on the corner of a mouth is larger than that of the other facial feature points on a mouth, *f10* becomes positive. Thus, a larger value of *f10* represents a higher position of the corner of a mouth.

All of the above equations were taken from Nomiya et.al [7] research paper on “Unsupervised Emotional Scene Detection for Lifelog Video Retrieval Based on Gaussian Mixture Model”. Since these are complex mathematical formula, it is quite hard to implement them in high level coding language like java, therefore python was used to convert these mathematical formula to computer language. The implementation of these mathematical formulae are illustrated in APPENDIX 1 of this thesis.

3.3. Machine learning techniques for human emotion identification

There are multiple techniques of training a machine to identify human facial emotions. Shan et.al in their research “Facial expression recognition based on Local Binary Patterns” mentions they have examined different machine learning methods, including template matching, Support Vector Machine (SVM), Linear Discriminant Analysis (LDA) and the linear programming technique, to perform facial expression recognition using Local Binary Patterns features [23]. They have conducted experiments on the Cohn–Kanade database, in which the images have been given a label. “Each image sequence begins with a neutral expression and proceeds to a peak expression. The peak expression for each sequence in fully Facial Action Coding System (FACS) (Ekman, Friesen, & Hager, 2002; Ekman & Friesen, 1979) coded and given an emotion label”[24]. These learning is widely known as “Classification”.

Like mentioned above, template matching, SVM, LDA and linear programming techniques, use labeled datasets to train a machine to recognise facial expressions. However, there are other datasets, which isn’t labeled can also be used in training to identify facial emotions. Nomiya et.al [7] uses an unlabelled video dataset, which are Lifelog Videos to recognise facial expressions. They used facial expression recognition using by a clustering algorithm based on Gaussian mixture model using the feature vectors [7].

These techniques are categorised as “unsupervised learning, mostly known as clustering or exploratory data analysis, no labeled data are available”[26]. Similarly the current research also has unlabelled data, therefore unsupervised classification is the best method of classifying emotions of humans.

To do unsupervised classification many different clustering algorithms can be used. Wunsch and Xu [26] have surveyed many of these clustering algorithms. In the conclusion of this survey, they have found out that “There is no clustering algorithm that can be universally used to solve all problems” [26]. The article goes on to discuss that “it is not accurate to say ‘best’ in the context of clustering algorithms” [26]. In this dissertation, given the context of the problem with unlabeled emotions in the video data, I used unsupervised learning method to identify different emotions expressed by users while performing their tasks at work environment. I selected a popular clustering algorithm known as k-means clustering [27] due to its robustness and ease of use as a clustering algorithm in the experiments to group user facial images with similar emotions such as happiness, sadness, and neutral.

Summary

CHAPTER 4: IMPLEMENTATION

In order to implement the methodology mentioned in the previous chapter, it is necessary to look at the technologies available and analyse which of those are relevant for the project.

4.1. Selection of the coding languages and libraries

There are many computer languages available to implement a design concept. However there are some important factors that should be considered in selection of the most appropriate language to implement the design. Similarly there are many software libraries that would perform similar functionality, however many of the advanced libraries are proprietary while some others are free to use and a handful of proprietary libraries give evaluation keys for research purposes.

One of the essential parts in this research is to detect facial feature points from a video feed. There are many proprietary software libraries that could be used for extraction of facial features. Some of those software products are

EmoVu - Learning based emotion recognition software that could read the emotions of people [14].

Emotient - On demand emotion analysis of videos using facial expression recognition [15].

nViso - 3D Facial Imaging software [16]

Even though the products mentioned above have advanced technologies, it is not possible to use for research purposes since they are proprietary. Therefore to work on this project, a free API should be used. There are only a handful of free API libraries to choose from to recognise facial features.

OpenCV - This open source library provides functionality to do many projects in computer vision [17]. However using this library, the facial feature identification will have to be coded and it doesn’t come as off the shelf functionality. This library is also not very easy to use unlike the other libraries.

flandmark - Open source C library (with interface to MATLAB) implementing a facial landmark detector in static images[18].

There are few proprietary software that gives an evaluation key for research projects. Evaluating several of such software, I chose Luxand face recognition API [8] for detection of facial features.

