VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI-590018, KARNATAKA

A Project Report
on

FACIAL EMOTION DETECTION FOR ENHANCED LEARNING OF HUMAN EMOTIONS FOR AUTISTIC CHILDREN

Submitted in partial fulfillment of the requirements for the VIII Semester of degree of ***Bachelor of Engineering in Information Science and Engineering*** *of VisvesvarayaTechnological University, Belagavi*

by

Akshay P	Athira Rajeev	Deethya J Reddy	S Rakshitha
1RN19IS018	1RN19IS041	1RN19IS055	1RN19IS125

Under the Guidance of

Dr. S Sathish Kumar
Professor
Department of ISE

ESTD:2001
An Institute with a difference

Department of Information Science and Engineering

RNS Institute of Technology

Dr. Vishnuvaradhana Road, Rajarajeshwari Nagar post, Channasandra, Bengaluru-560098

2022-2023

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhana Road, Rajarajeshwari Nagar post,

Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

ESTD:2001
An Institute with a difference

CERTIFICATE

Certified that the project work entitled ***Facial Emotion Detection for Enhanced Learning of Human Emotions for Autistic Children*** has been successfully completed by **Akshay P (1RN19IS018), Athira Rajeev (1RN19IS041), Deethya J Reddy (1RN19IS055) and S Rakshitha (1RN19IS125),** bonafide students of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during the academic year **2022-2023**. The project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Dr. S Sathish Kumar Project Guide	**Dr. Suresh L** Prof. and HOD
Dr.Prakasha S / Dr.Bhagyashree Ambore Project Coordinator	**Dr. Ramesh Babu H S** Principal

External Viva

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

DECLARATION

We, **AKSHAY P [USN: 1RN19IS018], ATHIRA RAJEEV [USN: 1RN19IS041], DEETHYA J REDDY [USN: 1RN19IS055], S RAKSHITHA [USN: 1RN19IS125],** students of VIII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Project entitled ***Facial Emotion Detection for Enhanced Learning of Human Emotions for Autistic Children*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VIII Semester of degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2022-2023.

Place : Bengaluru

Date :

AKSHAY P	**(1RN19IS018)**
ATHIRA RAJEEV	**(1RN19IS041)**
DEETHYA J REDDY	**(1RN19IS055)**
S RAKSHITHA	**(1RN19IS125)**

ABSTRACT

Facial Emotion recognition is a challenging problem in the field of image analysis and computer vision. It involves the processes of extracting the facial features and then pattern recognition regardless of the background quality or variations in facial expressions or posture. Deep learning, especially CNN supports the implementation of FR technology.

ASD (Autism Spectrum Disorder) is a neurodevelopmental disorder which presents itself with impairments in social interaction and communication. Early detection and supportive therapy can aid in the management of the issues associated. The existing system of ASD therapy is a manual process and is not found to be a truly effective method. Different artificial intelligence techniques support the development of therapy and intervention tools by carefully defining the context of behaviour and emotional expressions, usage of which could help in achieving significant improvement in an ASD-affected person's social behavior and thus providing a better quality of life.

The purpose of this study was to develop a system capable of automatically detecting facial expressions through facial cues and to interface the described system with an application in order to allow social interaction with children with ASD. The facial recognition system is built using two steps. The first step is a process through which the facial features are picked up or extracted, and the second step is pattern classification. The classification of emotions is done using a Deep Convolutional Neural Network (DCNN) classifier and interfaced with an application. This option does not seek to eliminate the specialists from the therapy, but on the contrary, to facilitate their activities to achieve concentration in other exercises that could be more critical.

ACKNOWLEDGMENT

At the very onset, we would like to place our gratefulness to all those people who helped us in making this project a successful one.

Coming up with this project to be a success was not easy. Apart from the sheer effort, the enlightenment of our very experienced teachers also plays a paramount role because it is they who guide us in the right direction.

First of all, we would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of project work.

I would like to express my thanks to our Director **Dr. M K Venkatesha** for his support and inspired me towards the attainment of knowledge.

In this regard, we express our sincere gratitude to the principal **Dr. Ramesh Babu H S,** for providing us with all the facilities in this college.

We are extremely grateful to our own beloved Professor and Head of the Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize us in the right direction with all his wisdom.

We place our heartfelt thanks to **Dr. S Sathish Kumar,** Professor, Department of Information science and Engineering for having guided the project and all the staff members of our department for helping us out at all times.

We thank **Dr. Prakasha S** and **Dr. Bhagyashree Ambore,** Project coordinators, Department of Information Science and Engineering for supporting and guiding us all through.

We thank our beloved friends for having supported us with all their strength and might. Last but not the least; we thank our parents for supporting and encouraging us throughout. Wemade an honest effort in this assignment.

AKSHAY P
ATHIRA RAJEEV
DEETHYA J REDDY
S RAKSHITHA

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	-	Convolution Neural Network
ASD	-	Autism Spectrum Disorder
FER	-	Facial Emotion Recognition
DCNN	-	Deep Convolutional Neural Network
FTTC	-	Fault-Tolerant Cooperative Control
HLD	-	High-level design
HOG	-	Histogram Orientation Gradient
LoRa	-	Long Range Wireless Data Telemetry
ML	-	Machine Learning
RGB	-	Red Green Blue
SMC	-	Sliding Model Control
UAV	-	Unmanned Aerial Vehicles
VGG	-	Visual Geometry Group
WSN	-	Wireless Sensor Network

Chapter 1

INTRODUCTION

Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human visual system and enabling computers to identify and process objects in images and videos in the same way that humans do. Facial Emotion recognition presents a challenging problem in the field of image analysis and computer vision. In daily life, the role of non-verbal communication is significant; overall communication involves around 55% to 93%. The facial recognition system is built using two steps. The first step is a process through which the facial features are picked up or extracted, and the second step is pattern classification.

1.1 Overview

A facial recognition system should be able to instantly detect a face in an image or a video. This involves extracting its features and then recognising it, regardless of lighting, expression, illumination, ageing, transformations (translate, rotate and scale image) and pose, which is a difficult task. Deep learning, specifically the convolutional neural network (CNN), has recently made commendable progress in FR technology. Facial emotion analysis is efficiently used in surveillance videos, expression analysis, gesture recognition, smart homes, computer games, depression treatment, patient monitoring, anxiety, detecting lies, psychoanalysis, paralinguistic communication, detecting operator fatigue and robotics.

According to the Diagnostic and Statistical Manual of Mental Disorders, Autism Spectrum Disorder (ASD), commonly referred to as autism, is a neurodevelopmental disorder characterized by persistent deficits in social communication and social interaction across multiple contexts and restricted, repetitive patterns of behavior, interests, or activities, with these symptoms being shown in the early developmental period. In 2013 the term ASD became in an umbrella term for a set of behavior disorders, namely early infantile autism, childhood autism, Kanner's autism, high-functioning autism, atypical autism, pervasive developmental disorder, childhood disintegrative disorder, and Asperger's disorder. Furthermore, since there is no cure, early diagnosis is very important, since the sooner this disorder is detected, the sooner the treatment can begin. Treatment includes occupational therapy, applied behavioral analysis, sensory integration therapy, etc. Again, although autism is not a curable disorder, the aforementioned treatments can help decrease the social deficits associated with ASD.

Chapter 2

LITERATURE REVIEW

FaceNet2ExpNet: Regularizing a Deep Face Recognition Net for Expression Recognition (2016)

A creative concept is using static photos to train an expression recognition network. To model the high-level neurons of the expression network, we first suggest a novel distribution function. Based on this, a meticulously developed two-stage training algorithm is created. The convolutional layers of the expression net are trained in the pre-training stage, regularised by the face net; in the refining stage, fully connected layers are added to the pre-trained convolutional layers, and the entire network is trained together. The model trained using this method captures enhanced high-level expression semantics, as demonstrated via visualization. This method significantly outperforms all others it is compared with, achieving 98.6% vs the previous best of 97.3% for six classes, and 96.8% vs 92.1% for eight classes.[1]

Facial emotion recognition-based real-time learner engagement detection system in online learning context using deep learning models (2021)

It is essential to ensure that students are properly engaged throughout online learning sessions in order to make the learning environment more interactive, similar to traditional offline classrooms. This study suggests a deep learning-based method for identifying online learners' real-time involvement using facial expressions. This is accomplished by analysing the students' facial expressions throughout the online learning session to categorise their moods. The engagement index (EI), which predicts the engagement states "Engaged" and "Disengaged," is calculated using the information on face emotion recognition. The best predictive classification model for real-time engagement detection is determined by evaluating and comparing various deep learning models, including Inception-V3, VGG19, and ResNet-50. The overall effectiveness and accuracy of the suggested approach are evaluated using several benchmarked datasets as FER-2013, CK+, and RAF-DB. On benchmarked datasets as well as our custom-made dataset, experimental findings demonstrated that the suggested system achieves an accuracy of 89.11%, 90.14%, and 92.32% for Inception-V3, VGG19, and ResNet-50, respectively. In real-time learning circumstances, ResNet-50 surpasses the competition with an accuracy of 92.3% for classifying facial emotions.[2]

Adaptively Learning Facial Expression Representation via C-F Labels and Distillation (2021)

In order to correct this class imbalance, a novel adaptive supervised objective called AdaReg loss is proposed in this study. It boosts the discrimination ability of expression representations by reweighting category important coefficients. A novel coarse-fine (C-F) labelling technique is developed to lead the model from highly comparable representations that are simple to categorise to those that are challenging to do so, drawing inspiration from the cognitive mode of humans. On the basis of this, the emotional education mechanism (EEM), a novel training structure made up of a knowledgeable teacher network (KTN) and a self-taught student network, is developed to transfer knowledge (STSN). In particular, KTN combines the results of coarse and fine streams while learning expression representations ranging from simple to complex.

The STSN can optimise prospective performance and compress the original KTN under the guidance of the pre-trained KTN and existing learning experience. Extensive tests on publicly available benchmarks show that the suggested solution outperforms the state-of-the-art frameworks with results of 88.07% on RAF-DB, 63.97% on AffectNet, and 90.49% on FERPlus.[3]

Learning to Amend Facial Expression Representation via De-albino and Affinity (2021)

Amending Representation Module (ARM), a unique architecture, is proposed. ARM serves as a replacement for the pooling layer. Theoretically, it could be integrated onto any network's back end to handle padding erosion. By deconstructing facial features to make representation learning easier and reducing the weight of degraded features to counteract the negative effects of padding, ARM effectively improves facial expression representation. Public benchmark tests show that this ARM significantly improves FER's performance. The validation accuracy rates for RAF-DB, Affect-Net, and SFEW are respectively 90.42%, 65.2%, and 58.71%, exceeding those of the most recent state-of-the-art techniques.[4]

Emo-mirror: a proposal to support emotion recognition in children with autism spectrum disorders (2021)

Autism spectrum disorder (ASD) is a neurodevelopmental disorder defined as persistent difficulty in maturing the socialization process. Various artificial intelligence techniques have been used to enhance the outcomes of therapeutic interventions. Two traditional convolutional neural network (CNN) architectures are selected to test the smart mirror training process: VGG

16 and ResNet 50 [32, 33]. Concerning the outcomes: (i) While there isn't a single official technology that can be used to connect with persons with ASD, there is a trend toward using accessible technologies like desktop and mobile computers. (ii) Artificial intelligence is the main recognition method employed.

