

**MOOD AND MOTIVATION DETECTION WITH FACIAL EMOTION RECOGNITION**

**A Thesis Submitted to the**

**Dokuz Eylül University, Department of Computer Engineering**

**In Partial Fulfillment of the Requirements for the Degree of B.Sc.**

**by**

**Vuslat YOLCU**

**İbrahim ERHAN**

**Advisor**

**Assoc.Prof.Dr. Semih UTKU**

**June, 2020**

**İZMİR**

**SENIOR PROJECT EXAMINATION RESULT FORM**

We have read the thesis entitled **“MOOD AND MOTIVATION DETECTION WITH FACIAL EMOTION RECOGNITION”** completed by **Vuslat YOLCU** and **İbrahim ERHAN** under advisor of **Assist. Prof. Dr. Semih UTKU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of B.Sc.

Advisor

Committee Member Committee Member

Prof. Dr. Alp KUT

Chair

Department of Computer Engineering

**MOOD AND MOTIVATION DETECTION WITH FACIAL EMOTION RECOGNITION**

**ABSTRACT**

Facial expressions have a great place in transmitting information to the other person. By a computer, the perception of human facial expressions consists of three steps detection of the face, feature extraction, and classification. Here, a series of images are automatically classified as 7 basic emotions. The implementation of algorithms here will contribute to areas of psychological analysis and increases the efficiency of human productivity in work areas. Knowing the workers' moods and working quality has a great impact on increasing productivity and efficiency. Machine learning algorithms and advancements in computational power give the development capability of these systems. The developed algorithms have been implemented with OpenCV, dlib, and python used as open source.

**MOOD AND MOTIVATION DETECTION WITH FACIAL EMOTION RECOGNITION**

**ÖZET**

Yüz ifadelerinin üç adımdan oluşur, bunlar yüzün tespit edilmesi, yüzün özelliklerinin çıkarılması ve yüzün sınıflandırılmasıdır. Gerçek zamanlı gelen veriler otomatik olarak 7 temel duygu içinde sınıflandırılmaktadır. Algoritmaların uygulanması psikolojik analiz alanlarında ve çalşma ortamında insanların verimliliğini arttıracak şekildedir. Çalışanların çalışma sırasındaki motivasyonunu ve ruh halini tespit etmek yapılan işteki verimliliği büyük oranda arttırmaktadır. Bu amaçla makine öğrenmesi algoritmaları ve hesaplama gücündeki gelişmeler sistemin tasarlanmasında büyük rol aldı. Geliştirilen algoritmalar OpenCV, dlib ve python kullanılarak uygulandı.

CONTENTS

**Page**

[ABSTRACT](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) III

[ÖZET](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) IV

[LIST OF FIGURES](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) VIII

[LIST OF TABLES](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) IX

[**CHAPTER ONE INTRODUCTION**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **1**

[1.1 Background Information](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 1

[1.2 Problem Definition](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 2

[1.3 Motivation/Related Work](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 2

[1.4 Goal/Contribution](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 3

[1.5 Project Scope](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 4

[1.6 Methodologies/Tools/Libraries](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 4

[1.6.1 Convolutional Neural Networks](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 4

[1.6.2 TensorFlow](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 5

[1.6.3 OpenCV](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 5

[1.6.4 Dlib](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 5

[**CHAPTER TWO LITERATURE REVIEW**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **6**

[2.1 Processing of System Parameters](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 6

[2.2 Feature Extraction](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 7

[2.3 Classification](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 8

[2.3.1 Convolutiona Neural Networks](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 8

[2.3.2 Support Vector Machine](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 9

[**CHAPTER THREE REQUIREMENT ENGINEERING**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **11**

[3.1 Functional Requirements](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 11

[3.1.1 Hardware Requirements](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 11

[3.1.2 Software Requirements](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 12

[3.2 Non-Functional Requirements](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 12

[**CHAPTER FOUR DESIGN**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **15**

[4.1 General Architectural View](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 15

[4.2 Entity Relationship Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 16

[4.3 Class Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 17

[4.4 Use Case Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 18

[4.5 Sequence Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 19

[4.6 Deployment Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 21

[**CHAPTER FIVE IMPLEMENTATION**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **22**

[5.1 Face Detection Implementation](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 22

[5.2 CNN Model Implementation for Emotion Recognition](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 24

[5.2.1 Tensorflow Keras API](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 24

[5.2.2 Model Architecture and Layers](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 25

[5.2.2.1 Layers and Parameters](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 25

[5.2.2.2 Compiling the Model](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 28

[5.2.2.3 Saving the Model Weights](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 29

[5.2.3 Model Evaluation and Results](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 30

[5.3 MongoDB Deployment](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 30

[5.3.1 MongoDB Architecture](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 30

[5.3.2 Node.js REST API Implementation](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 31

[5.3.3 Sending Face Features to MongoDB](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 31

[5.4 PostgreSQL Deployment](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 32

[5.4.1 Relational Database Architecture](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 32

[5.4.2 Node.js REST API Implementation](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 32

[5.4.2 Sending Labeled Classification Data](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 32

[5.5 Simulating the Data Flow and Database Connections](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 32

[**CHAPTER SIX TEST AND EXPERIMENTS**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **34**

[6.1 Experiments of Face Detection Algorithms](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 34

[6.2 Experiments of Facial Emotion Detection Algorithm](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 39

[**CHAPTER SEVEN CONCLUSION AND FUTURE WORKS**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **41**

[**REFERENCES**](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) **42**

LIST OF FIGURES

**Page**

[Figure 4.1 Overall System Architecture](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 15

[Figure 4.2 Entity Relationship Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 16

[Figure 4.3 Class Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 17

[Figure 4.4 Use Case Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 18

[Figure 4.5 Controller Side Sequence Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 19

[Figure 4.6 Server Side Sequence Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 20

[Figure 4.7 Deployment Diagram](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 21

[Figure 5.1 Landmark Demonstrations on Sample Image](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 23

[Figure 5.2 Detected Landmarks from Sample Image](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 23

[Figure 5.3 General Architecture of Model Layers](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 25

[Figure 5.4 Representaion of Input Image and Filter Size](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 26

[Figure 5.5 Illusturation of Convolution Processes](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 27

[Figure 5.6 Illusturation of Pooling Process](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 27

[Figure 5.7 Illusturation of MongoDB Architecture](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 30

[Figure 5.8 Dataflow Simulation Interface](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 33

[Figure 6.1 Activation Functions](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 39

LIST OF TABLES

**Page**

[Table 6.1 Results for Haar Classifier](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 34

[Table 6.2 Result for DNN](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 35

[Table 6.3 Results for HoG](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 35

[Table 6.4 Resuls for CNN](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 36

[Table 6.5 Results of the image processing techniques for Haar Classifier](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 37

[Table 6.6 Results of the image processing techniques for DNN](file:///F:\kişisel\okul\YL%20TEZ\Thesis_OzgurDeniz%20Gunseli_2012900915_en.docx#_Toc18429256) 38

# **CHAPTER ONE**

**INTRODUCTION**

**1.1 Background Information**

The fact that computers and computer-based systems make definitions of human behaviors and human emotions, generate data and automatically make the necessary inferences have increased the need for these systems. Although it may seem very easy for a human to identify, recognize and then analyze emotions, performing these actions for a computer requires complex calculations and algorithms. Facial expressions are a very simple phenomenon for a human being and have a very important place in human behavior in every aspect. During communication, words, as well as facial expressions, have a great place in transmitting information to the other person.

The development and acceleration of computers have made applications such as machine learning usable in these areas. Understanding the importance of analyzing human behaviors in computer environments in order to increase productivity in human life has accelerated the development of systems such as computer-based face recognition and emotion recognition. In the computer world, the use of face recognition algorithms or systems has become inevitable. For example, it is inevitable for artificial intelligence applications.

The perception of human facial expressions consists of three steps, in the most general sense, by a computer: the detection of the face, the feature extraction, and the classification of the facial expression. Each step constitutes a separate field of research and many methods have been developed to date, which has advantages over their particular circumstances. One of the most important criteria in applications using face identification and emotion expressions is the accuracy criteria in defining the target feature.

**1.2 Problem Definitions**

By nature, people may have different emotions from time to time, and emotion levels may change momentarily or over a long period of time. For example, anger can be shown as a momentary change. This anger may be related to work or personal life. In both cases, one's feelings greatly affect him or her. These effects may be reflected directly in the face of the person or may be reflected in the communication between colleagues. As a result, these emotions, which are not properly controlled and tracked, may decrease the productivity of the work environment, especially in the workplace. It is very important to follow the emotions by the managers or authorized persons in order to eliminate the occurrences and to predict future situations. Conditions should be measured well in order for the working individuals to be satisfied with their working environment, to continue their work efficiently and to provide emotional control. Measurements and data are becoming an increasingly important source for companies.