Luxand face recognition API also known as FaceSDK is a high-performance, multi-platform face recognition, identification and facial feature detection solution [8]. The main reasons why FaceSDK was picked over other software are that,

* 1. They provide an evaluation key for research purposes.
  2. It detects 66 facial features in still pictures and videos.
  3. It is compatible with many different coding languages such as Visual C++, C#, Objective C, VB, Java and Delphi
  4. It is compatible with many operating systems such as Microsoft Windows, MacOS X, Linux, iOS, Android.

Machine learning component of the project is one of the most important components of the project. It is best to use a software package that exists in the market rather than implementing it since it is not the major focus of this project. There are many such packages available. Some of the packages considered are

MATLAB - Proprietary software which has many different components including a component for machine learning. It has many different modules such as classification, regression and clustering [19]. It is a very powerful too, however it is not practical to use this tool as it is proprietary.

GoLearn - GoLearn is a 'batteries included' machine learning library for Go [20]. This does seem to have proper documentation and how to use it therefore even though it is an open source project, it not easy to work with this tool.

The package that works the best for machine learning component of this project is scikit-learn package. There are a number of reasons why scikit-learn machine learning package was chosen over other packages. Firstly it’s a free software package. Secondly it has many different modules such as classification, regression, clustering, preprocessing and effective to write scripts using those libraries. Thirdly the documentation of the software package is extensive, thereby making it easier to work with it. Most importantly scikit-learn is a python machine learning package which is where most of the calculations are done in this project, thereby sending data to scikit-learn is easy.

Python is considered as the one of the best languages for crunching, the best bring ‘R’ [9]. Python is also easier to learn than R. Another reason to choose python is that to convert the facial feature points extracted by Luxand API need to be used to calculate facial features. Nomiya et.al has defined the following ten types of facial features using the facial feature points in order to detect discriminative movement of facial feature points in the appearance of various facial expressions [7]. The paper discusses 10 mathematical formula to calculate each of the 10 features. Considering the requirement, python was considered to be the best coding language for the data preprocessing and machine learning part of the project.

Another coding language needed to be considered to implement Luxand API to extract the facial feature points it offered as it doesn’t support python. In this section of the project, it focuses on collection of data rather than the actual research part, therefore it is necessary to implement it speedily. Hence Java was selected as the coding language since it is more familiar to the author.

As the operating system to implement this project, author decided to use Mac OS. The reason for this is because it is the operating system that is readily available and more comfortable in use for the author.

4.2. Collection of videos to create a dataset

There is a multitude of publicly available datasets that could be used for various data mining projects [10] [11]. Since, the specific area that the author is considering in this thesis is very specific to work related stress with webcam footage, it was difficult to find any publicly available datasets. Unfortunately, the one database that is quite inline with what the author is working on, Lifelog videos [7] has been discontinued now. Therefore an application had to be implemented in order to gather the necessary webcam feed videos of people working on the computer, to work on the research.

This video capture application is coded using java and used a generic webcam Java API, ’Webcam capture’ [12]. The application uses the webcam device connected to the computer and it captures video feed of the user. The figure 4.1 shows a screenshot of the video capture application.

Figure 4.1: Video capture application

6 employees of an IT company were given the application to capture the webcam feed during the course of the day. The participants wasn’t given any particular instructions, other than how to run and stop it. Around 6 hours of video was captured of each of the participant. The webcam feed was saved in .ts format as it doesn’t take much space in the participants computer. Thereby made a dataset of videos of people working in front of the computer.