Along with the above, the most used classifier by some authors is the SVM [28, 38–40], reaching a recognition rate of 98.54% [28]. The smart mirror achieves the necessary interaction between the child and the recognition software. Together with an information system, the results obtained in each of the therapies are saved and can measure progress.

Regarding the classification model results: The different tests carried out to achieve the highest level of assertiveness among some of the most used architectures when working with images are evident, thus reaching 93.3% using the CK dataset and the VGG16 architecture.[5]

Using emotion recognition technologies to teach children with autism spectrum disorder how to identify and express emotions (2021)

In this paper, a novel system based on tactile user interfaces implemented with NFC technology and face-based emotion recognition software is presented to support the proposal on the existing literature related to specialized therapies for children with this disorder. The system is designed to assist children with ASD in recognizing and expressing their emotions. The primary interactive components of this program are new user interfaces and automatic emotion identification using the device's built-in camera (TUI). Using NFC (near-field communication) technology, natural interfaces have also been created for kids to utilize when manipulating various game-related items.

TUIs give kids a comfortable, straightforward, and logical way to engage with the game. According to experts, a serious game is created without distracting aspects so that kids' attention may be concentrated on learning how to recognize emotions in various contexts and how to express those emotions. ASD children and their psychotherapists evaluated the software program in a real-world situation, with extremely positive outcomes and the specialists' endorsement of the system as a viable tool for instructing concepts related to emotions.[6]

Learning Vision Transformer With Squeeze And Excitation For Facial Expression Recognition (2021)

As various databases of facial expressions have been made accessible over the last few decades, the Facial Expression Recognition (FER) task has gotten a lot of interest. The multiple sources of the available databases raised several challenges for facial recognition task. Mouath Aouayeb, Wassim Hamidouche et al have proposed to learn a vision Transformer jointly with a Squeeze and Excitation (SE) block for FER task. The proposed method is evaluated on different publicly available FER databases including CK+, JAFFE, RAF-DB and SFEW. Experiments demonstrate that the model outperforms state-of-the-art methods on CK+ and SFEW and achieves competitive results on JAFFE and RAF-DB.[7]

Learn From All: Erasing Attention Consistency for Noisy Label Facial Expression Recognition (2022)

Noisy label Facial Expression Recognition (FER) is more challenging than traditional noisy label classification tasks due to the inter-class similarity and the annotation ambiguity. Yuhang Zhang, Chengrui Wang et al have explored dealing with noisy labels from a new feature-learning perspective. The proposed system finds that FER models remember noisy samples by focusing on a part of the features that can be considered related to the noisy labels instead of learning from the whole features that lead to the latent truth. Inspired by that, proposed a novel Erasing Attention Consistency (EAC) method to suppress the noisy samples during the training process automatically. Specifically, first utilize the flip semantic consistency of facial images to design an imbalanced framework. Then randomly erase input images and use flip attention consistency to prevent the model from focusing on a part of the features.[8]

Optimal Facial Feature Based Emotional Recognition Using Deep Learning Algorithm (2022)

Humans have traditionally found it simple to identify emotions from facial expressions, but it is far more difficult for a computer system to do the same. Automatic emotion recognition has been the subject of numerous studies, most of which use a machine learning methodology. Tarun Kumar Arora, Pavan Kumar Chaubey et al have we improved the convolutional neural network technique to identify 7 fundamental emotions and evaluated several preprocessing techniques to demonstrate how they affected the CNN performance. is research focuses on improving facial features and expressions based on emotional recognition. By identifying or recognising facial expressions that elicit human responses, it is possible for computers to make more accurate predictions about a person's mental state and to provide more tailored responses. As a result, we examine how a deep learning technique that employs a convolutional neural network might improve the detection of emotions based on facial features (CNN). As a result, the proposed model reveals the same seven emotions of the facial acting coding system.[9]

Facial Emotion Detection And Recognition (2022)

Facial emotional expression is a part of face recognition, it has always been an easy task for humans, but achieving the same with a computer algorithm is challenging. With the recent and continuous advancements in computer vision and machine learning, it is possible to detect emotions in images, videos, etc. Amit Pandey, Aman Gupta, Radhey Shyam proposed a face expression recognition method based on the Deep Neural Networks especially the convolutional neural network (CNN) and an image edge detection is proposed. The edge of each layer of the image is retrieved in the convolution process after the facial expression image is normalized. To maintain the texture picture's edge structure information, the retrieved edge information is placed on each feature image. In this research, several datasets are investigated and explored for training expression recognition models. The suggested method uses training sample image data to directly input the picture pixel value. The ability to accurately determine emotions was greatly enhanced by the removal of the background.[10]

Facial Emotion Recognition Using Transfer Learning in the Deep CNN (2021)

Human facial emotion recognition (FER) has attracted the attention of the research community for its promising applications. Mapping different facial expressions to the respective emotional states are the main task in FER. The classical FER consists of two major steps: feature extraction and emotion recognition. M.A.H.Akhand, Shuvendu Roy et al proposed a very Deep CNN (DCNN) modeling through Transfer Learning (TL) technique where a pre-trained DCNN model is adopted by replacing its dense upper layer(s) compatible with FER, and the model is fine-tuned with facial emotion data. A novel pipeline strategy is introduced, where the training of the dense layer(s) is followed by tuning each of the pre-trained DCNN blocks successively that has led to gradual improvement of the accuracy of FER to a higher level. The evaluation results reveal the superiority of the proposed FER system over the existing ones regarding emotion detection accuracy. [11]

Emotion Recognition Based on Facial Expressions Using Convolutional Neural Network (CNN) (2021)

Over the last few years, there has been an increasing number of studies about facial emotion recognition because of the importance and the impact that it has in the interaction of humans with computers. With the growing number of challenging datasets, the application of deep learning techniques have all become necessary. Sabrina Begaj, Ali Osman Topal, Maaruf Ali have studied the challenges of Emotion Recognition Datasets and also try different parameters and architectures of the Conventional Neural Networks (CNNs) in order to detect the seven emotions in human faces, such as: anger, fear, disgust, contempt, happiness, sadness and surprise. Have chosen iCV MEFED (Multi-Emotion Facial Expression Dataset) as the main dataset for the study, which is relatively new, interesting and very challenging. The aim in this work were to go in detail in every step of Deep Learning, starting from finding the dataset and ending up analyzing the results. The work was separated into two main parts: Dataset and the construction of the CNN.[12]

Facial Emotion Recognition Using a Novel Fusion of Convolutional Neural Network and Local Binary Pattern in Crime Investigation (2022)

The exploration of facial emotion recognition aims to analyze the psychological characteristics of juveniles involved in crimes and promote the application of deep learning to psychological feature extraction. Dimin Zhu, Yuxi Fu et al have proposed a facial emotion recognition model is constructed by increasing the layers of the convolutional neural network (CNN) and integrating CNN with several neural networks such as VGGNet, AlexNet, and LeNet-5. Second, based on the feature fusion, an optimized Central Local Binary Pattern (CLBP) algorithm is introduced into the CNN to construct a CNN-CLBP algorithm for facial emotion recognition. The validity analysis is conducted on the algorithm after the preprocessing of face images and the optimization of relevant parameters.[13]

Methods for Facial Expression Recognition with Applications in Challenging Situations (2022)

In the last few years, a great deal of interesting research has been achieved on automatic facial emotion recognition (FER). FER has been used in a number of ways to make human-machine interactions better, including human centre computing and the new trends of emotional artificial intelligence (EAI). Anil Audumbar Pise, Mejdal A. et al have addressed the latest advances in computational intelligence-related automated emotion recognition using recent deep learning models. They show that both deep learning-based FER and models that use architecture-related methods, such as databases, can collaborate well in delivering highly accurate results. They emphasize the fact that machines are already able to recognize more complex emotions, implying that the emergence of human-machine collaboration will become more and more commonplace.[14]

Face Recognition and Identification using Deep Learning Approach (2020)

Generally, the human recognition system involves 2 phases - face detection and face identification. KH Teoh, RC Ismail et al have described the concept of how to design and develop a face recognition system through deep learning using OpenCV in Python and it includes the operations of automatically detecting followed by verifying a person from either picture or video. Haar feature-based cascade classifiers are used along with TensorFlow as the framework. They show that the distance between the face and camera, the background, the lighting intensity and the variation in the orientation of the face in training images affect the recognition process. [15]

FECTS: A Facial Emotion Cognition and Training System for Chinese Children with Autism Spectrum Disorder (2022)

Traditional training methods such as card teaching, assistive technologies (e.g., augmented reality/virtual reality games and smartphone apps), DVDs, human-computer interactions, and human-robot interactions are widely applied in autistic rehabilitation training in recent years. Guobin Wan, Fuhao Deng et al propose a novel framework for human-computer/robot interaction and introduce a preliminary intervention study for improving the emotion recognition of Chinese children with an autism spectrum disorder. The core of the framework is the Facial Emotion Cognition and Training System (FECTS, including six tasks to train children with ASD to match, infer, and imitate the facial expressions of happiness, sadness, fear, and anger) based on Simon Baron-Cohen’s E-S (empathizing-systemizing) theory. The training data and the interactive data during the preliminary study are all recorded using the FECTS and are uploaded, analysed, and visualized by a cloud-based evaluation system.[16]

Facial Emotion Recognition for Autism Children (2020)

People can accurately identify a standard face and understand face expression with a single glance. However, children with Autism Spectrum Disorder (ASD) often have problems communicating and socializing. K.Prem Kumar, K. Murugapriya et al propose Emotion detection for Autism spectrum disorder children (ASD). For a better involvement of the children’s social behaviour, here a face is captured in real-time and age, gender and emotions are predicted by Facial expression recognition (FER) using the Haar Cascade Classifier and CAFFE model along with Keras, Tensorflow, OpenCV, dlib and numpy. The model is used to detect emotion in fractions and the trained model will generate the output and display which emotion has the highest value.[17]

Emotion Detection of Autistic Children Using Image Processing (2019)

Pooja Rani has studied emotion detection of autistic children from facial expressions and works on four emotions, namely sad, happy, neutral, and angry. Detection of the emotions of autistic children is performed utilizing image processing and machine learning algorithms and two techniques, support vector machine and neural network are employed. The features are extracted from the faces of autistic children with the local binary pattern. An overall accuracy of 90% was achieved local binary pattern + support vector machine method and 70% using local binary pattern + neural network method.[18]

Challenges in Representation Learning: A report on three machine Learning contests (2013)

This paper describes three machine learning contests that were held as part of the ICML workshop “Challenges in Representation Learning.” The purpose of the workshop, organized by Ian Goodfellow et al was to explore latest developments in representation learning. In the Facial Expression recognition challenge, Facial Expression Regression 2013 (FER-2013) dataset was used. Where OpenCV face recognition was used later the cropped images were then resized to 48x48 pixels and converted to grayscale where they mapped fine-grained emotion keywords in to same seven broad categories used in Toronto Face Database. They also found that the human accuracy on the FER-2013 dataset was 65%.[19]