In terms of practical living conditions, it is difficult to follow employees emotionally. For example, in a company environment with very crowded teams, it can be difficult to analyze each individual's psychological-emotional analysis and extract data, and also increase resource requirements. On the other hand, it is seen that some managers or human resources officers do not give importance to such issues. Establishing a computer-based system or infrastructure for identifying employee emotions will both reduce resources spent on this area and increase employee productivity. Decreasing costs and increased productivity will result in a huge increase.

**1.3 Motivation/Related Work**

Emotions have a big impact on our lives. It is not only affecting the daily life, it has an important role to play in mood and motivation at work. An employee's emotion has a significant influence on his/her job performance, decision-making skills, leadership, team spirit, and job satisfaction.

The emotions and its management has been a big issue for human resource practitioners, administrators, and other specialists. The motivation of our work is developing a decision-making system that analyzes the employees' emotions and motivation from their facial expressions. In this way, human resource practitioners can track employees' productivity by the analysis of their mood and motivation.

The implementation of the detection of mood and motivation of employees will be provided by the facial emotion recognition system. Before starting the project, related works utilizing the facial emotion recognition were examined as well. The related projects have different purposes such as robotics, education, e-learning, health care, security of physical property or information. However, employees' mood and motivation detection is not implemented by the related works. Besides, when developing the project, support vector machine(SVM) and deep neural networks were preferred to train data set by the other studies. Likewise, convolutional neural networks will be used to train the data set in the project.

**1.4 Goal/Contribution**

In the previous sections, it was mentioned that related studies were conducted on different subjects such as education, health care, security, and entertainment. However, this project's goal is different than related studies on facial emotion recognition. A decision support system that helps human resources practitioners can track employees' mood and motivation at work. Thus, this work will be used to increase the productivity of the employees at work.

Technically, the project's goal consists of training a convolutional neural network with labeled images of static facial emotions. The network trained by the convolutional neural network will be used as part of software to detect emotions in real-time. The aimed to perceive the mental, and physical conditions of employees such as happiness, sadness, anger, fatigue, and insomnia. They will be used to provide support by analyzing situations that lead to loss of motivation and productivity.

**1.5 Project Scope**

The project is carried out by three members. Two of them are students working in the project and the other member is an expert to help the project progress. The project is planned to be completed between 23 September 2019 ad 27 December 2019. During the project progress process, hardware tools such as Nvidia Jetson Nano Developer and software tools such as Google's library Tensorflow will be utilized.

**1.6 Methodologies/Tools/Libraries**

Facial recognition and expression extraction will be performed by using machine learning applications and libraries that are widely used today. OpenCV, dlib, TensorFlow libraries will be used.

There are many projects on this subject. However, the difference between the projects is determined by the data sets and algorithms used. For example, one algorithm can identify distant faces well, while another algorithm identifies faces that are close but angled. Algorithms used in the first stage of finding faces are important in this angle.

Accuracy is provided by models designed with data sets. The use of the appropriate data set and the abundance of training data increase the accuracy. In addition, the language to be used during development is the Python programming language. The reason for using this language is that it has a lot of libraries and is a popular language and the possibility of finding solutions on the internet in case any error occurs.

***1.6.1 Convolutional Neural Networks***

A convolutional neural network (CNN) is one of the most popular algorithms for deep learning, a type of machine learning in which a model learns to perform direct classification tasks from images, video, text or sound. CNN's are especially useful for finding patterns in images to recognize objects, faces, and scenes. It learns directly from image data, uses patterns to classify images and eliminates the need for manual feature extraction.

A convolutional neural network has tens or hundreds of layers. Each layer serves to reveal a different property. The output of each layer is used as input for the other layer. Filters applied to images can start as very simple features, such as brightness and edges, and increase complexity in properties that uniquely identify the object.

Complex network architectures work with large amounts of data, so GPUs significantly speed up processing time to train a model because GPUs use optimized hardware architectures for matrix operations.

***1.6.2 TensorFlow***

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.‍

***1.6.3 OpenCV (Open Source Computer Vision)***

OpenCV is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions, etc.

***1.6.4 Dlib***

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real-world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high-performance computing environments. Dlib's open-source license allows to use it in any application, without any payment.

**CHAPTER TWO**

**LITERATURE REVIEW**

The process of facial expression recognition is defined into three stages:

* Preprocessing of input images
* Feature Extraction
* Facial expression recognition and classification

This literature survey report gives the main stages of the facial expression recognition project. It was understood that most of the projects intersect under these three main headings as a result of the literature research. The main reason for the difference between similar projects, articles or other published papers is related to how the methods they use or develop.

**2.1 Preprocessing of System Parameters**

To improve system performance, pre-processing is normally performed before issuing features for facial expression systems. The purpose of these steps, which includes scaling, density normalization, and size compensation, is to have images that contain only one face and express a distinctive emotion. Cropping and scaling operations were made taking the center point of the nose. Normalization was done to give the picture more clarity and make facial features are more distinctive (Emmanuel and Revina, 2018).

The color values ​​of an image vary according to the reflectance of that object and the perception of the device that makes up the image. Therefore, colors are not an accurate descriptive feature. It is necessary to pre-process the colors in order to increase accuracy. These color normalization processes make the image independent of the ambient light from which it was taken and the device that captures it (Finlayson and Tian, 1999).

A Facial Expression Recognition System Project by Matang N. mentions an image pre-processing includes removal of noise and normalization against pixel positions and brightness (Matang, 2016).

The article named, “Gauss–Laguerre wavelet textural feature fusion with geometrical information for facial expression identification” (Gavrilova et al., 2012), proposes a new feature extraction method. In order to achieve this, some preliminary procedures have been performed as follows. Input image is cropped and reduced to 128x96 pixels. Each person's face geometry is normalized to be easily detected by the face recognition model. To perform this normalization, eye positions should be determined.

Another article named, “Facial Expression Recognition Using Three Step Recognition Approach” states these preprocessing steps to remove noise and unnecessary data (Bhatnagar and Choubey, 2019). First, altering tone and immersion levels of image. Then converting 24-bit RGB image to 8-bit grayscale image and adjusting the size of the image to 512x512 pixels.

**2.2 Feature Extraction**

Feature extraction is a process that can be carried out after preprocessing. The main goal is figuring out which parts of an image are distinctive can uniquely describe the image. Feature extraction spots the move from graphic to implicit data depiction. The data depicted from feature extraction can be used for classification (Emmanuel and Revina, 2018).

Various pattern recognition techniques are used to classify different facial expressions based on facial features by most of the existing Facial Emotion Recognition Systems. The feature extraction methods can be categorized as geometric-feature based approaches, texture-based approaches, appearance feature-based approaches and hybrid features(combination of them) (Liliana, 2019). Geometric-feature based methods use the location of facial feature points such as lip corners, eye corners or the shape of facial components such as mouth, eye, brows. Another, widely used approach appearance-based method use texture feature of the facial image that is powerful to the variation of the illumination and the misalignment. Gabor wavelet features are widely used and the LBP features are intended for texture analysis. However, the LBP features have two main advantages that are tolerance to illumination changes and their computational simplicity. Hence, they have become very popular for Facial Emotion Recognition.

Considering the results of previous projects, Neural Network Classification applied a dataset consists of a set of 30 faces of people with varying skin color and lighting conditions with applied different feature extraction are examined. The accuracy values of edge detection, projection filtering & edge focusing, projection filtering, without edge detection additional condition and best combination are 80%, 73.3%, 70%, 73.3%, and 93.3% respectively (De Silva and Hui, 2003). In conclusion, combined approaches give the best result.

**2.3  Classification**

***2.3.1 Convolutional Neural Networks***

The development of the convolutional neural networks is related to the study of monkey's visual cortex. The relation between the study and convolutional neural networks is the way that cells are arranged on the visual cortex of monkeys. The visual cortex is represented as two types of cells which are simple, and complex cells. The simple cells detect and focus on edges and shapes. On the other hand, complex cells detect different types of objects. Hence, the objects or visual areas are visualized by different types of cells in the cortex (Spiers, 2016).

The first implementation of this study called Neocognitron and the model is developed by K. Fukushima in 1980 (Fukushima, 1980). The first layer of the model represents simple cells, the second layer represents complex cells. The local invariance property of the visual cortex is a great achievement of Neocognitron. Nevertheless, the learning process was a big issue for Neocognitron. Backpropagation method was not implemented for tuning weight values by checking an error measure of the network until 1985. After the introduction of backpropagation into ANN and CNN became more robust (Hinton et al., 1985). The main idea of backpropagation is measuring the gradient of the error according to weights and the value of the gradient is changing when the weight values are updated. Until the error of the network is minimized, the value of gradient in Gradient Descent is updated. Gradient Descent is an optimizer is used in backpropagation to autotune the parameters and looking for a local minimum of a function to minimize the error.