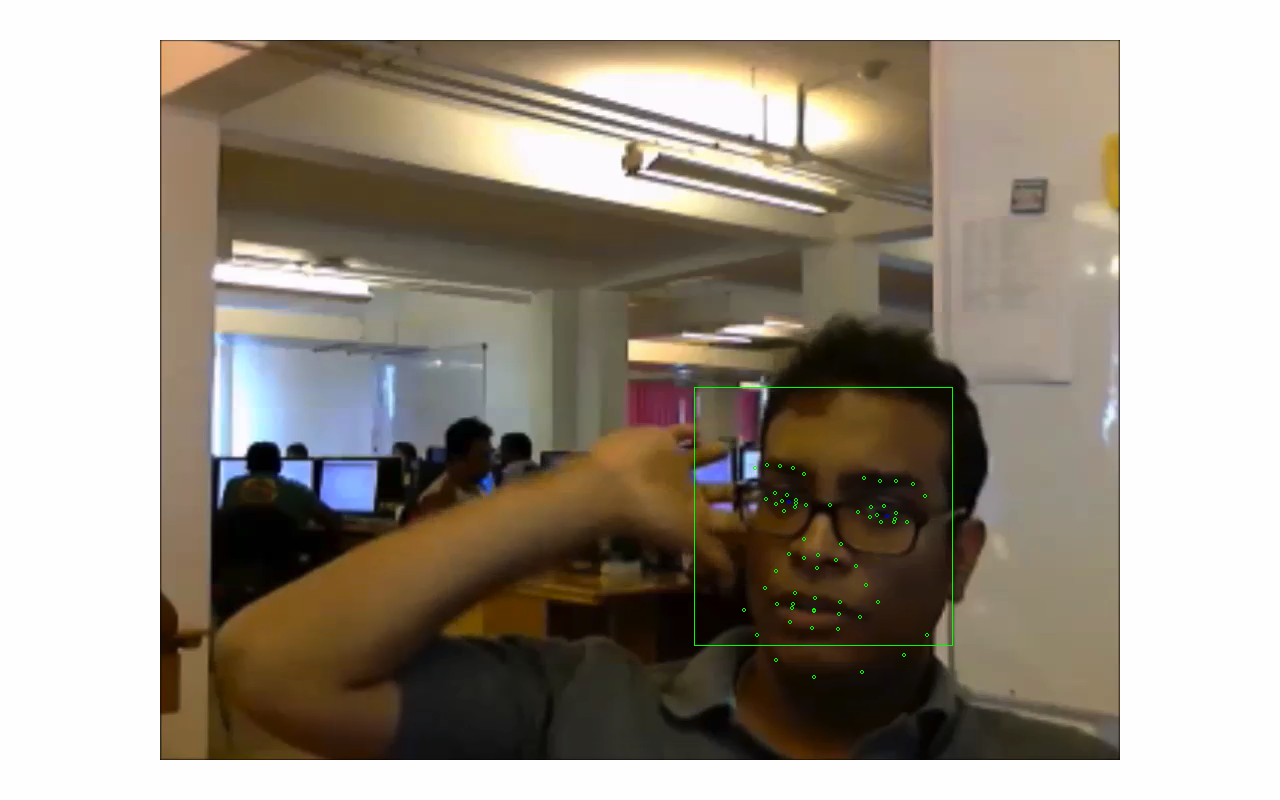
4.3. Creating a software application to extract feature points

Now that there is a dataset to work with, it is time to use the videos to gather the facial feature points. As mention earlier Luxand API was chosen as the library to achieve this.

A software application needed to be coded to run the videos in the dataset, use the Luxand API to gather facial feature points and save it in a format to be used in python calculations. JavaFX MediaPlayer API was used in creating an application to run the videos in the dataset. However, JavaFX MediaPlayer had a limitation of not being able to play certain formats and .ts format was one of them [21]. Therefore the .ts file format needed to be converted to a format which could be played with JavaFX MediaPlayer. The online site ZamZar [22] was used to convert the .ts format video to .mp4 format for JavaFX MediaPlayer to play them.

Once the videos could be played with JavaFX MediaPlayer API, it was apparent that Luxand API had a limitation too. “Detection of 66 facial features, smooth facial feature tracking in video.” [8] did work in Mac OS while it worked on Windows OS. Therefore an alternative needed to be found.

The feature of facial features detection for still pictures was working in Mac OS, hence to use that feature, using java.awt.Robot functionality took a screenshot every second the video is running. Each of these screenshots is a ‘Frame’ in this thesis. Using the still pictures, the software application was able to find the facial feature points of the participants face and Image 2 is such a screenshot taken by the application.