Covariance Pooling for Facial Expression Regression (2018)

Facial expressions play an important role in communicating the state of our mind. Both humans and computer algorithms can greatly benefit from being able to classify facial expressions. Classifying facial expressions into different categories requires capturing regional distortions of facial landmarks. Dinesh Acharya et al first employed a kind of manifold networks in conjunction with traditional convolutional networks for spatial pooling within individual image feature maps in an end-to-end deep learning manner. By doing so, they were able to achieve a recognition accuracy of 58.14% on the validation set of Static Facial Expressions in the Wild (SFEW 2.0) and 87.0% on the validation set of Real-World Affective Faces (RAF) Database. [20]

Exploring Emotion Features and Fusion Strategies for Audio – Video Emotion Recognition (2020)

The audio-video based emotion recognition aims to classify a given video into basic emotions. In this paper, Hengshun Zhou et al have described their approaches in EmotiW 2019, which mainly explores emotion features and feature fusion strategies for audio and visual modality. They have explored both speech-spectrogram and Log Mel-spectrogram to evaluate several facial features with different CNN Models and they also have explored intra-modal and cross-modal fusion methods for fusion strategies. With careful evaluation, they obtained and accuracy of 65.5% on the AFEW validation set and 62.48% on the test set and rank second in the challenge. [21]

Facial Emotion Recognition Using Convolution Neural Networks (FERC) (2020)

Facial expression for emotion detection has always been an easy task for humans, but achieving the same task with a computer algorithm is quite challenging. With the recent advancement in computer vision and machine learning, it is possible to detect emotions from images. In this paper, Ninad Mehendale er al have proposed a novel technique called facial emotion recognition using convolutional neural networks (FERC). FERC uses the advantages of CNN and supervised learning (feasible due to big data). The main advantage of the FERC algorithm is that it works with different orientations (less than 30°) due to the unique 24-digit long EV feature matrix. The FERC is based on two-part convolutional neural network (CNN): The first-part removes the background from the picture, and the second part concentrates on the facial feature vector extraction. On comparing FERC with other algorithms like Alexnet, VGG, GoogleNet and Resnet, FERC provided the highest accuracy score of 96%.[22]

Chapter 3

ANALYSIS

3.1 Existing System and Limitations

The recognition of facial emotions is a component of social cognition and is essential for effective communication and social interaction. Children with ASD have deficiencies in initiating and responding to social or emotional interaction. As a result, they have difficulty recognizing and understanding the emotions and mental states of others. Their difficulty with emotional recognition can generate unfavorable consequences over time, which can lead to psychiatric comorbidity, poor occupational performance, and problems in social relationships.

The existing system of emotion recognition therapies are manual processes, which use tangible products with fixed settings and which today represent a danger due to the possible transmission of diseases. In addition, this traditional process sometimes does not generate motivation in patients; on the contrary, it makes them feel intimidated.

The main limitations to assessing Emotional Regulation in individuals with ASD: on the one hand, the reliability of clinical measurements is still under debate whereas, on the other hand, methods based on observations are affected by difficult-to-interpret emotions without defining the context of the child's baseline behaviors and emotional expressions. Helping their emotion recognition skills through intervention tools could significantly improve these children's social interactions. Therefore, different artificial intelligence techniques have to be applied to improve the results obtained in these therapies.

3.2 Problem Identification

Facial Emotion Recognition (FER) is the technology that analyses facial expression from both static images and videos in order to reveal information on one's emotional state.

Emotions are the key to understanding human interactions, especially those conveyed with facial expressions. In overall communication, the involvement of non-verbal communication is significant, around 55% to 93%.

Figure 3.1 Process of Emotion Recognition

The existing FER system as shown in Fig 3.1 lacks to provide consistent overall accuracy and it cannot be used along with other models for its better application i.e., lack of interoperability. Hence, we propose the use of Deep Convolutional Neural Network(DCNN) to achieve high and consistent accuracy.

Here in this project, we are creating a system in which we will use the Facial Emotion Recognition (FER) as a backbone and will integrate it into an app so that it will be helpful for people with ASD to better understand human emotions and learn to interact with people accordingly.

3.3 Objectives

The main objective of the proposed project is to develop a model for Facial Emotion Recognition (FER) which detects emotions and the emotion to be displayed on the screen in a text or graphic format so as that the people with autism spectrum disorder (ASD) can better understand the emotions of humans and learn to interact with them accordingly.

The features that we are implementing in this project are as follows:

- To input images from dataset and to classify the emotions using Deep Convolutional Neural Network (DCNN) to achieve high and consistent accuracy.
- Once the emotion is detected, we will integrate it with an application which scans the real-time image or video of the person and will display the emotion of that person in a text or graphic manner.

3.4Proposed System

Recent advances in computer vision and machine learning brought more and more affordable solutions for facial analysis that paved the way for developing non-invasive technological frameworks which can be applied to extract facial measurements in a non-invasive way. This option does not seek to eliminate the specialists from the therapy, but to facilitate their activities to achieve concentration in other exercises that could be more critical.

The proposed system for facial expression recognition provides a personalized interface to help children with autism identify and understand human emotions. Human emotions are basically represented in different classes of emotions, for example, happy, sad, natural, anger, surprise, hatred and fear. There are three stages in detecting emotions, selection of images, database training, and emotions classification. The classification of emotions is done using DCNN classifier and interfaced with an application.

Deep Convolutional Neural Networks (CNNs) are a class of artificial neural networks that are commonly used in computer vision tasks such as image recognition, object detection, and segmentation. The architecture of a Deep Convolutional Neural Network is composed of multiple layers of convolutional and pooling operations, followed by fully connected layers for classification. These layers are designed to learn increasingly complex and abstract features of the input image, with the early layers detecting simple features such as edges and corners, and the later layers recognizing more complex structures such as textures and object parts. DCNNs have achieved state-of-the-art performance in various computer vision tasks and are widely used in real-world applications such as self-driving cars, medical image analysis, and video surveillance.

Chapter 4

METHODOLOGY

4.1 System implementation

Structural Design

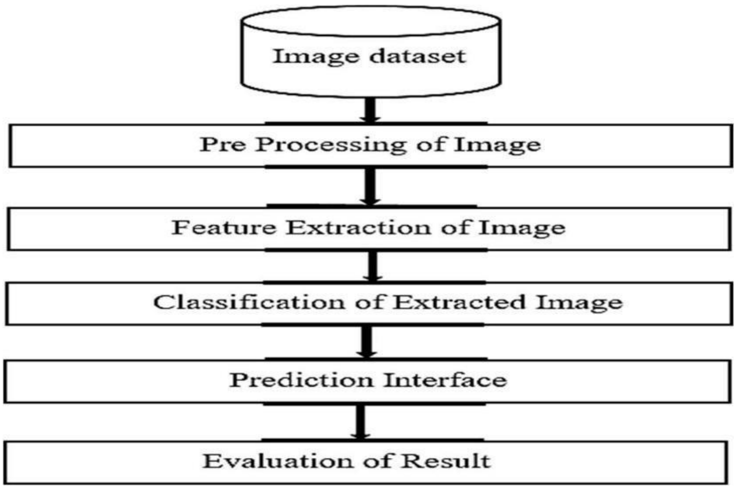


Figure 4.1 Structural chart of the proposed system

The figure 4.1 shows that the proposed system involves the following steps. First step involves pre-processing of captured images. The pre-processed image undergoes feature extraction, where various features of the fire and smoke and fire types are extracted and certain algorithms are applied.The data that is stored is compared with the pre-processed image and approximate result is generated**.**

Convolution Neural Network

Convolutional neural network is the special type of feed forward artificial neural network in which the connectivity between the layers is inspired by the visual cortex. Convolutional Neural Network (CNN) is a class of deep neural networks which is applied for analyzing visual imagery. They have applications in image and video recognition, image classification, natural language processing etc. Convolution is the first layer to extract features from an input image. Convolution preserves the relationship between pixels by learning image features using small squares of input data.It is a mathematical operation that takes two inputs such as image matrix and a filter or kernel. Eachinput image will be passed through a series of convolution layers with filters (kernels) to produce output feature maps. Here is how exactly the CNN works.

Basically, the convolutional neural networks have 4 layers that is the convolutional layers, eLU layer, pooling layer and the fully connected layer.

Convolutional Layer

In convolution layer after the computer reads an image in the form of pixels, then with the help of convolution layers we take a small patch of the images. These images or patches are called the features or the filters. By sending these rough feature matches is roughly the same position in the two images, convolutional layer gets a lot better at seeing similarities than whole image matching scenes. These filters are compared to the new input images if it matches then the image is classified correctly. Hereline up the features and the image and then multiply each image, pixel by the corresponding feature pixel, add the pixels up and divide the total number of pixels in the feature. We create a map and put the values of the filter at that corresponding place. Similarly, we will move the feature to every other position of the image and will see how the feature matches that area. Finally, we will get a matrix as an output.

eLU Layer

eLU layer is nothing but the rectified linear unit, in this layer we remove every negative value from the filtered images and replaces it with zero. This is done to avoid the values from summing up to zeroes. This is a transform function which activates a node only if the input value is above a certain number while the input is below zero the output will be zero then remove all the negative values from the matrix.

Pooling Layer

In this layer we reduce or shrink the size of the image. Here first we pick a window size, then mention the required stride, then walk your window across your filtered images. Then from each window take the maximum values. This will pool the layers and shrink the size of the image as well as the matrix. The reduced size matrix is given as the input to the fully connected layer.

Fully Connected Layer

We need to stack up all the layers after passing it through the convolutional layer, ReLU layer and the pooling layer. The fully connected layer used for the classification of the input image. These layers need to be repeated if needed unless you get a 2x2 matrix. Then at the end the fully connected layer is used where the actual classification happens.

4.2 Typical CNN Architecture

CNN architecture is inspired by the organization and functionality of the visual cortex and designed to mimic the connectivity pattern of neurons within the human brain. The neurons within aCNN are split into a three-dimensional structure, with each set of neurons analyzing a small region orfeature of the image.