Considering the results of the related works from different CNN applied datasets, one of the examples is FER2013, which consists of seven basic emotions and the dataset based on posed images in a static environment. This study demonstrates the success of CNN. However, accuracy values are not distributed equally for each class (Ahmed et al., 2019). The overall accuracy is 91.12%. Nevertheless, the recognition rate for disgust and fear is 45% and %41 respectively. Another developed facial emotion recognition system based on images of different ages and in dynamic environments in the dataset named is RAF-DB. The classification consists of 7 basic emotions and 95.78% accuracy has achieved and the recognition rate in disgust and fear only stands at 62.16% and 51.25% respectively.

***2.3.1 Support Vector Machine***

SVM is a classification technique, and the objective of the SVM algorithm is to find a hyperplane in on N-dimensional space(N- the number of features) that classifies the data points distinctly. Support Vector Machines are maximal margin hyperplane classifiers that exhibit high classification for small training sets and good generalization performance on difficult to separate data (Gandhi, 2018). Kernel functions are employed to efficiently map input data that may not be separated linearly to a high dimensional feature space SVM's exhibits quite good classification accuracy even if the amount of data is not satisfactory. Support Vector Machine is a good choice to distinguish similar expressions such as anger and disgust. On the other hand, the appropriate kernel function selection provides more adjustments to optimize the SVM classifier for particular domains. In briefly, SVM's are memory efficient and effective in the high number of data.

Considering the results of different datasets that are applied Support Vector Machine, such as the Cohn-Kanade AU-coded facial expression database (Kanade, Cohn & Tian, 2000) has an overall accuracy of 87.9%. However, the recognition rate of fear stands at 76.2% (El Kaliouby and Michel, 2005).

**CHAPTER THREE**

**REQUIREMENTS/REQUIREMENT ENGINEERING**

**3.1 Functional Requirements**

For the facial emotion recognition system, the functional requirements are explained based on what the system should do. The used hardware, type of software being developed and the expectations of the user determine the functional requirements. Hence, the functional requirements are defined into two titles that are hardware requirements and software requirements.

***3.1.1 Hardware Requirements***

**• Camera**

Camera, device for recording an image of an object on an object on a light-sensitive surface. Cameras are required to obtain input images. So that, input images can be preprocessed to recognize facial expressions.

**• Raspberry Pi 4**

Raspberry Pi 4 is required to perform face detection and feature extraction. It reads data directly from the camera continuously. At the same time, it can store the preprocessed data when if there is not any connection to the server. After face detection and feature extraction tasks are performed, data is sent to the server to be classified later.

**• NVIDIA Jetson Nano Developer Kit**

Nvidia Jetson Nano Developer Kit is required to perform preprocessing and classification of the data. It reads data directly from the server and preprocessing it after then classifies the processed data.

***3.1.2 Software Requirements***

**• Dlib**

To process the input images taken from the camera, Dlib library is required. It has many approaches to preprocess the data such as HOG(Histogram of Oriented Gradient) based object detector to face detection and Facial Landmark Detector to feature extraction. Besides, to classify processed data it has classification approaches such as Neural Networks as well.

**• OpenCV**

OpenCV may be used for face detection as well. OpenCV and Dlib are the most commonly used face detection open source libraries. Haar Cascade Classifiers can be used for face detection with OpenCV.

**• Tensorflow**

TensorFlow is an open-source library for numerical computation that makes machine learning faster and easier. In our application, the classification of the preprocessed images is required and Tensorflow has the classification model that is required in the application which is Convolutional Neural Network.

**• Mobile or Web Front-End Application**

Mobile or Web Front-End Application required to track each employee's psychological-emotional analysis by human resources officers or managers to increase the productivity of employees.

**3.2 Non-Functional Requirements**

**• Cameras used to obtain input images**

A system that collects data from camera systems in the first stage. So that, system can detect people and their mental states to increase the efficiency of the employees.

**• Raspberry Pi 4 used to perform face detection and feature extraction**

After the input images obtained by cameras, face detection and feature extraction of the images should be done. The operations mentioned are handled by Raspberry Pi 4. It is connected to one or more visual data source to preprocessing and to collection of the data. Considering the number of visual data sources, Raspberry Pi 4 may be increased for the size of the area. Hence, to reduce the cost mini-computers preferred.

**• Nvidia Jetson Nano Developer used to perform preprocessing and classification**

After face detection and feature extraction is handled by Raspberry Pi 4, Nvidia Jetson Nano Developer is used to preprocess and classify the images. The face detection and feature extraction operations applied images are recevied from the server which was sent by Raspberry Pi 4 to the server. Besides, it provides to run multiple neural networks in parallel for preprocessing the input image.

**• Dlib used to process the input image**

For the detection of faces, Dlib HOG Based face detection is used and for the feature extraction, Dlib's Facial Landmark Detector is selected.

**• Tensorflow used for classification of the preprocessed image**

Tensorflow is a free and open-source software library for dataflow and differencing programming across a range of tasks. It enables model trainings on developed artificial intelligence models and provides great performance of model training with GPU. In the application emotion recognition process is carried out with the Convolutional Neural Network approach and to implement CNN, Tensorflow is selected.

**• Mobile or Web front-end application used to build interface to users**

An application is designed to build an interface to users. Each employee's mood and motivation will be indicated in the application and propose methods to increase the productivity of the employee. The output will not contain any personal data.

**CHAPTER FOUR**

**DESIGN**

**4.1 General Architectural View**

The operating environment of the system is divided into three different environments. The part where the visual data received from the cameras are preprocessed and their properties are determined constitutes the end point architecture of the system. A camera or multiple cameras send images to an end point(s) system. There is an many-to-many relation between cameras and end-point controllers. These low-power controllers preprocess the received data and send it to the servers in the host system. A centralized system or part of the system's major side has high processing power and runs classification algorithms and stores them in databases. Application software, which is the front of the system, that works on different types of devices or works on a web-based basis, can use the data obtained from the databases in their own analysis algorithms. Figure 4.1 visually represents the architecture of the system.

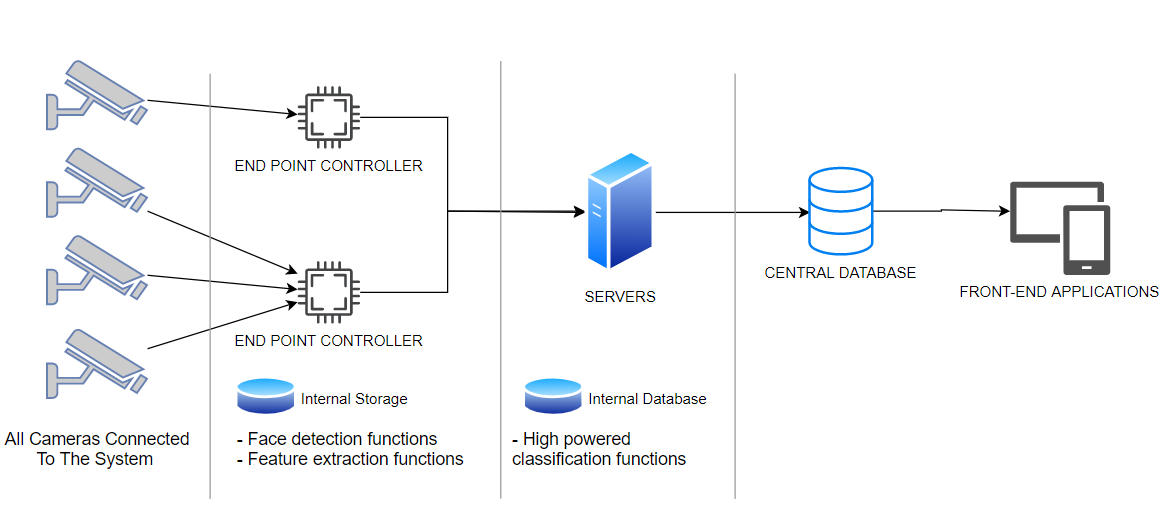


Figure 4.1 Overall system architecture

**4.2 Entity Relationship Diagram**

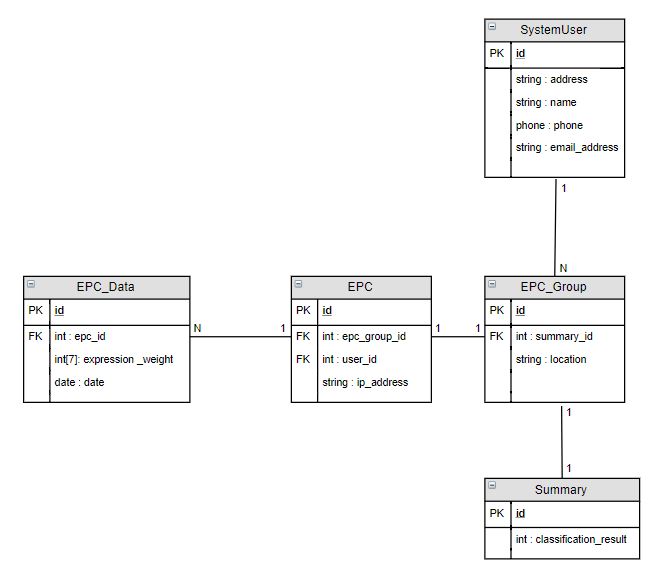


Figure 4.2 Entity relationship diagram

Database contains user information stored in the system, EPC (End-point Controller) terminal information of users, EPC groups created by the user, and classification results of data collected from ECP groups are stored in separated table which name is Summary. The results of the classification process are recorded in the EPC\_Data table. This table contains the classification result of each data read from EPC. For example a region or a department that uses more than one controller that means a group of controllers work as part of the system. Multiple controllers operating in a zone or department are grouped by the system. EPCs that are close to each other means that they are involved in the same system analysis, and therefore each EPC in the database is kept as a member of a group.