Figure 4.2: Output image from Luxand API

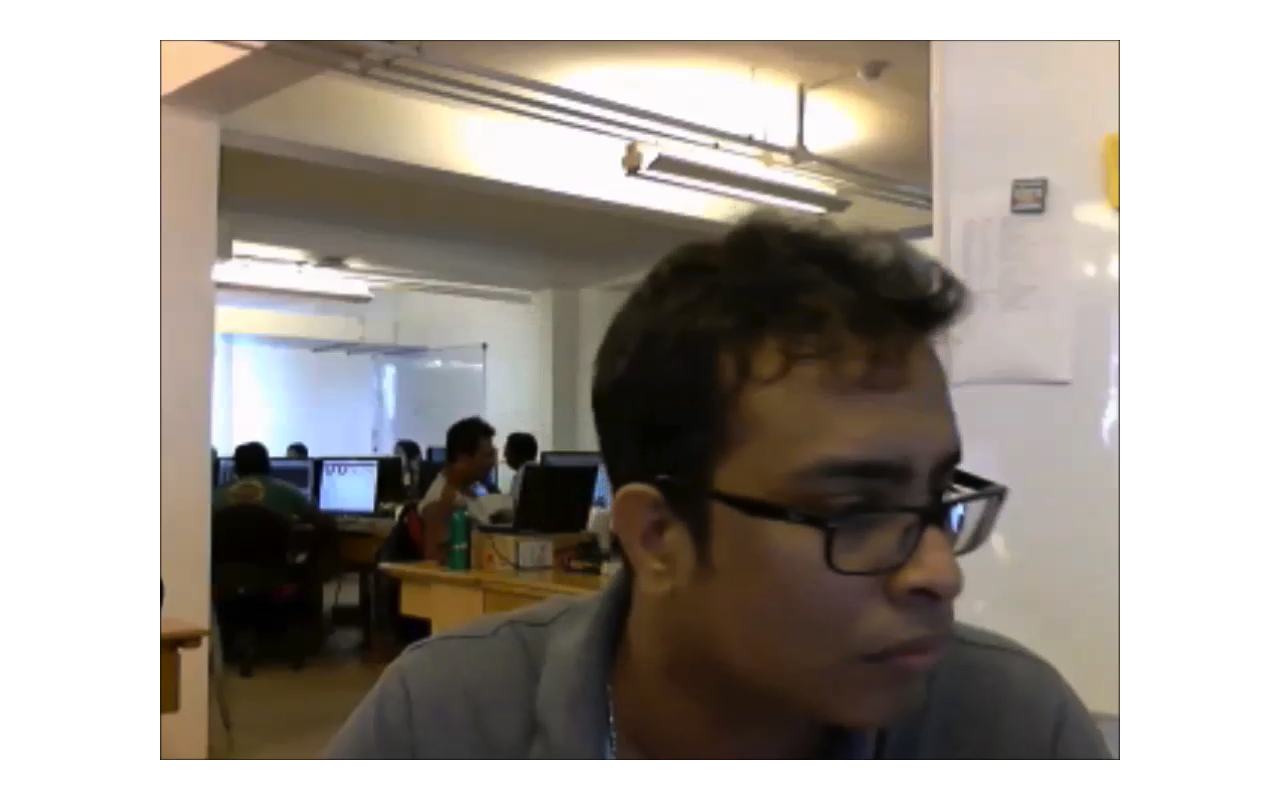
There are frames where no face could be detected by the Luxand API. Below image shows such a situation.

Figure 4.3: Luxand API not detecting the face

The facial feature points of the participant is extracted as Cartesian points from Luxand API. The software application that was created, puts the points extracted to a .csv file. The some of the content of the .csv file is shown below.

FrameNumber,FeatureNumber,FeatureXAxis,FeatureYAxis

1,0,715,482

1,1,801,467

1,2,762,520

1,3,732,577

1,4,813,560

1,5,695,588

1,6,861,562

1,7,719,604

‘FrameNumber’ is the frame from which a face was successfully identified and ‘FeatureNumber’ ranges from 0 to 65 one for each detected facial feature point of Luxand API. ‘FeatureXAxis’ and ‘FeatureYAxis’ are the X and Y Cartesian points of the image.