In other words, each group of neurons specializes in identifying one part of the image. CNNsuse the predictions from the layers to produce a final output that presents a vector of probability scoresto represent the likelihood that a specific feature belongs to a certain class. Figure 4.2 shows the Typical CNN Architecture

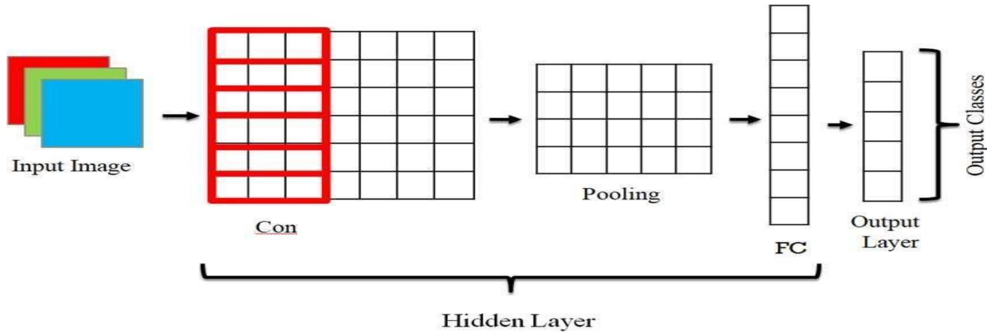


Figure 4.2 Typical CNN Architecture

As shown in Fig 4.3, CNN is composed of several kinds of layers:

- **Convolutional layer**- In convolution layer after the computer reads an image in the form of pixels, then with the help of convolution layers we take a small patch of the images. These images or patchesare called the features or the filters. By sending these rough feature matches is roughly the same position in the two images, convolutional layer gets a lot better at seeing similarities than whole image matching scenes. It creates a feature map to predict the class probabilities for each feature byapplying a filter that scans the whole image, few pixels at a time.
- **Pooling layer (down sampling)**- scales down the amount of information the convolutional layer generated for each feature and maintains the most essential information (the process of the convolutional and pooling layers usually repeats several times).
- **Fully connected layer**- "flattens" the outputs generated by previous layers to turn them into a singlevector that can be used as an input for the next layer. Applies weights over the input generated by the feature analysis to predict an accurate label.
- **Output layer**- generates the final probabilities to determine a class for the image.

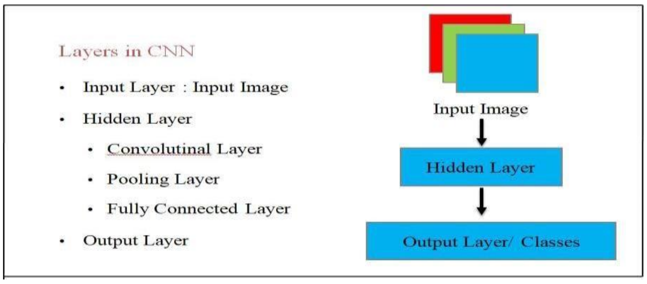


Fig. 4.3 Layers in CNN

Convolutional Layer

Convolutional Layer is the first step in CNN, here 3*3 part of the given matrix which was obtained from High-pass filter is given as input. That 3*3 matrix is multiplied with the filter matrix for the correspondingposition and their sum is written in the particular position. This is shown in the below figure. This output is given to pooling layer where the matrix is further reduced. Figure 4.4 shows the Convolutional Layer.

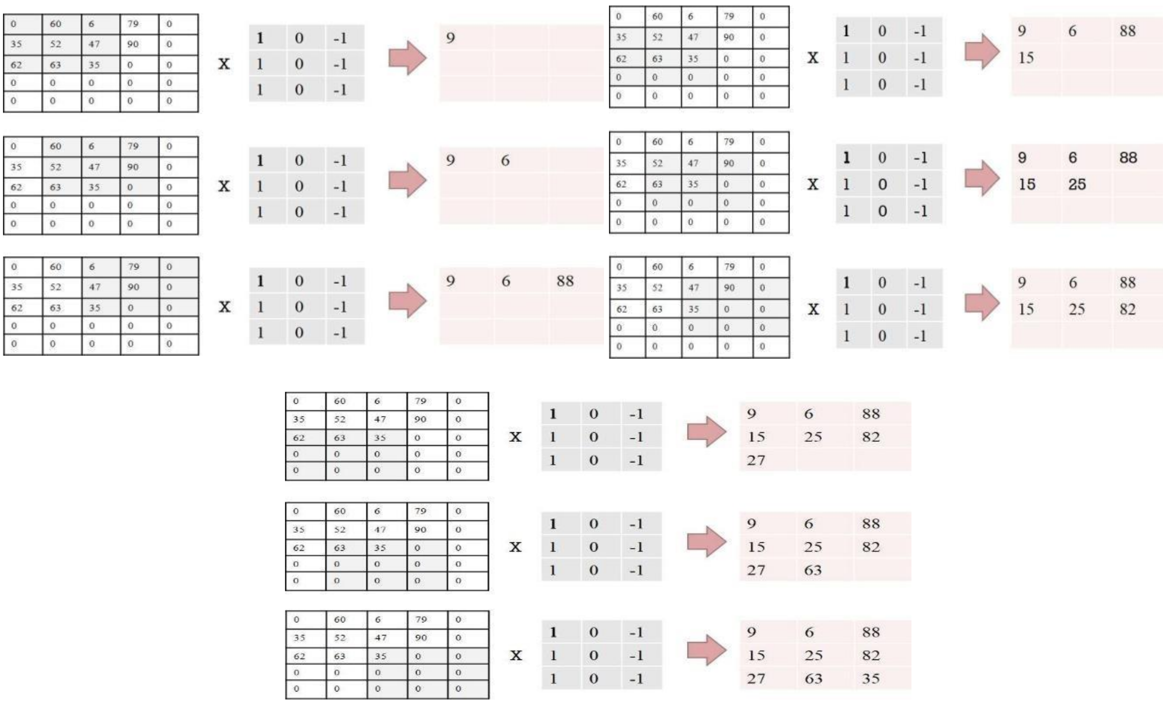


Figure 4.4 Convolutional Layer

Convolution is followed by the rectification of negative values to 0s, before pooling. Here, it is notdemonstrable, as all values are positive. In fact, multiple iterations of both are needed before pooling.

Pooling Layer

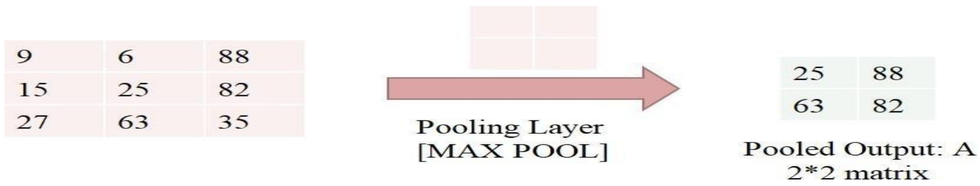


Figure 4.5 Pooling Layer

In Pooling layer 3*3 matrix is reduced to 2*2 matrix, this is done by selecting the maximum of the particular 2*2 matrix for the particular position. Figure 4.5 shows the Pooling Layer.

Fully connected layer and Output Layer

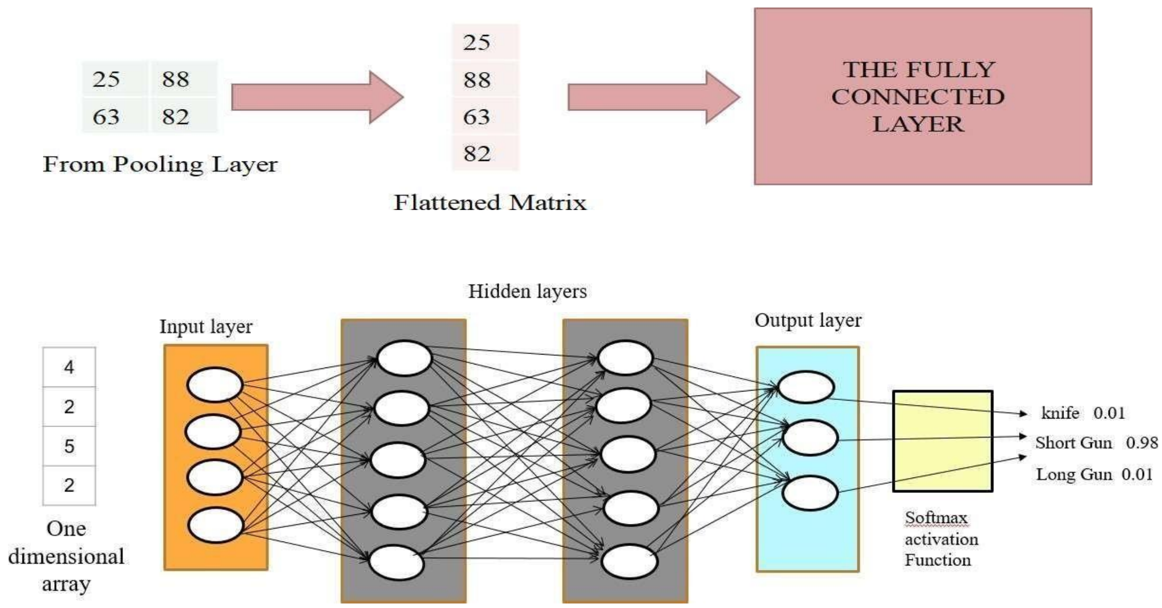


Figure 4.6 Fully connected layer and Output Layer

The output of the pooling layer is flattened and this flattened matrix is fed into the Fully Connected Layer. In the fully connected layer there are many layers, Input layer, Hidden layer and Output layers are parts of it. Then this output is fed into the classifier, in this case SoftMax ActivationFunction is used to classify the image into smoke and fire present or not. Figure 4.6 shows the fullyconnected layer and Output Layer.

4.3 DCNN Architecture

A typical Deep Convolutional Neural Network (DCNN) architecture consists of several layers that are stacked on top of each other. These layers work together to process input data and learn features at different levels of abstraction. The most commonly used layers in a DCNN are:

Convolutional Layers: These are the primary layers in a DCNN that perform convolution operations on input data. Convolution involves applying a set of learnable filters to the input data to extract local patterns or features. Each filter produces a feature map, which represents the presence of a specific pattern in the input data. Convolutional layers are responsible for learning meaningful spatial hierarchies and detecting local patterns, such as edges, corners, and textures.

Pooling Layers: These layers are used to reduce the spatial dimensions of the feature maps produced by the convolutional layers. Pooling helps to down sample the feature maps and reduce the amount of computation required, while also capturing the most salient features. Common pooling operations include max pooling, average pooling, and global average pooling.

Activation Layers: These layers apply an activation function to the output of the convolutional or pooling layers, introducing non-linearity into the network. Common activation functions include ReLU (Rectified Linear Unit), ELU (Exponential Linear Unit), sigmoid, and tanh. Activation functions introduce non-linearities, which allow the network to learn complex, non-linear patterns in the data.

Fully Connected Layers: These layers are used to make predictions based on the learned features. Fully connected layers are typically placed at the end of the network and take the flattened feature maps as input. They are responsible for mapping the learned features to the final output classes or regression targets. Fully connected layers are also known as dense layers.

Dropout Layers: These layers are used for regularization and preventing overfitting in the network. Dropout randomly sets a fraction of the input units to 0 at each training iteration, which helps to prevent the network from relying too heavily on any single input unit and encourages the network to learn more robust and generalized features.

Batch Normalization Layers: These layers are used to normalize the activations in a network by scaling and shifting them during training. Batch normalization helps to accelerate training and improve the stability of the network by reducing the internal covariate shift, which is a change in the distribution of the network's inputs during training.

Flatten Layer: This layer is used to convert the multi-dimensional feature maps produced by the convolutional layers into a flattened, one-dimensional vector that can be used as input to the fully connected layers.