**4.3 Class Diagram**

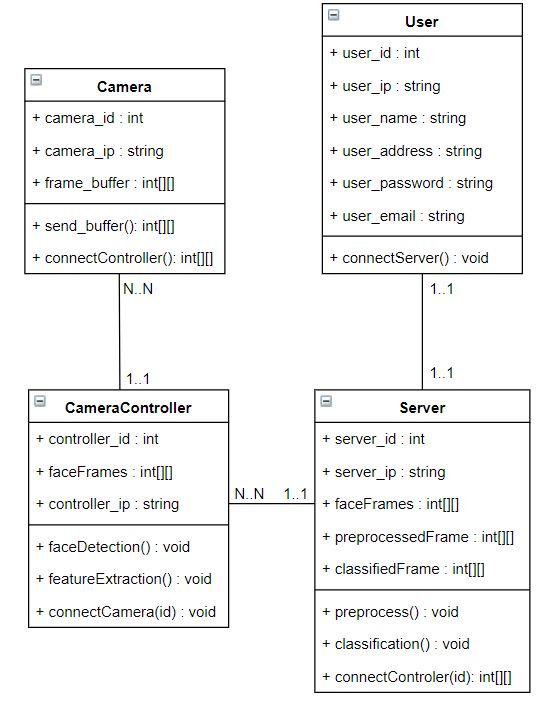
****

Figure 4.3 Class diagram

It’s hard to generalize the system objects in a single class diagram since the system work as a huge distributed environment. Some parts of the system work in the low powered device and its software do not support object-oriented methodology and the other part of the system software can use object-oriented programming advantages.

If focus on the main architecture of the system that includes three main part needed to develop. Camera, controllers, servers and users are the main objects. All objects have an “id” variable to identify them and the communication functions to send and get data between each other. Camera sends the frame buffer that includes data about pictures to controller which is detected and extracts features from the picture. Server has all classification and preprocessing functions to finalize the system processes. User objects can connect the server to get data for front-end applications that wanted to run analysis on data.

* 1. **Use Case Diagram**

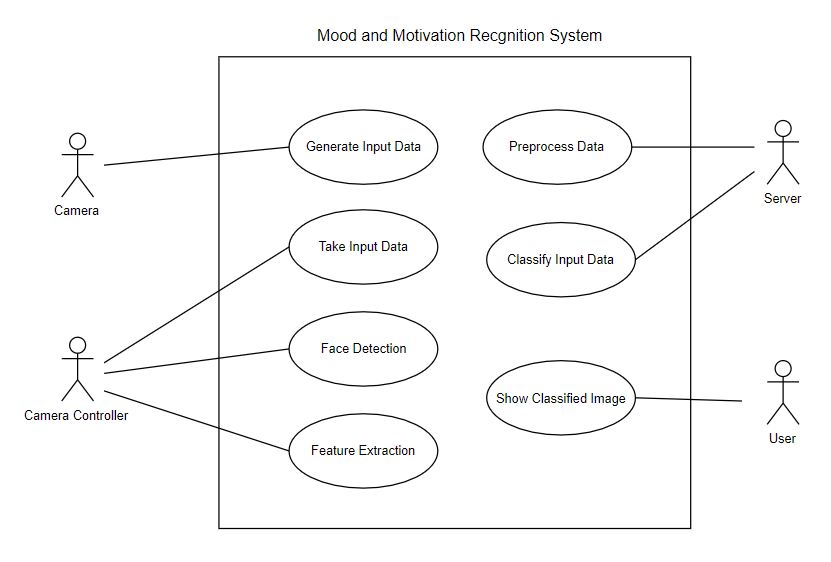
****

Figure 4.4 Use case diagram

As mentioned in the class diagram, the object now are the actor. All use cases that an actor can manage are shown in Figure 4.4. Camera generates input data, controller detects faces and extract features. Server runs preprocessing and classification algorithms and users can manage all manipulations on data and shows all images on system.

**4.5 Sequence Diagram**

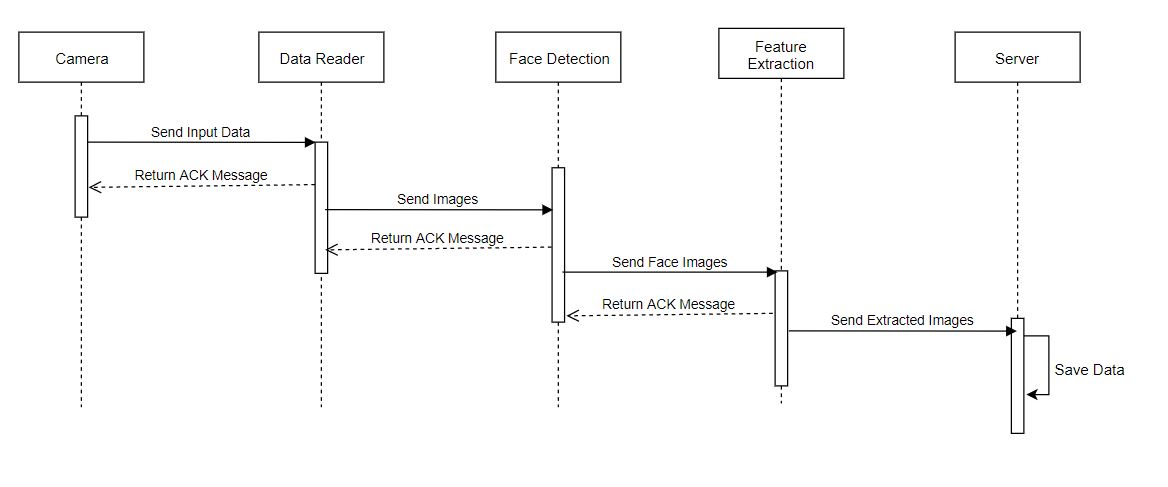


Figure 4.5 Controller side sequence diagram

The end-point controller part of the system works as face detector and feature extractor. All sequences between camera and local database are mentioned in Figure 4.5. The camera sends data to the reader and inter processes run and finally server saves extracted images.

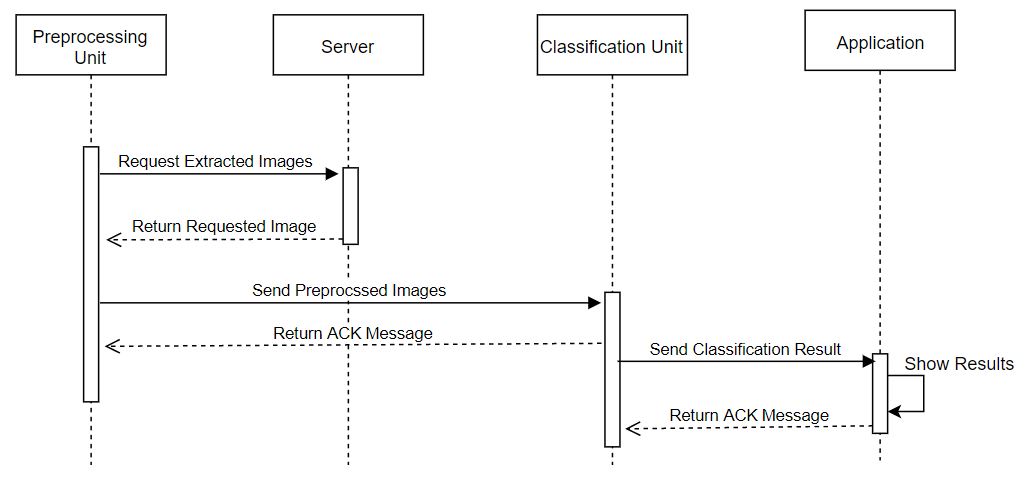


Figure 4.6 Server side sequence diagram

If focusing on server side of the system, the all sequences between server and application are shown in Figure 4.6. Preprocessing unit requests images and send them to the classification unit. Applications receives classification result and shows them to user.