As per the prototype architecture shown in Figure (3.1), once the feature points are extracted, facial feature values need to be calculated. In order to calculate the facial feature values, the 10 mathematical formula mentioned as equation 1 to 11 need to be coded in python. The code for these mathematical formula are attached in APPENDIX 1. For each facial feature a python method has been implemented for the ease of use. The output from these methods are then assigned to a pandas data frame.

4.4. Pre-processing of facial feature values

Having decided to use k-means algorithm for the unsupervised classification section of the research, there are some pre-processing that should be done in order to the clustering algorithm to perform at it’s best. In clustering analyses, standardisation may be especially crucial in order to compare similarities between features based on certain distance measures [29]. Therefore standardise, is to bring values of features or attributes from different dynamic range into a specific range [30]. There are different methods of standardisation. Three such standardisation methods are Z-score analysis, Decimal scaling analysis and Min-max analysis. In “Standardization and Its Effects on K-Means Clustering Algorithm” [30] all three of them have been analysed, and as its conclusion, they have mentioned that Z-score as “the most powerful method that will give more accurate and efficient result among the three methods in K-means clustering algorithm” [30].

In my implementation I tried out two of the above mentioned standardisation methods as they are implemented in sklearn package. The results of the k-means clustering differed only by a small amount which would be discussed further in the next chapter.

One of the common problems in machine learning is overfitting of data. This is when the model or the algorithm fits the data too well [31]. To take measures to check if the data does not overfit, the data used for this research was split in to two parts. 75 percent of the data is used for training of K-means model while 25 percent of the data are used for testing purposes. Through sklearn package data could be easily split into two such parts.

X\_train, X\_test = train\_test\_split(df1\_normalize, test\_size=0.25, random\_state=42)

The above code snippet shows the code line in which the splitting of the data happens. df1\_normalize variable holds the data which has been standardised by Z-score analysis. test\_size argument defines the percentage of data to be split as the test data sample. random\_state is a generator in order to initialise the centres. After the splitting of the data X\_train will hold 75 percent of data from df1\_normalize, while the rest of the data will be stored in X\_test variable.

4.5. Training K-means clustering model

Sklearn package has easy implementation of training a model through K-means clustering algorithm. To train K-means model, the training data (X\_train) was used. The below code snippet shows the training of the K-means.

From the parameter n\_clusters the number of clusters the data to be grouped is defined, while n\_init is the number of times the k-means algorithm will be run with different centroid seeds. max\_iter parameter gives the maximum number of iterations of the k-means algorithm for a single run.

trained\_model = KMeans(n\_clusters=2,n\_init=10,max\_iter=750).fit(X\_train)

To find the best fit for the Kmeans model, the numbers of the above 3 parameters was changed and evaluated. The videos available in the dataset aren’t prearranged dataset to include lot of data with many different emotions. Videos in the dataset are recorded in real-life scenario and doesn’t include many different emotions, therefore it was best to train the K-means model for 2 clusters, as then it would have better chance of classifying the data into frames with happy emotions and frames with neutral faces. Therefore the parameter n\_clusters was kept to 2.

The number of data that is available isn’t a very large number, therefore having the K-means model run with many different centroid seeds isn’t ideal. Therefore n\_init parameter was kept to 10.

The parameter max\_iter was fiddled with the most. The default number of max iterations is 300 however, even with 500 max iterations the results varied with each run of K-means. Therefore, after much deliberations the max number of iterations was assigned with 750.

Summary

CHAPTER 5: EVALUATION & RESULTS

This chapter focuses on illustrating the results of the implementation chapter and evaluating the final product.

5.1. Discussion on the feedback from the users participated for data collection

CHAPTER 6: CONCLUSION & FUTURE WORK

REFERENCES

[1] K. Sullivan, 'Deadlines, stress cause leaders to make poor choices', FierceHealthcare, 2015. [Online]. Available: http://www.fiercehealthcare.com/story/neuroimaging-breaks-down-executive-leadership-decision-making/2014-04-29. [Accessed: 07- Dec- 2015].

[2] Helpguide.org, 'Stress Symptoms, Signs, & Causes - Helpguide.org', 2015. [Online]. Available: http://www.helpguide.org/articles/stress/stress-symptoms-causes-and-effects.htm. [Accessed: 31- Oct- 2015].