These are some of the common layers used in a typical DCNN architecture. The specific architecture and number of layers used in a DCNN can vary depending on the task, data, and desired performance. Experimentation and tuning of the architecture are often done to optimize the network's performance for a specific task.

4.4 Proposed Architecture

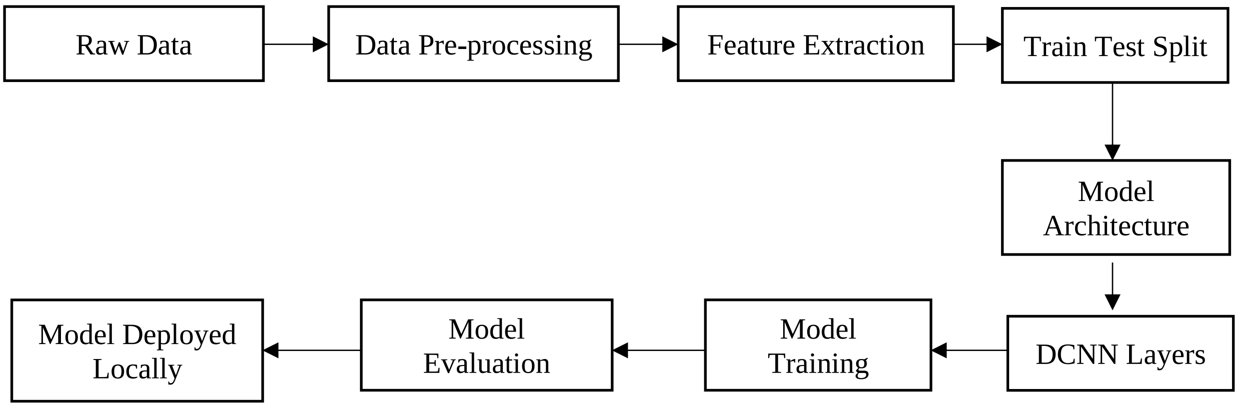


Figure 4.7 Proposed Architecture

The figure 4.7 depicts the proposed architecture for this project where in the following process take place:

- Input of Raw Data
- Data Pre-processing
- Feature Extraction
- Train Test Split
- Input of Processed Data to Model Architecture
- DCNN Layer application
- Model Training
- Model Evaluation
- Model Deployed Locally

Chapter 5

SYSTEM DESIGN

5.1 System requirements

System requirement specifications gathered by extracting the appropriate information to implement the system. It is the elaborative conditions which the system need to attain. Moreover, the SRS delivers a complete knowledge of the system to understand what this project is going to achieve without any constraints on how to achieve this goal. This SRS not providing the information to outside characters but it hides the plan and gives little implementation details.

Specific Requirement

- Require access to a client session of Python and Keras toolbox for job submission.
- A shared file system between user desktops and cluster.
- Maximum of Python worker per physical CPU core.

Hardware Requirement

- Processor: Intel core
- Processor Speed: 1.86 GHz.
- RAM: $4GB^{+}$
- Hard Disk Space: 500 GB^{+}
- Monitor: 15 VGA Color

Software Requirement

- Operating system: Windows 10
- Coding Language: Python
- Software Tool: NumPy, Pandas, SKLearn, Keras, TensorFlow
- Toolbox: Image processing Toolbox

5.2 Functional and Non-Functional Requirements

5.2.1 FUNCTIONAL REQUIREMENTS:

- System should do minimal computations on its own.
- System should capture image.
- System should automatically detect emotions on its own.

5.2.2 NON-FUNCTIONAL REQUIREMENTS:

- The Camera is used take video.
- Requirement data will be stored in the python database.
- System should be reliable.
- System should be flexible for future enhancements.
- System should be Easily Implementable.
- System should be Easy to Implement.
- Cost of Implementation should be low.

5.3 System Design

- The proposed system includes five modules. The initial stage is the image acquisition stage through which the real-world sample is recorded in its digital form using a digital camera.
- In the next stage of the research image was subjected to a pre-processing stage. Making use of its clarity, the blurred images are removed from the dataset.
- The feature extraction aspect of an image analysis focuses on identifying inherent features of the objects present within an image.
- Expression analysis is performed to gather the features extracted and comparing it with the training set examples.
- Classification maps the data into specific groups or classes.

The block diagram of Facial Emotion Recognition is shown below in Fig 5.1.

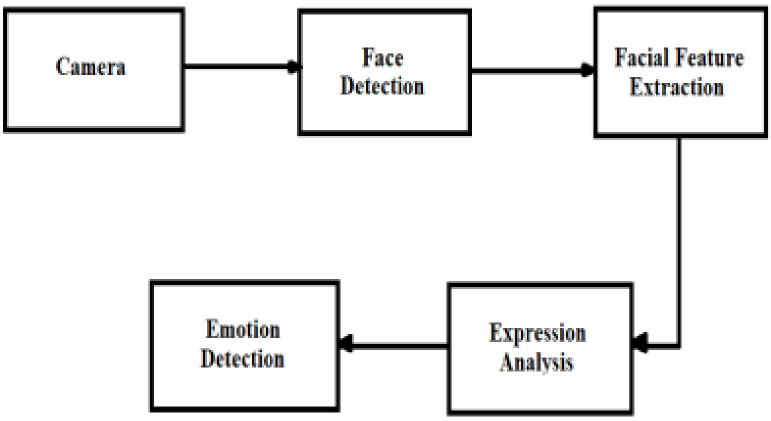


Fig 5.1 Block Diagram of FER

High-level design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces. The HLD uses possibly nontechnical to mildly technical terms that should be understandable to the administrators of the system. In contrast low level design further exposes the logical detailed design of each of these elements for programmers.

High level design is the design which is used to design the software related requirements. In this chapter complete system design is generated and shows how the modules, sub modules and the flow of the data between them are done and are integrated. It consists of very simple phases and shows the implementation process.

Design Consideration:

The design consideration briefs about how the system behaves for the boundary environments and what action should be taken if the abnormal case happens. Some of the design considerations are data collection, pre-processing methods and Classification and prediction.

The design considerations are formulated to bring to the attention of the designers in applying the universal accessibility design principles and requirements to buildings and facilities. They can also be used to identify barriers in existing systems.

The proposed system has the following steps for facial emotion detection

i. Image Pre-Processing
ii. Identification
iii. Feature Extraction
iv. Emotion Detection

- ***Image Pre processing***

  The image processing is a mechanism that focuses on the manipulation of images in different ways in order to enhance the image quality. Images are taken as the input and output for image processing techniques. The images are reduced in size so that the values are useful for feature extraction and identification of region of interest.

- ***Identification***

  In this stage identify the region which needs to proceed for further process, it is involved in the identification of the particular region of the image that is used for the further processlike feature extraction and classification of the images. The output of the pre-processing step is given as the input for the identification process. The region of interest obtained by the pre-processing of the images. That region is considered as proceeding part of the image from which emotion will be identified. The identified facial images are given to the feature extraction process.

- ***Feature Extraction***

  In this stage extract the required feature from the identified region which are obtained from the previous step. That region is compressed by converting reduced size matrix to control over fitting. The reduction of the matrix size helps in reduce the memory size of the images. Then the flattening process is applied to the reduced matrix, in which the reduced matrix is converted to one-dimension array, which is used for final detection.

- ***Emotion Detection***

  The methodology is proposed using DCNN (Deep Convolutional Neural Networks) model. After the feature extraction takes place, feature scaling is performed. The dataset is split into train and test datasets. Using DCNN model predictions are provided based on the training dataset. The emotion is classified either as happy, sad, surprise, angry,neutral, etc.

5.4 System Architecture

A system architecture is the conceptual model that defines the structure, behavior, and more views of a system. A system architecture can consist of system components and the sub-systems developed, that will work together to implement the overall system.

The figure 5.2 shows the system architecture for the proposed system. The input imageis pre- processed to get the clear vision of the image. In the next step identifies the part which needs to proceed further. Then required feature are extracted by In the CNN convolution layer. By passing those features into different layer of CNN we get compressed image, that feature is used for detection of the facial emotion.

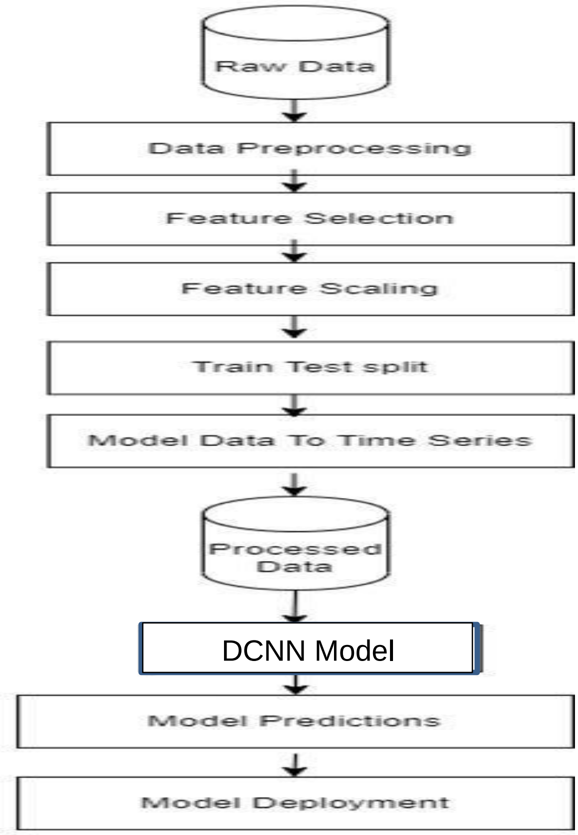


Figure 5.2 System Architecture of the Facial Emotion detection

- ***Module Specification:***

Module Specification is the way to improve the structure design by breaking down the system into modules and solving it as independent task. By doing so the complexity is reduced and the modules can be tested independently. The number of modules for our model is three, namely pre- processing, identification, feature extraction and detection.

This project has four sets in the emotion detection system as shown below figure. So, each phase signifies the functionalities provided by the proposed system. In the data pre-processing stage, the blurred images and images containing insufficient data are removed. The second phase is to extract the feature from the identified region in the convolution layer of CNN. This includes the part of image which is considered as a required part of image which is used for the detection of the emotion. All the required information of the image is convertedinto pixel and stored in the form of image.

In the final phase each feature from the previous phase is considered these features are extracted from the convolution layer of the CNN and sent to fully connected layer. Apply artificial neural network to those features by continuous iteration. In the hidden layer of ANN each feature is efficiently identified and finally get the prediction. Based on that value emotion will be detected.

- ***Specifications using use case diagrams:***

A use case is a set of scenarios that describing an interaction between a source and a destination. A use case diagram displays the relationship among actors and use cases. The two main components of a use case diagram are use cases and actors.

The figure 5.3 shows that the use case diagram in the Unified Modelling Language (UML) is a type of behavioral diagram defined by and crated from a use-case analysis. Here the user can collect the data and load the data to the system. The system can store the data for training and testing the model, here system is taken as actor. The training and testing data are given to the CNN for further classification.