**4.6 Deployment Diagram**

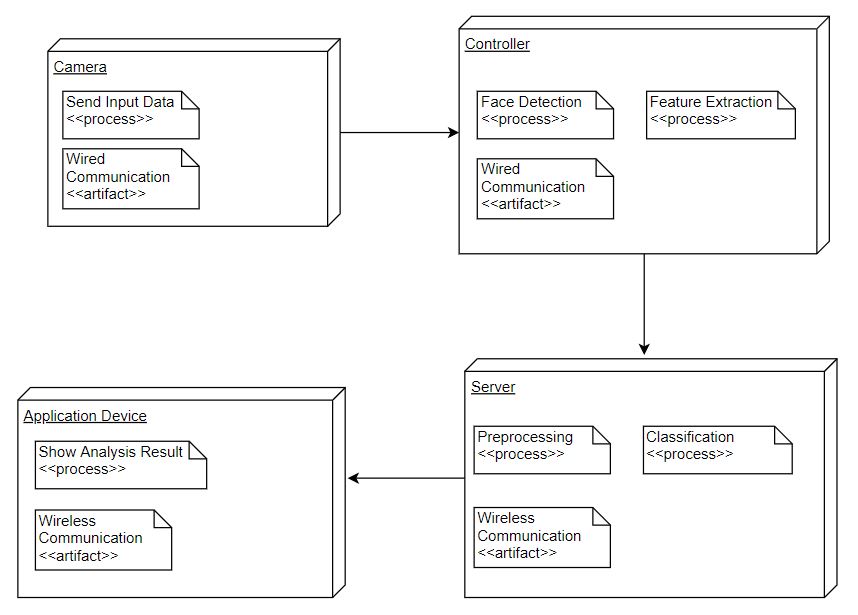
****

Figure 4.7 Deployment diagram

The final physical appearance of the system is as shown in Figure 4.7. Distributed cameras send image-related data to the controllers and the controllers transmit the images to the server after completing the operations shown in the figure. Physically, the server is located in a centralized system and runs classification algorithms that require powerful processing power. As a result, separate application devices from the other system do the necessary operations with the information they receive from the server.

**CHAPTER FIVE**

**IMPLEMENTATION**

**5.1 Face Detection Implementation**

There are two main stages of the facial emotion recognition system. At the first stage of the facial emotion recognition, face detection is implemented. After then, emotion recognition is carried out. Nowadays, so many techniques have been developed for face detection. Some of the popular ones tested on different datasets to choose the most suitable technique for the project. Those popular algorithms which are tested Viola & Jones Algorithms(Haar Classifier), DNN Face Detector, HoG(Histogram Oriented Gradient) and CNN Face Detector.

CNN face detector is the algorithm which is implemented for face detection. Indeed, some of the other algorithms are quite successful as well. The main reason for choosing CNN face detector is accuracy and f1-score values are more successful and consistent compared to other algorithms. Haar and HoG algorithms are not very successful. However, DNN face was very successful and fps(frame per second) rate was better than CNN. The drawback of the DNN was, it is not consistent which means that it was not reliable on different datasets. That is why CNN is implemented on the project. Since it is very reliable at the detection of different angles of the faces. In addition, it works very impressive on the GPU compared to the CPU. In order to work on GPU, Cuda is used which improved the fps rate.

The implementation of CNN Face Detector is carried out by dlib library. The model created for face detection is applied by using the function from dlib library. The function detects the faces in the image by using the model created before from different datasets. The important point of the function is given input to the function must be converted into grayscale. Since it does not work on RGB images. After the faces are detected they are cropped and stored to apply emotion recognition. On the other hand, before applying face detection, image processing is applied to get better results. The most preferred image processing technique is histogram equalization. The histogram equalization distributes the intensity values of the image equally. It is very useful due to the environment of the camera can be too dark or too bright. In that case, our model can not detect faces as we desired. The problem is handled by using histogram equalization.

After removing the faces detected in the frames in the face identification stage, they proceed to the classification stage. At this stage, the SVN and CNN machine learning classification algorithms that we train classifies so that it can be determined which is the closest to 7 types of emotions. At this stage, the fer2013 data set was used because each face data in this data set was signed ready for use in model training. The emotion number, grayscale 48x48 image pixel values are kept in the data set.

In the training of classification models, it is aimed to determine the best performance according to different performance measures by using HOG and face landmark features. The trained “Dlib shape predictor model” was used during feature extraction. This model determines the coordinates of 68 landmarks in the face picture. In the next stage, these coordinates were used in model training. HOG (Histogram Oriented Gradient) was tried as another feature extraction method.



Figure 5.1 Landmark demonstrations on sample image

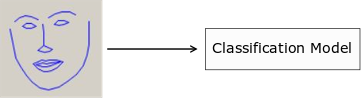


Figure 5.2 Detected landmarks from sample image

As explained in the design phase, CNN and SVN models are used in the project since they are successful in image-based classifications. It is very clear in every respect that CNN is more successful, but it is considered a disadvantage that it requires more processor power. SVM is faster and uses less processor power, but the hit value is less.

In SVM model training, "svm" library in Python "sklearn" was used. SVM model parameters are shown below;

* Kernel = rfb (rfb, linear, poly, sigmoid)
* Decision function = ovr (OneVsOne, OneVsRest)
* Gamma = auto (or float number)
* Features = landmarks (hog, landmark + hog)

The CNN model was used in the next phase of the application. Faces, such as the method used in SVN, are first detected using the Dlib library. The perceived face pictures are brought to a common size and grayscale form. Then, landmarks and/or HOG features extracted to the CNN neural network are sent. Using the features in the 30,000 face data in the FER2013 data set, the CNN model will be trained for use on the classification server.

**5.2 CNN Model Implementation for Emotion Recognition**

***5.2.1 Tensorflow and Keras API***

Keras is a top level API of Tensorflow. It is used to build and train models in the machine learning area. It provides the user with significant advantages and ease of use. It is optimized for general use cases and provides easy feedback in error situations. It offers a modular structure, allowing to configure the model layers separately. Custom-made layers can be designed for scientific test results and experiments, and test metrics can be easily recorded with loss function.

***5.2.2 Model Architecture and Layers***

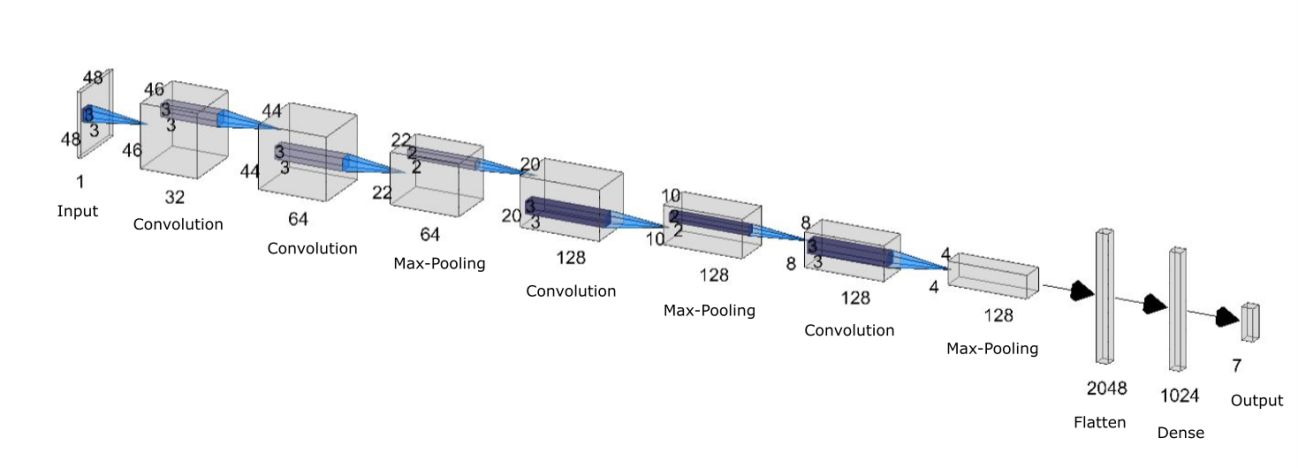


Figure 5.3 General architecture of model layers

Individual layers are designed and created with Keras and usually a model consists of layer graphs. The most common model type is the so-called "sequential" model type, where layers are stacked.

model = tf.keras.Sequential()

Sequential model was used in this project. Figure 5.2 includes the design form of the CNN model used in the emotion classification phase of this project, and general input / output types, layer volumes and filter sizes are clearly visible. The model basically consists of convolution + pooling layers. The parameters and filter types of each layer can be changed for the best accuracy value.

*5.2.2.1 Layers and Parameters*

First of all, the emotion classification model consists of input, convolution, max-pooling layers. Transfers each layer output sequentially to another layer. The first layer of the model is always an input layer. Code implementations are done in Python. Descriptions of all layers and parameters are listed below;

**Input Layer:** It is the layer where the initial dimension is defined and this layer is the first layer of modeling. The input image is filtered starting from this layer and sent to the other layers as input. All the images used enter the model with a pixel array of 48x48x1 size. It is also a convolution layer.

tf.keras.Conv2D(32,kernel\_size=(3,3),activation=’relu’,input\_shape=(48,48,1))

Since the number of filters is 32 and the size is 3x3, the output size of this layer is reduced to 44x32 and transferred to the next convolution layer.