[3] msnbc.com, 'Can stress actually be good for you?', 2006. [Online]. Available: http://www.nbcnews.com/id/15818153/ns/health-mental\_health/t/can-stress-actually-be-good-you/#.VjQqI64rKDU. [Accessed: 31- Oct- 2015].

[4] Jing Zhai and Armando Barreto, “Stress Recognition Using Non-invasive Technology”, Electrical and Computer Engineering Department, Biomedical Engineering Department Florida International University.

[5] Suranga D.W. Gunawardhane, Pasan M. De Silva, Dayan S.B. Kulathunga, Shiromi M.K.D. Arunatileka, “Non invasive human stress detection using key stroke dynamics and pattern variations”, University of Colombo School of Computing .

[6] Michael Sung and Alex (Sandy) Pentland, “PokerMetrics: Stress and Lie Detection through Non-Invasive Physiological Sensing”, MIT Media Laboratory, Human Dynamics Group.

[7] Hiroki Nomiya, Atsushi Morikuni and Teruhisa Hochin, “Unsupervised Emotional Scene Detection for Lifelog Video Retrieval Based on Gaussian Mixture Model” in 17th International Conference in Knowledge Based and Intelligent Information and Engineering Systems, 2003, pp. 375 – 384

Luxand FaceSDK. Luxand, Inc., 2015.

[8] A. Nicolaou and A. Nicolaou, 'The 9 Best Languages For Crunching Data', Fast Company, 2014. [Online]. Available: http://www.fastcompany.com/3030716/the-9-best-languages-for-crunching-data. [Accessed: 02- Nov- 2015].

[9] Archive.ics.uci.edu, 'UCI Machine Learning Repository', 2015. [Online]. Available: http://archive.ics.uci.edu/ml/. [Accessed: 02- Nov- 2015].

[10] Inf.ed.ac.uk, 'Datasets for Data Mining', 2015. [Online]. Available: http://www.inf.ed.ac.uk/teaching/courses/dme/html/datasets0405.html. [Accessed: 02- Nov- 2015].

[11] Webcam-capture.sarxos.pl, 'Webcam Capture in Java', 2015. [Online]. Available: http://webcam-capture.sarxos.pl/. [Accessed: 02- Nov- 2015].

[12] F. Sun, C. Kuo, H. Cheng, S. Buthpitiya, P. Collins and M. Griss, 'Activity-aware Mental Stress Detection Using Physiological Sensors', Carnegie Mellon University, 2010.

[13] EmoVu, 'EmoVu emotion recognition software', 2015. [Online]. Available: http://emovu.com/e/. [Accessed: 11- Dec- 2015].

[14] Emotient, 'Emotient Analytics - Emotient', 2015. [Online]. Available: http://www.emotient.com/products/emotient-analytics/. [Accessed: 11- Dec- 2015].

[15] Nviso.ch, '3D Facial Imaging Software and Emotion Analytics Cloud Service | nViso', 2015. [Online]. Available: http://www.nviso.ch/. [Accessed: 11- Dec- 2015].

[16] Opencv.org, 'OpenCV | OpenCV', 2015. [Online]. Available: http://opencv.org/. [Accessed: 12- Dec- 2015].

[17] Cmp.felk.cvut.cz, 'flandmark - open-source implementation of facial landmark detector', 2015. [Online]. Available: http://cmp.felk.cvut.cz/~uricamic/flandmark/. [Accessed: 12- Dec- 2015].

[18] In.mathworks.com, 'Machine Learning with MATLAB', 2015. [Online]. Available: http://in.mathworks.com/solutions/machine-learning/?requestedDomain=www.mathworks.com. [Accessed: 13- Dec- 2015].

[19] GitHub, 'sjwhitworth/golearn', 2015. [Online]. Available: https://github.com/sjwhitworth/golearn. [Accessed: 13- Dec- 2015].

[20] Docs.oracle.com, "javafx.scene.media (JavaFX 2.2)", 2015. [Online]. Available: https://docs.oracle.com/javafx/2/api/javafx/scene/media/package-summary.html. [Accessed: 20- Dec- 2015].

[21] Zamzar.com, "Zamzar - video converter, audio converter, image converter, eBook converter", 2015. [Online]. Available: http://www.zamzar.com/. [Accessed: 20- Dec- 2015].