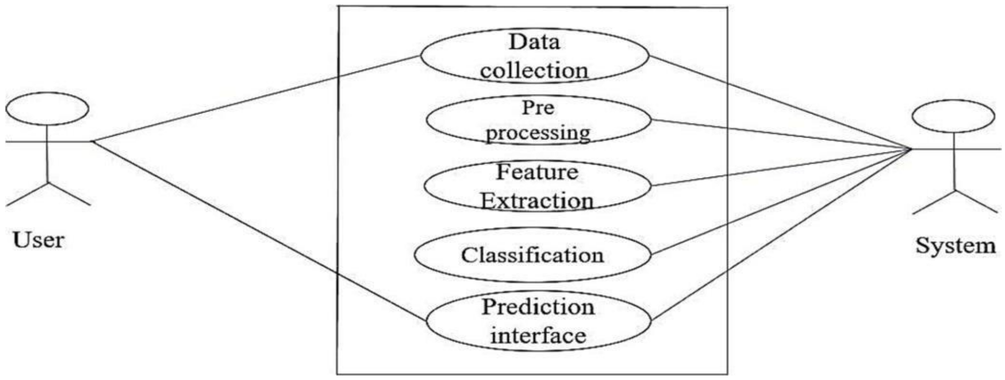


Figure 5.3 Use case diagram of the proposed system

Classification of data done by different layers of CNN. The feature extraction is done by convolution layer of the CNN and then using the Artificial Neural Network in the fully connected layer the expression can be identified. The detection is based on the prediction value calculated by using Deep Convolutional Neural Network. Based on prediction value emotion will be detected.

5.5 Data Flow Diagram

A data flow diagram (DFD) is graphic representation of the "flow" of data throughan information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context level DFD first which shows the interaction between the system and outside entities.

Data flow diagrams show the flow of data from external entities into the system, how the data moves from one process to another, as well as its logical storage. There are only four symbols:

1. Squares representing external entities, which are sources and destinations of information entering and leaving the system.
2. Rounded rectangles representing processes, in other methodologies, may be called 'Activities', 'Actions', 'Procedures', 'Subsystems' etc. which take data as input, do processingto it, and output it.
3. Arrows representing the data flows, which can either, be electronic data or physical items. Itis impossible for data to flow from data store to data store except via a process, and external entities are not allowed to access data stores directly.
4. The flat three-sided rectangle is representing data stores should both receive information for storing and provide it for further processing.
5. It is also used to analyses a particular problem and the solution for it in steps.
6. A user loads the data and the system reads the data provided by the user.
7. Based on feature extraction and classifier the model will be trained and tested.

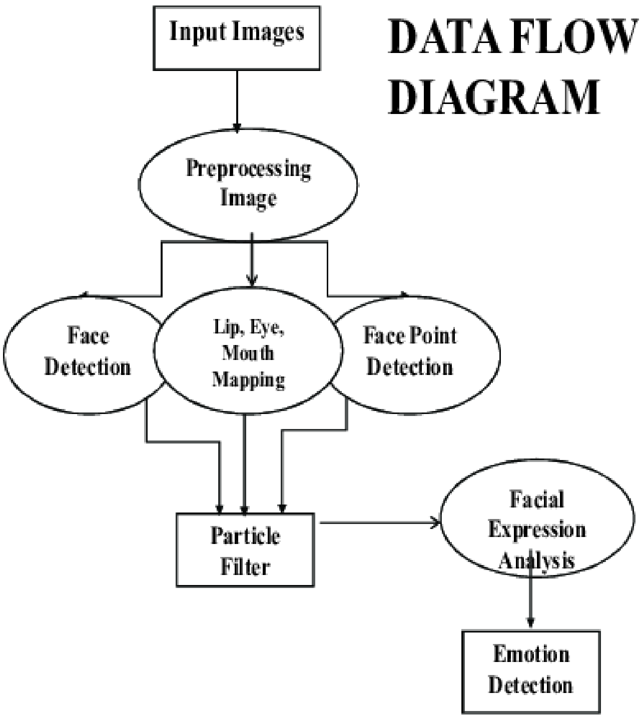


Figure 5.4 Data Flow Diagram of the Facial Emotion detection

The figure 5.4 shows the entire process of Facial Emotion Recognition where thefollowing process takes place:

1. Input of images
2. Pre-processing
3. Face Detection
4. Mapping of features of the face
5. Face point detection
6. Particle filter
7. Facial Expression Analysis
8. Emotion Detection

5.6 Flowchart Diagram

A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows.

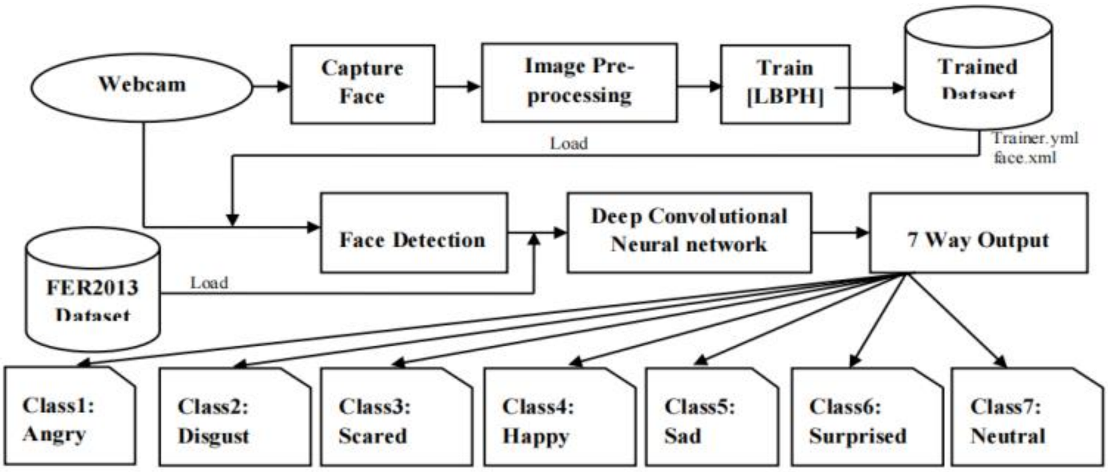


Fig 5.5 System diagram for facial emotion detection

A webcam is used to capture, identify, and recognize the facial expressions of a person, which is done through the use of software. In the camera, a rectangular frame on the face area is obtained; this identification of the face region from a non-facial region is accomplished by the employment of the Viola Jones method, the LBPH Face Recognizer algorithm, and the Haarcascade frontal face dataset, among other techniques.

During the Face Detection process: A trained dataset is used to match the face in a video camera with the face in the dataset. In order to classify the obtained face, convolutional neural networks are used in conjunction with the FER2013 database to do the classification. The facial expression represents the chance of acquiring the maximum level of expression based on the characteristics of the individual. One of seven possible facial expressions is presented in conjunction with the recognized picture of the subject. Figure 5.6 depicts the flowchart for this model.

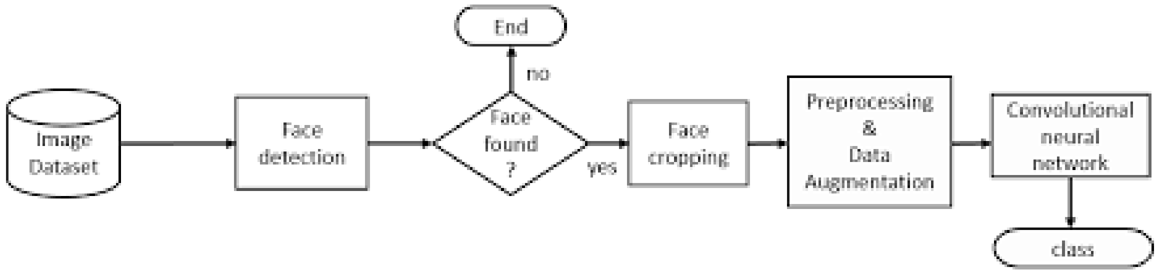


Fig 5.6 Flowchart for facial emotion detection

5.6.1 Flowchart of model

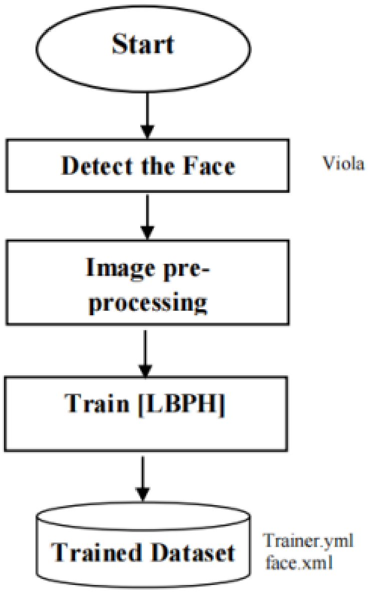


Fig 5.7 Flowchart of training

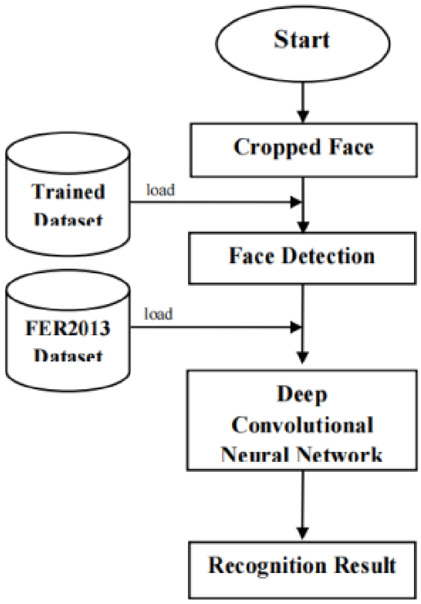


Fig 5.8 Flowchart of testing

During training Phase, the system received a training data comprising grayscale images of faces with their respective expression label and learns a set of weights for the network. The training step took as input an image with a face. Thereafter, an intensity normalization is applied to the image. The normalized images are used to train the Convolutional Network.

To ensure that the training performance is not affected by the order of presentation of the examples, validation dataset is used to choose the final best set of weights out of a set of trainings performed with samples presented in different orders.

The output of the training step is a set of weights that achieve the best result with the training data. During test, the system received a grayscale image of a face from test dataset, and output the predicted expression by using the final network weights learned during training. Its output is a single number that represents one of the seven basic expressions.