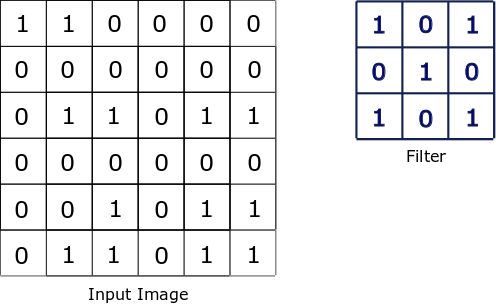


Figure 5.4 Representaion of input image and filter size

**Convolutional Layers:** It is the main component of a CNN model and creates a layer feature map. The Emotion classification model consists of three convolution layers with the same parameters. The number of filters used is 32, 64, 128, respectively. Each layer focuses on a different feature, such as vertical lines, horizontal lines and corners. In this way, as you move into the network, the shapes in the face pictures are perceived better. The size of the picture is halved after the pooling phase, but since the number of filters doubles, there is no loss of features.

tf.keras.Conv2D(64,kernel\_size=(3, 3), activation=’relu’)

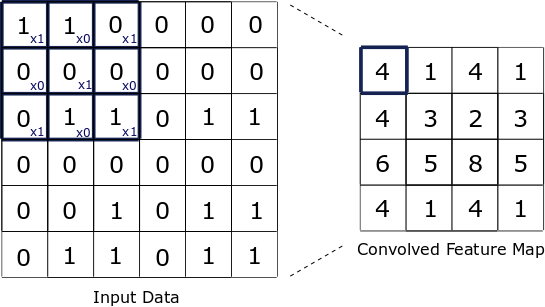


Figure 5.5 Illusturation of Convolution Processes

**Pooling Layers:** Generally, it is the layer that is placed to decrease the computing power, the number of processed data can be reduced by using the pooling layer. The model uses three pooling layers to come after each convolution layer. Pooling filter is set to 2x2. This reduces the number of processed data to be halved in a single dimension after each pooling layer.

tf.keras.MaxPooling2D(pool\_size=(2, 2))

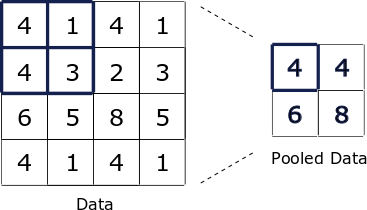


Figure 5.6 Illustration of Pooling Process

**Flattening:** In order to send the feature map created after the Convolution + pooling phase to the fully connected layer in the first layer of the neural network, we need to place the flattening layer. The volume of the data up to this stage decreases to 4x4x128, and when this point is reduced to a single dimension, a 2048 array is obtained (4x4x128 = 2048).

tf.keras.Flatten()

**Dense Layer:** There are two layers in the neural network of model. One of them is hidden layer, the other is the last layer, the output layer.

tf.keras.Dense(1024, activation=’relu’)

**Dropout:** As they are very prone to CNN model overfitting, a dropout value has been set after each convolution + pooling layer to prevent overfitting. This value is a probability value between 0-1 and allows neurons to be randomly deactivated. There are three dropout layers in the model; their values are 0.25, 0.25, and 0.5 respectively. The value has been increased according to the number of neurons in the layers.

tf.keras.Dropout(0.5)

**Output Layer:** There are seven neurons to determine each emotion class to complete the classification phase. The Softmax activation function converts the output values from these seven neurons into probability-based values. The largest value of these values between 0-1 activates the corresponding neuron and returns an array of seven as output. The index, which has a value of 1, is the emotion index that is classified by the model.

tf.keras.Dense(7, activation=’softmax’)

*5.2.2.2 Compiling the Model*

After building the model, learning processes are started by calling the "compile" method. The important parameters used in this method are as follows.

* **optimizer:** This object contains a description of the procedures to be used in the training phase. In this project, "adam" optimizer was used.
* **loss:** Determines the type of function that tries to minimize the margin of error during optimization. Since this model has more than one categorized class, the “categorical\_crossentropy” function has been used.
* **metrics:** It provides the determination of metric values, which enables the monitoring of the training phase.

The "model.fit ()" function is used to determine which dataset the training phase will be on. The parameters used and their definitions are listed below.

* **epochs:** Trainig stage is done with epochs. Epoch is an iteration on training data. As the number of epoch increases, the model learns the training dataset better, but this can lead to overfitting.
* **batch\_size:** Since the number of instances in dataset is very large, the model divides the data into smaller batches and carries out the training phase over the data in these batches. This parameter allows us to determine the size of these batches.
* **validation\_data:** Validation data is used to measure and track the performance of prototype models created until reaching the best performance model. This dataset consisting of input and label tuples are sent to the "fit" function as an argument.

*5.2.2.3 Saving the Model Weights*

The created model configuration and weight values of neural connections can be saved for later use. The saved pattern configuration can then be used to initialize a model without coding. Keras supports recording the model's weight values in files with the extension ".h5". Model configurations (layers and parameters) are saved in “.json” files.

model.load\_weights('my\_model.h5')

json\_string = model.to\_json()

***5.2.3 Model Evaluation and Results***

To evaluation of the model includes splitting the given dataset into training and test, optimizing the number of epochs, batch number, activation functions, and the size of the layers in the network. In addition to those parameters, accuracy and f1-measure is another important evaluation metrics. In the next chapter how those parameters are adjusted and the results of them are explained.

**5.3 MongoDB Deployment**

MongoDB is an open-source database that uses a document-oriented data model and a non-structured query language. It is one of the most powerful NoSQL systems and databases. NoSQL means that it does not use the usual rows and columns as the relational database management. It is built on documents and collections. The database consists of a set of key-value pairs that help documents to have different fields and structures. The data model that MongoDB uses is very elastic which helps to combine and store data of multivariate types without having to compromise on the powerful indexing options, data access, and validation rules.

***5.3.1 MongoDB Architecture***

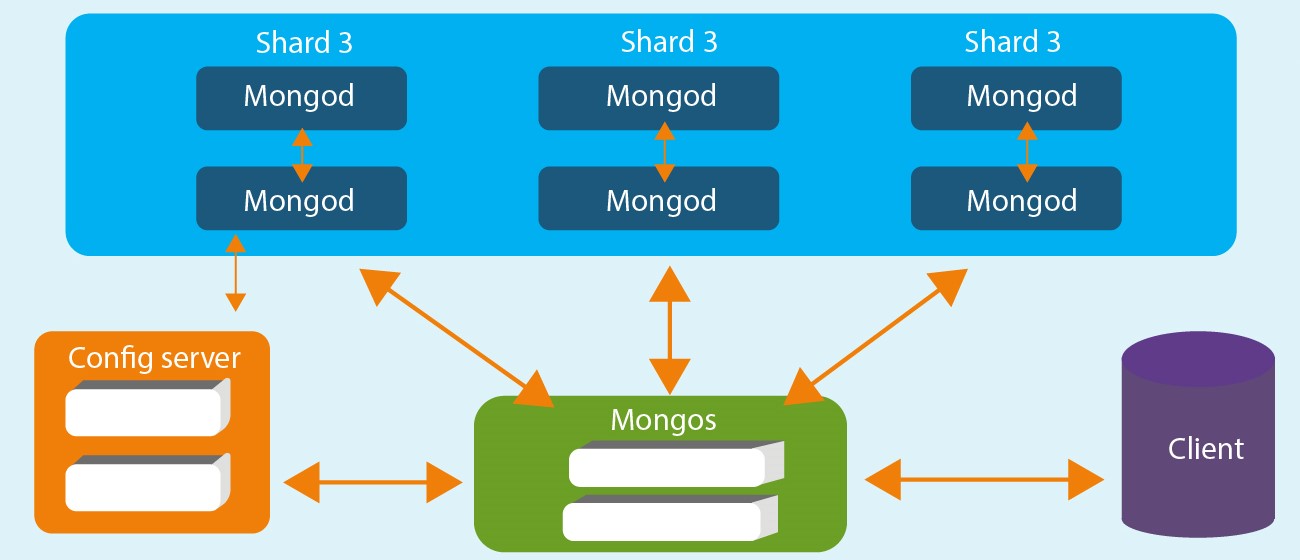
****

Figure 5.7 Illustration of MongoDB Architecture

The architecture of MongoDB consists of a database, collection, and documents. Also, it is illustrated in the Figure 5.6. Database can be called the physical container for data. Each of them has its own set of files on the file system with multiple databases existing on a single MongoDB server. A collection is a group of database documents. The equivalent in RDBMS is a table for collection and all collection exists in a single database. The documents within a collection are meant for the same purpose or for serving the same end goal. Besides, documents are a set of key-value pairs that can be designated as a document. They are associated with dynamic schemas and a document in a single collection does not have to possess the same structure.

In the project, the architecture can be summarized as features below.