[22] C. Shan, S. Gong and P. McOwan, "Facial expression recognition based on Local Binary Patterns: A comprehensive study", Image and Vision Computing, vol. 27, no. 6, pp. 803-816, 2009.

[23] Pitt.edu, "The Affect Analysis Group at Pittsburgh", 2015. [Online]. Available: http://www.pitt.edu/~emotion/ck-spread.htm. [Accessed: 23- Dec- 2015].

[24] M. Weber, "Unsupervised Learning of Models for Object Recognition", Doctor of Philosophy, California Institute of Technology Pasadena, California, 2000.

[25] Scholarsmine.mst.edu, "Survey of clustering algorithms", 2005. [Online]. Available: http://scholarsmine.mst.edu/cgi/viewcontent.cgi?article=2429&context=faculty\_work. [Accessed: 24- Dec- 2015].

[26] A. Jain, "Data clustering: 50 years beyond K-means", Pattern Recognition Letters, vol. 31, no. 8, pp. 651-666, 2010.

[27] Alberto De Santos, Carmen Sánchez-Avila, Javier Guerra-Casanova and Gonzalo Bailador-Del Pozo (2011).Real-Time Stress Detection by Means of Physiological Signals, Recent Application in Biometrics, Dr. Jucheng Yang (Ed.), ISBN: 978-953-307-488-7, InTech, Available from: <http://www.intechopen.com/books/recentapplication-in-biometrics/hand-biometrics-in-mobile-devices>

[28] Sebastianraschka.com, "Feature Scaling", 2015. [Online]. Available: http://sebastianraschka.com/Articles/2014\_about\_feature\_scaling.html#z-score-standardization-or-min-max-scaling. [Accessed: 28- Dec- 2015].

[29] I. Bin Mohamad and D. Usman, "Standardization and Its Effects on K-Means Clustering Algorithm", Research Journal of Applied Sciences, Engineering and Technology, vol. 6, no. 7, pp. 3299-3303, 2013.

[30] E. Statistician, "Machine Learning Lesson of the Day – Overfitting and Underfitting | StatsBlogs.com | All About Statistics", Statsblogs.com, 2014. [Online]. Available: http://www.statsblogs.com/2014/03/20/machine-learning-lesson-of-the-day-overfitting-and-underfitting/. [Accessed: 28- Dec- 2015].

APPENDIX 1 : Facial feature calculation mathematical formula in python

1.1. Implementation of the mathematical formulae

def leastSq(x,y): # Method to calculate the least square of points

A = np.vstack([x, np.ones(len(x))]).T

m, c = np.linalg.lstsq(A, y)[0]

return m

def gradEyes(x,y,x1,y1): # Gradient of eyebrows

leftEye = leastSq(x,y)

rightEye = leastSq(x1,y1)

return (leftEye-rightEye)/2

def normalizeFactor(point27, point28): #Facial normalisation factor calculate

norFac = scsp.distance.cdist(point27, point28, 'euclidean') # The euclidean distance between right eye and left eye

return norFac

def distEyesEyebrows(points, lN): #Distance between the eyes and the eyebrows

totalEucDis = 0

for i in range(len(points)-10):

eucDis = scsp.distance.cdist(points[i], points[i+10], 'euclidean')

totalEucDis += eucDis

distEyes = totalEucDis/(10\*lN)

return distEyes

def areaOfTriangle(point1, point2, point3): # Calculate area of a triangle to calculate the area of a polygon

area = abs((point1[0][0]\*(point2[0][1]-point3[0][1]) + point2[0][0]\*(point3[0][1]-point1[0][1]) + point3[0][0]\*(point1[0][1]-point2[0][1])) / 2)

return area

def areaBetweenEyes(point5, point6, point16, point15, lN): # Calculate area of a 4 point polygon

tri1 = areaOfTriangle(point5,point6,point16)

tri2 = areaOfTriangle(point5,point16,point15)

areaEyes = (tri1 + tri2)/ lN\*\*2

return areaEyes

def areaofOctogon(point1,point2,point3,point4,point5,point6,point7,point8): # Calculate area of a octogon with triangles