5.7 Frame structure diagram of DCNN model

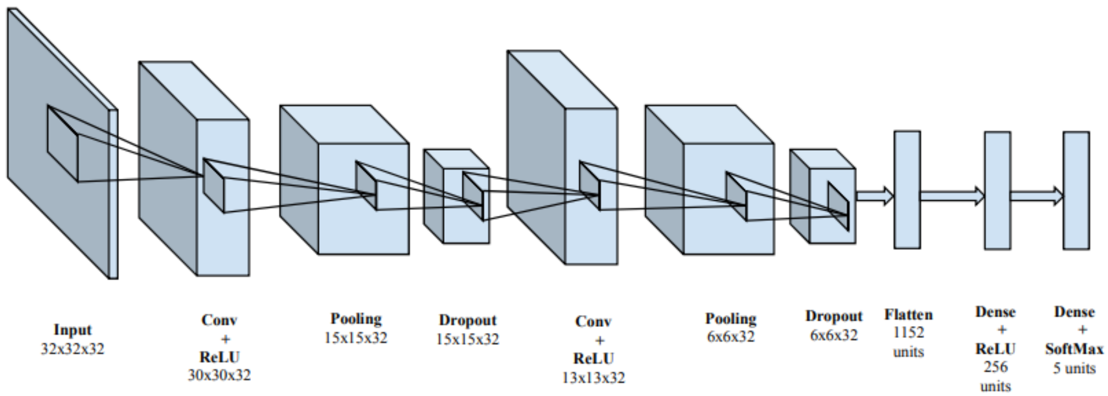


Fig 5.9 Architecture of the proposed facial emotion recognition model using DCNN

The architecture for the proposed facial emotion recognition model is depicted in Figure 4.5. The model uses two convolution layers with dropouts after each convolution layer. The input image is resized to 32 x 32 and is given to the first convolution layer. The output from the convolution layer, called feature map, is passed through an activation function. The activation function used here is ReLU (Rectified Linear Unit) that makes the negative values zero while the positive values remain the same. This feature map is given to the pooling layer of pool size 2 x 2 to reduce the size without losing any information. Dropout layer is used so as to reduce the overfitting.

This process again continues for the next convolution layer as well. Finally, a 2-dimensional array is created with some feature values. Flatten layer is used to convert these 2-dimensional arrays to a single dimensional vector so as to give it as the input of the neural network, represented by the dense layers. Here a two-layer neural network is used, one is input and the other is output. The output layer has 5 units, since 5 classes need to be classified. The activation function used in the output layer is softmax, which produces the probabilistic output for each class.

Model: "DCNN"

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 48, 48, 64)	1664
batchnorm_1 (BatchNormalization)	(None, 48, 48, 64)	256
conv2d_2 (Conv2D)	(None, 48, 48, 64)	102464
batchnorm_2 (BatchNormalization)	(None, 48, 48, 64)	256
maxpool2d_1 (MaxPooling2D)	(None, 24, 24, 64)	0
dropout_1 (Dropout)	(None, 24, 24, 64)	0
conv2d_3 (Conv2D)	(None, 24, 24, 128)	73856
batchnorm_3 (BatchNormalization)	(None, 24, 24, 128)	512
conv2d_4 (Conv2D)	(None, 24, 24, 128)	147584
batchnorm_4 (BatchNormalization)	(None, 24, 24, 128)	512
maxpool2d_2 (MaxPooling2D)	(None, 12, 12, 128)	0
dropout_2 (Dropout)	(None, 12, 12, 128)	0
conv2d_5 (Conv2D)	(None, 12, 12, 256)	295168
batchnorm_5 (BatchNormalization)	(None, 12, 12, 256)	1024
conv2d_6 (Conv2D)	(None, 12, 12, 256)	590080
batchnorm_6 (BatchNormalization)	(None, 12, 12, 256)	1024
maxpool2d_3 (MaxPooling2D)	(None, 6, 6, 256)	0
dropout_3 (Dropout)	(None, 6, 6, 256)	0
flatten (Flatten)	(None, 9216)	0
dense_1 (Dense)	(None, 128)	1179776
batchnorm_7 (BatchNormalization)	(None, 128)	512
dropout_4 (Dropout)	(None, 128)	0
out_layer (Dense)	(None, 3)	387

Fig 5.10 Model summary of the proposed facial emotion recognition model built using Keras

Figure 4.6 depicts a snapshot of the model summary of the proposed system which is built using the Keras DL Library.

Chapter 6

IMPLEMENTATION

6.1 Overview of system implementation

System implementation for face emotion recognition typically involves several steps, which can be summarized as follows:

1. Data collection: Collecting a dataset of face images with labeled emotions. This dataset is used for training and evaluation of the emotion recognition model. It may include images of faces with different expressions, poses, lighting conditions, etc or it may include the images in the form of pixel arrays.
2. Data pre-processing: Pre-processing the collected data, which may involve resizing the images to a consistent resolution, normalizing the pixel values, augmenting the data through techniques like rotation, flipping, or changing brightness/contrast, and splitting the dataset into training, validation, and test sets.
3. Model selection: Choosing an appropriate model architecture for emotion recognition. This could include traditional machine learning approaches such as Support Vector Machines (SVM), Decision Trees, or Random Forests, or deep learning approaches such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), or deeper learning models like Deep Convolutional Neural Networks (DCNN).
4. Model training: Training the selected model using the pre-processed training dataset. This involves feeding the images through the model, computing the loss or error, and updating the model's parameters (weights and biases) using an optimization algorithm (e.g., Exponential Linear Unit (elu) or Rectified Linear Unit (ReLU)) to minimize the loss.
5. Model evaluation: Evaluating the trained model's performance on the validation and test datasets using appropriate evaluation metrics such as accuracy and loss. Fine-tuning the model based on the evaluation results may be necessary to optimize the model's performance.
6. Model deployment: Deploying the trained model in a production environment for real-time inference. This may involve integrating the model into a web or mobile application, setting up an API for model inference, or deploying the model on edge devices for local inference.

7. Model monitoring and maintenance: Continuously monitoring the model's performance and making necessary updates or improvements as needed. This may involve updating the model with new data, optimizing the model for performance or memory usage, and addressing any issues that arise during deployment.
8. Testing and validation: Validating the deployed model's accuracy and performance on real-world data and making adjustments as needed. This may involve collecting additional feedback and validation data from users, monitoring the model's performance, and making necessary updates to ensure accurate and reliable emotion recognition in real-world scenarios.

Overall, system implementation for face emotion recognition requires a comprehensive approach involving data collection, pre-processing, model selection, training, evaluation, deployment, monitoring, and maintenance, while also considering security, privacy, and documentation aspects for a reliable and effective system.

6.2 Algorithm

1. Initialize the Deep Convolutional Neural Network (DCNN) model.

2. Add the first Conv2D layer with 64 filters, a kernel size of (5,5), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_1'.

3. Add BatchNormalization after the first Conv2D layer.

4. Add the second Conv2D layer with 64 filters, a kernel size of (5,5), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_2'.

5. Add BatchNormalization after the second Conv2D layer.

6. Add MaxPooling2D layer with a pool size of (2,2) and name it 'maxpool2d_1'.

7. Add Dropout layer with a dropout rate of 0.4 and name it 'dropout_1'.

8. Add the third Conv2D layer with 128 filters, a kernel size of (3,3), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_3'.

9. Add BatchNormalization after the third Conv2D layer.

10. Add the fourth Conv2D layer with 128 filters, a kernel size of (3,3), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_4'.

11. Add BatchNormalization after the fourth Conv2D layer.

12. Add MaxPooling2D layer with a pool size of (2,2) and name it 'maxpool2d_2'.

13. Add Dropout layer with a dropout rate of 0.4 and name it 'dropout_2'.

14. Add the fifth Conv2D layer with 256 filters, a kernel size of (3,3), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_5'.

15. Add BatchNormalization after the fifth Conv2D layer.

16. Add the sixth Conv2D layer with 256 filters, a kernel size of (3,3), 'elu' activation, 'same' padding, 'he_normal' kernel initializer, and name it 'conv2d_6'.

17. Add BatchNormalization after the sixth Conv2D layer.

18. Add MaxPooling2D layer with a pool size of (2,2) and name it 'maxpool2d_3'.

19. Add Dropout layer with a dropout rate of 0.5 and name it 'dropout_3'.

20. Add Flatten layer to flatten the tensor.

21. Add a fully connected Dense layer with 128 units, 'elu' activation, 'he_normal' kernel initializer, and name it 'dense_1'.