* Features: The stored data
* CameraID: Id of the camera
* ClusterID: The group of the camera
* Date: The date of the data received

***5.3.2 Node.js REST API Implementation***

First of all, environments such as Node.js and MongoDB are set up and their version is checked by the commands npm -v and mongo --version. After then a folder is created and a JSON file is created. The purpose of the JSON file is to give the necessary information to npm which allows it to identify the project. npm init prompt is used to enter some information about the app. The HTTP protocol is used for data transformation and post, get functions are used. In addition to them, ImageProcessing\_API and EmotionDetection\_API are created to use for camera and server sides. Their responsibility is classification of the given faces.

***5.3.3 Sending Face Features To MongoDB***

Sending face features to MongoDb is initialized with establishing a connection to the server. After then the database is created and the database is updated with the cropped face data. The cropped faces are sent to the server.

**5.4 PostgreSQL Deployment**

***5.4.1 Relational Database Architecture***

The relational database architecture can be summarized into 3 tables which are named camera, cluster, and emotionData tables. The purpose of the camera table is to store the ids of the cameras and its cluster, the aim of the cluster table id is numerating the clusters and emotionData's purpose is storing the emotion labels scaled 1 to 7.

***5.4.2 Node.js REST API Implementation***

There are two files that are named index.js and query.js for PostgreSQL. Query.js file manipulates the data such as selection, insertion. For instance, there are functions like getEmotion and createEmotion. Index.js file is working with the server.

***5.4.3 Sending Labelled Classification Data***

The classification server puts the data into the classification model using the data it receives from mongodb and labors this data by creating the indexix indeindex. Then, the labeled data is written to the relevant table in the relative database using the node.js rest api to the postgre database. Thus, all of the data in mongodb are processed in real time and ready for analysis.

**5.5 Simulating the Data Flow and Database Connections**

In general, an interface was prepared using C # form application to start the data flow and run the parts of the system separately. The aim here is both to be the basis for the admin side of the system and to speed up the developer's work by automating the execution of the necessary programs during the development stages.

The figure below also shows the interface of the simulation. After starting the servers via this interface, the desired camera - currently there are only 5 cameras for testing purposes - is started.

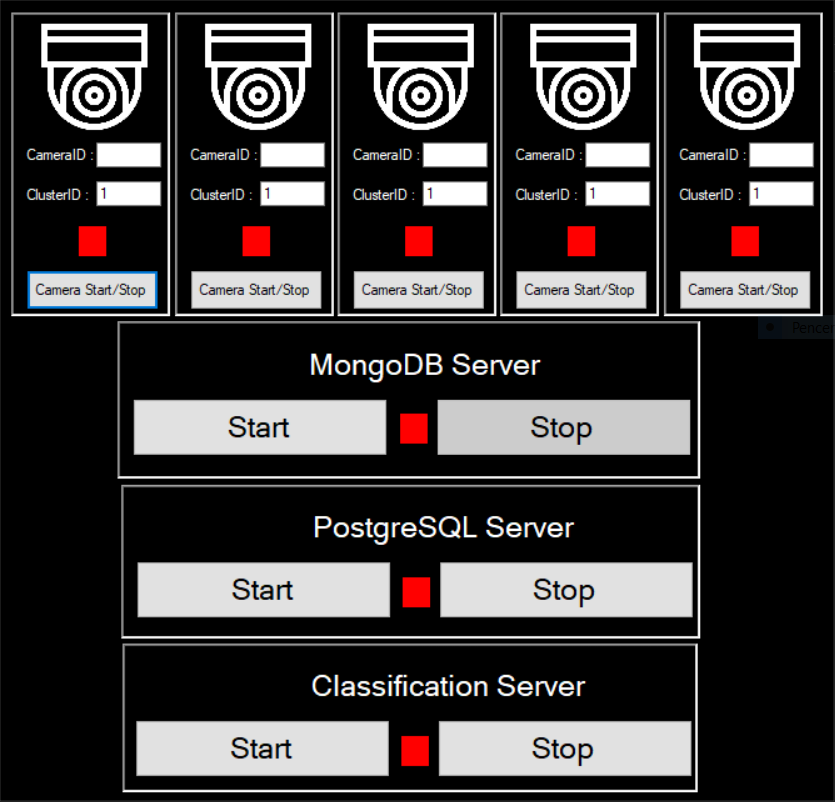


Figure 5.8 Dataflow Simulation Interface

**CHAPTER SIX**

**TEST AND EXPERIMENTS**

**6.1 Experiments of Face Detection Algorithms**

There are many available algorithms to detect faces. That is why the algorithms that might be applied are decided initially then suitable datasets and image processing techniques are determined. Those algorithms are Haar Classifier(Viola&Jones), DNN Face Detector, HoG(Histogram Oriented Gradient), and CNN(Convolutional Neural Network). Each algorithm is applied to different datasets to see performance clearly. Those datasets are UTKFace, Jaffe and FER2013.

Table 6.1 Results for Haar Classifier

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Total # of face images** | **Total # of non-face images** | **TP** | **FN** | **TN** | **FP** | **Accuracy** | **F-mesaure** | **Rate (img/s)** |
| UTKFace | 23707 | 3000 | 3127 | 20581 | 2886 | 114 | 22.50% | 23.20% | 143.62 |
| Jaffe | 213 | 3000 | 213 | 0 | 2886 | 114 | 96.45% | 78.88% | 52.85 |
| FER2013 | 35887 | 3000 | 785 | 35102 | 2886 | 114 | 9.44% | 4.26% | 292.38 |

First of all, Haar Classifier(Viola & Jones) is tested on three different datasets with image processing techniques and without image processing techniques. According to Table 1, the result of the algorithm is pretty low. It is not suitable for face detection. However, when Haar Classifier is applied to the Jaffe dataset, the result is much better compared to the other datasets. Since due to the low number of instances in the dataset and the high number of the non-facial datasets, algorithm did not detect faces on the non-facial dataset which increased the True-Negative value. Hence, it results seems better. Indeed, True Positive value is very low and it is not reliable for us.

Table 6.2 Result for DNN

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Total # of face images** | **Total # of non-face images** | **TP** | **FN** | **TN** | **FP** | **Accuracy** | **F-mesaure** | **Rate (img/s)** |
| UTKFace | 23707 | 3000 | 23680 | 28 | 2029 | 971 | 96.26% | 97.93% | 22.53 |
| Jaffe | 213 | 3000 | 213 | 0 | 2029 | 971 | 69.77% | 30.49% | 22.63 |
| FER2013 | 35887 | 3000 | 35599 | 288 | 2029 | 971 | 96.76% | 98.26% | 22.24 |

DNN Face Detector is tested on three different datasets the same as Haar Classifier. Its result is much better than Haar Classifier. However, the main drawback of the DNN is slower than the Haar Classifier. DNN Face Detector is very successful at detecting faces as we can see from Table 2, True Positive values are quite high. However, FP values are very high too which decreases the performance of the algorithm.

Table 6.3 Results for HoG

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Total # of face images** | **Total # of non-face images** | **TP** | **FN** | **TN** | **FP** | **Accuracy** | **F-mesaure** | **Rate (img/s)** |
| UTKFace | 23707 | 3000 | 22868 | 839 | 2926 | 74 | 95.54% | 97.43% | 2.24 |
| Jaffe | 213 | 3000 | 213 | 0 | 2926 | 74 | 97.69% | 85.20% | 0.3 |
| FER2013 | 35887 | 3000 | 24913 | 10974 | 2926 | 74 | 71.58% | 81.85% | 1.73 |

HoG(Histogram Oriented Gradient) is tested on three datasets as previous tests. Considering Table 3, the result of the algorithm is not as bad as Haar Classifier. However, it is as successful as DNN Face Detector.  True Positive values in the HoG are lower than the DNN as seen in Table 3. Especially on Fer2013, the difference is obvious. That is why Fer2013 consists of 48x48 px images. It shows that HoG is not good at detecting small-scale images. In our case, we need to use algorithms that are successful in detecting small-scale images. Thus, HoG is not a good choice.

Table 6.4 Resuls for CNN

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Total # of face images** | **Total # of non-face images** | **TP** | **FN** | **TN** | **FP** | **Accuracy** | **F-mesaure** | **Rate (img/s)** |
| UTKFace | 23707 | 3000 | 23583 | 125 | 2922 | 78 | 99.23% | 99.57% | 169.57 |
| Jaffe | 213 | 3000 | 213 | 0 | 2922 | 78 | 97.57% | 84.52% | 365.89 |
| FER2013 | 35887 | 3000 | 31132 | 4755 | 2922 | 78 | 87.57% | 92.79% | 292.64 |

The last algorithm tested on different datasets is CNN. Normally, for the test other algorithms are tested on CPU. However, CNN is very slow on CPU. It was going to take nearly two or three days to test each dataset. Hence, CNN and DNN algorithms are carried out on GPU and to work on the GPU Cuda is implemented. From Table 4, the best results are taken from the CNN algorithm. It is even better than DNN Face Detector. Unfortunately, the main drawback is slow processing. However, it is handled by working on GPU.