tri1 = areaOfTriangle(point1,point2,point3)

tri2 = areaOfTriangle(point1,point3,point4)

tri3 = areaOfTriangle(point1,point4,point5)

tri4 = areaOfTriangle(point1,point5,point6)

tri5 = areaOfTriangle(point1,point6,point7)

tri6 = areaOfTriangle(point1,point7,point8)

area = tri1 + tri2 + tri3 + tri4 + tri5 + tri6

return area

def areaOfEyes(point11,point12,point13,point14,point15,point23,point22,point21,point16,point17,point18,point19,point20,point26,point25,point24,lN): #area of eyes using area of octogon

areaLeftEye = areaofOctogon(point11,point12,point13,point14,point15,point23,point22,point21)

areaRightEye = areaofOctogon(point16,point17,point18,point19,point20,point26,point25,point24)

areaEyesBoth = (areaLeftEye + areaRightEye) \* (1/lN\*\*2)

return areaEyesBoth

def vTHRofEyes(point22,point13,point15,point11,point25,point18,point20,point16): # vertical of horizontal ratio of eyes

distance1 = scsp.distance.cdist(point22, point13, 'euclidean')

distance2 = scsp.distance.cdist(point15, point11, 'euclidean')

distance3 = scsp.distance.cdist(point25, point18, 'euclidean')

distance4 = scsp.distance.cdist(point20, point16, 'euclidean')

arctan1 = np.arctan(distance1/distance2)

arctan2 = np.arctan(distance3/distance4)

feature5 = 0.5 \* (arctan1 + arctan2)

# print feature5

return feature5

def areaCircumOfMouth(point1,point2,point3,point4,point5,point6,point7,point8,lN): # Area of the circumference of a mouth

area = areaofOctogon(point1,point2,point3,point4,point5,point6,point7,point8)

feature6 = area / lN\*\*2

return feature6

def vTHRofCircMouth(p1,p2,p3,p4): #Vertical to horizontal ratio of the circumference of a mouth

distance1 = scsp.distance.cdist(p1, p2, 'euclidean')

distance2 = scsp.distance.cdist(p3, p4, 'euclidean')

f8 = np.arctan(distance1/distance2)

return f8

def vposOfMouth(p29,p30,p31,p32,p33,p34,p35,p36,p37,p38,p39,p40,p41,p42,lN): #Vertical position of the corner of a mouth

a1 = np.array([p29[0][1],p30[0][1]])

a2 = np.array([p31[0][1],p32[0][1],p33[0][1],p34[0][1],p35[0][1],p36[0][1],p37[0][1],p38[0][1],p39[0][1],p40[0][1],p41[0][1],p42[0][1]])

f10 = (1/lN) \* (np.mean(a1)-np.mean(a2))

return f10

1.2. Calculate the value for each feature

feature1 = gradEyes(x,y,x1,y1) # Calculate Gradient of eyebrows

feature2 = distEyesEyebrows(points1to20, lN) # Calculate Distance between eyebrows and eyes

feature3 = areaBetweenEyes(point5,point6,point15,point16,lN) # Area between eyebrows

feature4 = areaOfEyes(point11,point12,point13,point14,point15,point23,point22,point21,point16,point17,point18,point19,point20,point26,point25,point24,lN) # Calculate Area of eyes

feature5 = vTHRofEyes(point22,point13,point15,point11,point25,point18,point20,point16) #Calculate Vertical to horizontal ratio of eyes

feature6 = areaCircumOfMouth(point29,point31,point32,point33,point30,point34,point35,point36,lN) # Calculate Area of the circumference of a mouth

feature7 = areaCircumOfMouth(point29,point37,point38,point39,point30,point40,point41,point42,lN) #Calculate Area of inner circumference of a mouth

feature8 = vTHRofCircMouth(point35,point32,point30,point29) #Vertical to horizontal ratio of the circumference of a mouth

feature9 = vTHRofCircMouth(point41,point38,point30,point29) #Vertical to horizontal ratio of the inner circumference of a mouth

feature10 = vposOfMouth(point29,point30,point31,point32,point33,point34,point35,point36,point37,point38,point39,point40,point41,point42,lN) #Vertical position of the corner of a mouth