Table 6.5 Results of the image processing techniques for Haar Classifier

|  |  |  |  |
| --- | --- | --- | --- |
| **Image Processing Method** | **Dataset** | **Accuracy** | **F1-mesaure** |
| - | Jaffe | 96.45% | 78.88% |
| Clahe(Histogram Equalization) | Jaffe | 95.89% | 75.40% |
| Histogram Equalization | Jaffe | 96.14% | 77.45% |
| Log Transformation | Jaffe | 97.26% | 82.81% |
| - | FER2013 | 9.44% | 4.26% |
| Clahe(Histogram Equalization) | FER2013 | 7.54% | 23.37% |
| Histogram Equalization | FER2013 | 9.52% | 36.50% |
| Log Transformation | FER2013 | 8.54% | 19.56% |
| - | UTKFace | 22.50% | 23.20% |
| Clahe(Histogram Equalization) | UTKFace | 14.52% | 8.12% |
| Histogram Equalization | UTKFace | 22.17% | 22.65% |
| Log Transformation | UTKFace | 18.47% | 15.64% |

According to Table 5, image processing techniques applied to different datasets. The applied image processing techniques are Clahe, Histogram Equalization, and Log Transformation. Clahe is a different version of Histogram Equalization, it does not lose any information which was supposed to get better results compared to Histogram Equalization. Nevertheless, the results indicate that it does not affect as much as Histogram Equalization. In some cases, image processing did not affect the result or is affected in a bad way. The best result is taken from the Fer2013 dataset. Histogram Equalization is affected the accuracy and F1-measure better than other image processing techniques.

Table 6.6 Results of the image processing techniques for DNN

|  |  |  |  |
| --- | --- | --- | --- |
| **Image Processing Method** | **Dataset** | **Accuracy** | **F1-mesaure** |
| - | Jaffe | 69.77% | 30.49% |
| Clahe(Histogram Equalization) | Jaffe | 70.18% | 30.50% |
| Histogram Equalization | Jaffe | 72.17% | 32.275 |
| Log Transformation | Jaffe | 79.08% | 38.78% |
| - | FER2013 | 96.76% | 98.26% |
| Clahe(Histogram Equalization) | FER2013 | 96.06% | 97.87% |
| Histogram Equalization | FER2013 | 96.77% | 98.26% |
| Log Transformation | FER2013 | 88.07% | 93.22% |
| - | UTKFace | 96.26% | 97.93% |
| Clahe(Histogram Equalization) | UTKFace | 96.39% | 98.00% |
| Histogram Equalization | UTKFace | 96.59% | 98.11% |
| Log Transformation | UTKFace | 93.64% | 96.39% |

Image processing techniques are examined for the DNN as well. They are shown in Table 6. On the Jaffe dataset, Log Transformation is more successful compared to other image processing techniques. Yet, it is not successful on Fer2013 and UTKFace datasets. Histogram Equalization is slightly better than Clahe on the other datasets. On the other hand, when no image processing techniques are applied, the accuracy and F1-measure values are not changed much and it does not have any cost for us.

**6.2 Experiments of Facial Emotion Detection Algorithms**

There are two popular emotion classifier algorithms which are SVM(Support Vector Machine) and CNN(Convolutional Neural Network). Those are two classification systems are tested to which one is more suitable for the project.

First of all, the SVM classifier is tested for emotion detection. Before testing the model is trained. In order to train an SVM model, the Fer2013 dataset is used then face landmarks extracted and HoG Features. The extracted face landmarks feeder into out multiclass SVM classifier. The best result obtained when face landmarks and HoG features used which is nearly 50% accuracy for seven emotions and for five emotions better accuracy rates obtained. It was close to 59% accuracy. When HoG features have not implemented the accuracy is nearly 39%. That is why face landmarks and HoG are both used while testing. On the other hand, SVM is a much more simple and faster approach compared to CNN. However, the results are not satisfying.

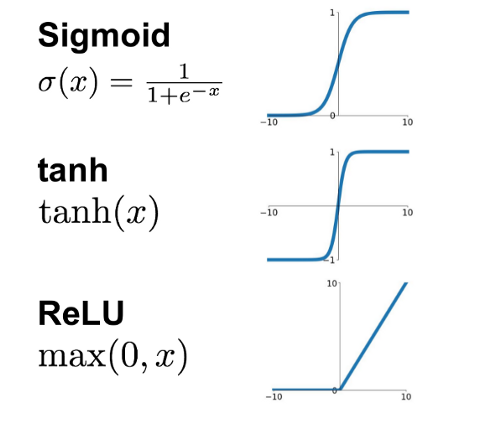


Figure 6.1 Activation Functions

The CNN classifier is tested after the SVM classifier is tested. For testing, the CNN classifier Fer2013 and CK+ datasets are separately used to create models and they are tested. In order to create a model, there are a few important points. In our test for CNN model implementations, Tensorflow Keras API is used. Since it is very modular and easy to extend. In order to create layers of the model activation functions and kernel size are declared. The chosen activation function is Relu due to the non-saturation of its gradient and it accelerates the convergence of stochastic gradient descent compared to the sigmoid and tanh function. When Relu, sigmoid, and the activation functions best result is obtained from the Relu activation function. It is pretty simple compared to the other activation functions and it does not involve expensive operations. However, it is still effective. The graphs of the activation functions are shown in Figure 6.1.

At the test phase of CNN model training, batch size, number of epochs, and dropout are the important parts of it. The batch size is chosen smaller at the beginning then it is increased while testing. After the batch size is increased it did not affect the accuracy desired. It is chosen 64 because the number of instances in the dataset is too high and it needs too much computing power and the accuracy did not change.

Another important parameter is number of epochs. It was chosen carefully. The higher number of epoch chosen it will be more effective for the model is not the right idea. While declaring the epoch size validation and training error is considered. If epoch number is not given right which may cause overfitting. It is understood by checking the validation loose number while training. If it is increasing that means the model started to overfitting which means it is memorizing. Therefore the epoch number is given carefully and increasing the epoch number increased the accuracy of the model. Another important parameter is dropout which can cause or prevent overfitting. Hence, it should be given very carefully. The dropout rates are given nearly 50% for hidden units and nearly 20% for visible units. After changing to 80% for hidden units and 40% for visible it increased the accuracy of the model. The maximum accuracy reached for the Fer2013 dataset is 65%. However, due to the low number of instances in the CK+ dataset, the result was not good.

# **CHAPTER SEVEN**

**CONCLUSION AND FUTURE WORKS**

Our project's purpose is the detection of employee’s motivation and mood during work. The main objective is by using the face emotions from camera systems and giving recommendations. In order to detect face emotions a model trained by using the convolutional neural network which has 65% accuracy for seven emotions from the Fer2013 dataset. Considering the results from the competition it is a quite successful result. At the next stage, the aim is to use the trained model creating a user-friendly application.

The system consists of three parts. The cameras are the data source of the system and detect the faces in the frames they receive using the face detection algorithm. After the detected faces are cropped, they are transferred to the second part through the database server. This section keeps the face data waiting to pass the classification stage on the basis of the pixel. The central classification algorithm, which runs continuously and waits for data, detects emotions and records the data on the relational database.

**REFERENCES**

Choubey, M. T. S. B., & Bhatnagar, S. (2019). Facial Expression Recognition Using Three Step Recognition Approach.

David E Rumelhart, Geoﬀrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985

De Silva, L. C., & Hui, S. C. (2003, December). Real-time facial feature extraction and emotion recognition. In *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint* (Vol. 3, pp. 1310-1314). IEEE.

Finlayson, G. D., & Tian, G. Y. (1999). Color normalization for color object recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, *13*(08), 1271-1285.

Gandhi, R. (2018). Support Vector Machine—Introduction to Machine Learning Algorithms. *Towards Data Science*.

K. Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaﬀected by shift in position. Biological Cybernetics, 36:193–202, 1980.

Liliana, D. Y. (2019, April). Emotion recognition from facial expression using deep convolutional neural network. In *Journal of Physics: Conference Series* (Vol. 1193, No. 1, p. 012004). IOP Publishing.

Matang, N. (2016). *A Facial Expression Recognition System* (Doctoral dissertation, Tribhuvan University).

Michel, P., & El Kaliouby, R. (2005). Facial expression recognition using support vector machines. In *The 10th International Conference on Human-Computer Interaction, Crete, Greece*.

Poursaberi, A., Noubari, H. A., Gavrilova, M., & Yanushkevich, S. N. (2012). Gauss–Laguerre wavelet textural feature fusion with geometrical information for facial expression identification. *EURASIP Journal on Image and Video Processing*, *2012*(1), 17.

Revina, I. M., & Emmanuel, W. S. (2018). A survey on human face expression recognition techniques. *Journal of King Saud University-Computer and Information Sciences*.

Spiers, D. L. (2016). Facial emotion detection using deep learning.

Uddin Ahmed, T., Hossain, S., Hossain, M. S., Ul Islam, R., & Andersson, K. (2019). Facial Expression Recognition using Convolutional Neural Network with Data Augmentation. In *Joint 2019 8th International Conference on Informatics, Electronics & Vision (ICIEV)*.