22. Add BatchNormalization after the Dense layer.

23. Add Dropout layer with a dropout rate of 0.6 and name it 'dropout_4'.

24. Add the output layer with the number of classes and 'softmax' activation, and name it 'out_layer'.

25. Compile the model with categorical cross-entropy loss, the specified optimizer (e.g., 'optim'), and accuracy metric.

26. Display the model summary.

27. Return the compiled DCNN model.

6.3 Pseudocode

```
# Importing Required Libraries
import os
import math
import numpy as np
import pandas as pd

import tensorflow as tf
import scikitplot
import seaborn as sns
from matplotlib import pyplot


from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report


from tensorflow.keras import optimizers
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Flatten, Dense, Conv2D, MaxPooling2D
from tensorflow.keras.layers import Dropout, BatchNormalization, LeakyReLU, Activation
from tensorflow.keras.callbacks import Callback, EarlyStopping, ReduceLROnPlateau
from tensorflow.keras.preprocessing.image import ImageDataGenerator

from keras.utils import np_utils
```

```
# Reading the Dataset
df = pd.read_csv('fer2013/fer2013.csv')

# Specifying each Emotion based on the Dataset
df.emotion.unique()
emotion_label_to_text = {0:'anger', 1:'disgust', 2:'fear', 3:'happiness', 4: 'sadness', 5: 'sur
prise', 6: 'neutral'}
```

```
# Splitting Current Dataset into Training and Validation(testing) Datasets
X_train, X_valid, y_train, y_valid = train_test_split(img_array, img_labels,
                                                    shuffle=True, stratify=img_labels,
                                                    test_size=0.1, random_state=42)
```

```
# Performing Image Data Normalization
X_train = X_train / 255.
X_valid = X_valid / 255.
```

```
def build_net(optim):
    """
    This is a Deep Convolutional Neural Network (DCNN). For generalization purpose       dropout
layer has been used in regular intervals.
    'ELU' has been used as the activation function as it avoids the dying problem in Relu and
also          performs          well          when          compared          to          LeakyRelu
atleast in this case. `he_normal` kernel initializer is used as it suits ELU. BatchNormalizatio
n is also used for better results.
    """
    net = Sequential(name='DCNN')

    net.add(
        Conv2D(
            filters=64,
            kernel_size=(5,5),
            input_shape=(img_width, img_height, img_depth),
            activation='elu',
            padding='same',
            kernel_initializer='he_normal',
            name='conv2d_1'
        )
    )
    net.add(BatchNormalization(name='batchnorm_1'))
    net.add(
        Conv2D(
            filters=64,
            kernel_size=(5,5),
            activation='elu',
            padding='same',
            kernel_initializer='he_normal',
            name='conv2d_2'
        )
    )
    net.add(BatchNormalization(name='batchnorm_2'))

    net.add(MaxPooling2D(pool_size=(2,2), name='maxpool2d_1'))
    net.add(Dropout(0.4, name='dropout_1'))

    net.add(
        Conv2D(
            filters=128,
            kernel_size=(3,3),
            activation='elu',
            padding='same',
            kernel_initializer='he_normal',
            name='conv2d_3'
        )
    )
```

```
net.add(BatchNormalization(name='batchnorm_3'))
net.add(
    Conv2D(
        filters=128,
        kernel_size=(3,3),
        activation='elu',
        padding='same',
        kernel_initializer='he_normal',
        name='conv2d_4'
    )
)
net.add(BatchNormalization(name='batchnorm_4'))

net.add(MaxPooling2D(pool_size=(2,2), name='maxpool2d_2'))
net.add(Dropout(0.4, name='dropout_2'))

net.add(
    Conv2D(
        filters=256,
        kernel_size=(3,3),
        activation='elu',
        padding='same',
        kernel_initializer='he_normal',
        name='conv2d_5'
    )
)
net.add(BatchNormalization(name='batchnorm_5'))
net.add(
    Conv2D(
        filters=256,
        kernel_size=(3,3),
        activation='elu',
        padding='same',
        kernel_initializer='he_normal',
        name='conv2d_6'
    )
)
net.add(BatchNormalization(name='batchnorm_6'))

net.add(MaxPooling2D(pool_size=(2,2), name='maxpool2d_3'))
net.add(Dropout(0.5, name='dropout_3'))

net.add(Flatten(name='flatten'))

net.add(
    Dense(
        128,
        activation='elu',
```

```
            kernel_initializer='he_normal',
            name='dense_1'
        )
    )
    net.add(BatchNormalization(name='batchnorm_7'))

    net.add(Dropout(0.6, name='dropout_4'))

    net.add(
        Dense(
            num_classes,
            activation='softmax',
            name='out_layer'
        )
    )

    net.compile(
        loss='categorical_crossentropy',
        optimizer=optim,
        metrics=['accuracy']
    )

    net.summary()

    return net
```

```
# Path to store the Model parameters
file_name = 'Model.h5'
checkpoint_path= os.path.join('Checkpoint2',file_name)
```

```
# Callback Functions to avoid overfitting
checkpoint = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_path,
                                                monitor='val_accuracy',
                                                verbose=1,
                                                save_freq='epoch',
                                                save_best_only=True,
                                                save_weights_only=False,mode='max')


early_stopping = EarlyStopping(
    monitor='val_accuracy',
    min_delta=0.00005,
    patience=11,
    verbose=1,
    restore_best_weights=True,
)
```

```
lr_scheduler = ReduceLROnPlateau(
    monitor='val_accuracy',
    factor=0.5,
    patience=7,
    min_lr=1e-7,
    verbose=1,
)

callbacks_list = [
    early_stopping,
    checkpoint,
    lr_scheduler,
]
```

```
# Generating augmented images for training DCNN model
train_datagen = ImageDataGenerator(
    rotation_range=15,
    width_shift_range=0.15,
    height_shift_range=0.15,
    shear_range=0.15,
    zoom_range=0.15,
    horizontal_flip=True,
)
train_datagen.fit(X_train)
```

```
# Training of DCNN Model
batch_size = 32 #batch size of 32 performs the best.
epochs = 100
optims = [
    optimizers.Nadam(learning_rate=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-
07, name='Nadam'),
    optimizers.Adam(0.001),
]

# Both Nadam and Adam has been used as optimizers for training of this model but Nadam has been
given more importance as it performs comparatively better and is also more popular.
model = build_net(optims[1])
history = model.fit_generator(
    train_datagen.flow(X_train, y_train, batch_size=batch_size),
    validation_data=(X_valid, y_valid),
    steps_per_epoch=len(X_train) / batch_size,
    epochs=epochs,
    callbacks=callbacks_list,
    use_multiprocessing=True
)
```

main.py

Real-Time Face Detection and Emotion Recognition Code – OpenCV

```
from tensorflow.keras import optimizers
from tensorflow.keras.models import load_model
from time import sleep
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.preprocessing import image
import cv2
import numpy as np

# Load Harr cascade for face detection
face_classifier = cv2.CascadeClassifier(r'D:\Final Year
Project\FER\haarcascade_frontalface_default.xml')

# Load pre-trained facial emotion recognition model
classifier = load_model(r'D:\Final Year Project\FER\Checkpoint\Model.h5',
compile=False)
optims = [
    optimizers.Nadam(learning_rate=0.001, beta_1=0.9, beta_2=0.999,
epsilon=1e-07, name='Nadam'),
    optimizers.Adam(0.001),
]
classifier.compile(
        loss='categorical_crossentropy',
        optimizer=optims,
        metrics=['accuracy']
    )
# Define list of emotions
emotion_labels = ['Angry','Disgust','Fear','Happy','Sad', 'Surprise',
'Neutral']

# Open Camera
cap = cv2.VideoCapture(0)


while True:

    # Capture frame from camera
    _, frame = cap.read()
    labels = [] # Defining labels

    # Convert frame to grayscale
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

    #detect faces in the frame
    faces = face_classifier.detectMultiScale(gray)

    # Iterate over detected faces
    for (x,y,w,h) in faces:

        # Extract Region of Interest(ROI)
        cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,255),2)
        roi_gray = gray[y:y+h,x:x+w]
        roi_gray = cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)



        if np.sum([roi_gray])!=0:
```

```
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi,axis=0)

            prediction = classifier.predict(roi)[0]
            label=emotion_labels[prediction.argmax()]
            label_position = (x,y)

cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),
2)
        else:
            cv2.putText(frame,'No
Faces',(30,80),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)

    # Display frame with emotions
    cv2.imshow('Emotion Detector',frame)

    # Exit loop on 'q' key press
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release camera and close all OpenCV windows
cap.release()
cv2.destroyAllWindows()
```

Chapter 7

TESTING

Testing can be stated as the process of verifying and validating whether a software or application is bug-free, meets the technical requirements as guided by its design and development, and meets the user requirements effectively and efficiently by handling all the exceptional and boundary cases.

The process of software testing aims not only at finding faults in the existing software but also at finding measures to improve the software in terms of efficiency, accuracy, and usability. It mainly aims at measuring the specification, functionality, and performance of a software program or application.

Test Case is a set of actions executed to verify a particular feature or functionality of your software application. A Test Case contains test steps, test data, precondition, postcondition developed for specific test scenario to verify any requirement. The test case includes specific variables or conditions, using which a testing engineer can compare expected and actual results to determine whether a software product is functioning as per the requirements.

7.1 Unit Testing

Unit Testing is a software testing technique by means of which individual units of software i.e., group of computer program modules, usage procedures, and operating procedures are tested to determine whether they are suitable for use or not. Unit Testing is defined as a type of software testing where individual components of software are tested. Unit Testing of the software product is carried out during the development of an application.

Table 7.1 Unit Test cases for FER DCNN model

Test Case No.	Description	Test Steps	Expected Results	Actual Results	Pass /Fail
1	Data Loading Test: Test if dataset is loaded properly and parsed into required format	Check if loaded dataset has same number of emotions	If there are 7 unique emotions specify each emotion using dictionary	7 Unique emotions are found and Each emotion value has been specified	Pass
2	Image Size: Test if the size of the image is 48*48 pixel size	Check if the image size is 48*48	Display 48 as the pixel length	48 is displayed as length of pixel verifying the image size as 48*48	Pass
3	Irrespective of the input image format store output in .h5 format	Check if .h5 file is created in the output folder after training is done	Output folder contains .h5 file	.h5 file is created in output folder	Pass
4	Depending on model different optimizers, callbacks are used for training	Use different optimized and callbacks based on model	Accuracy values improves based on the optimizers and callbacks used	Accuracy shows improvement based on the optimizers and callbacks that are used	Pass
5	Load the pre-trained model (.h5 file) into open cv to predict emotion	Check if the pre-trained model is determining the emotion	Based on the pre-trained model the emotion is detected	Emotions detected as per the loaded pre-trained model	Pass

7.2 System Testing

System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.

System Testing is carried out on the whole system in the context of either system requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system and also the expectations of the customer. It is performed to test the system beyond the bounds mentioned in the software requirements specification (SRS).

System Testing is basically performed by a testing team that is independent of the development team that helps to test the quality of the system impartial. It has both functional and non-functional testing.

System testing for facial emotion recognition using the FER2013 dataset involves assessing the performance and accuracy of a trained model on a separate set of images that were not used during training. The steps for system testing are as follows:

- Prepare the test set: Extract a representative set of images from the FER2013 dataset to be used for testing, which should reflect the real-world data the model will encounter during deployment.
- Pre-process the test images: Apply the same pre-processing steps (e.g., resizing, normalization, augmentation) to the test images as done during training to ensure consistency.
- Load the trained model: Load the facial emotion recognition model that was trained on the FER2013 training set.
- Predict emotions: Use the loaded model to predict emotions for the test images in the test set. Pass the pre-processed test images through the model and obtain the predicted emotion labels.

- Evaluate model performance: Compare the predicted emotion labels with the ground truth emotion labels in the test set to assess the performance of the model. Common evaluation metrics include accuracy, precision, recall, F1-score, and confusion matrix.
- Analyze results: Analyze the evaluation results to gain insights into the model's performance, identifying patterns, trends, and areas for improvement.
- Fine-tune the model: If the model's performance is unsatisfactory, consider fine-tuning by adjusting hyperparameters, retraining with additional data, or optimizing pre-processing steps.
- Repeat testing and refinement: Iterate the testing and refinement process until the model achieves satisfactory performance on the test set.
- Finalize model: Once the model achieves satisfactory performance, finalize it and prepare for deployment in a production environment.

System testing is crucial in the development and evaluation of facial emotion recognition models, ensuring accurate and reliable performance on real-world data. Thorough testing with various evaluation metrics and fine-tuning as needed is important to achieve desired performance levels.

7.2 Validation Testing

Validation is the process of checking whether the software product is up to the mark or in other words product has high level requirements. It is the process of checking the validation of product i.e., it checks what we are developing is the right product. It is validation of actual and expected product. Validation is the Dynamic Testing.

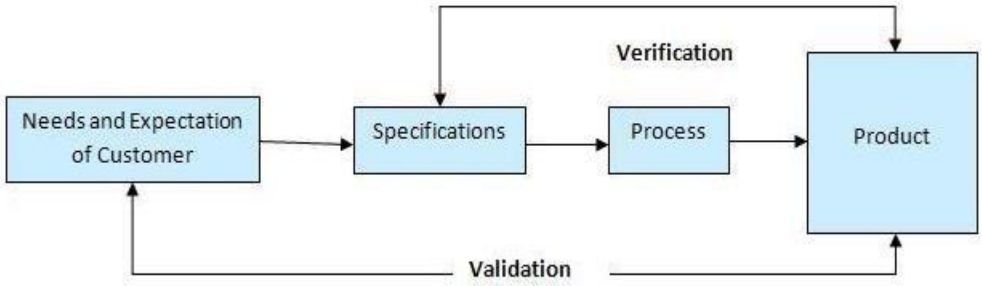


Fig 7.1 Software verification and validation

Validation testing is crucial to ensure that the trained model generalizes well to unseen data and performs accurately on real-world data. It helps in identifying any issues or areas of improvement and fine-tuning the model for better performance.

The need of the project was to implement a DCNN based model for real-time Face emotion recognition, which we refer to as FER. We also wanted to formulate a multi-modal objective function to train the model by evaluating the perceptual quality of an image based on its global content, color, local texture, and style information. We also wanted to carry out qualitative and quantitative performance evaluations compared to state-of-the-art models.

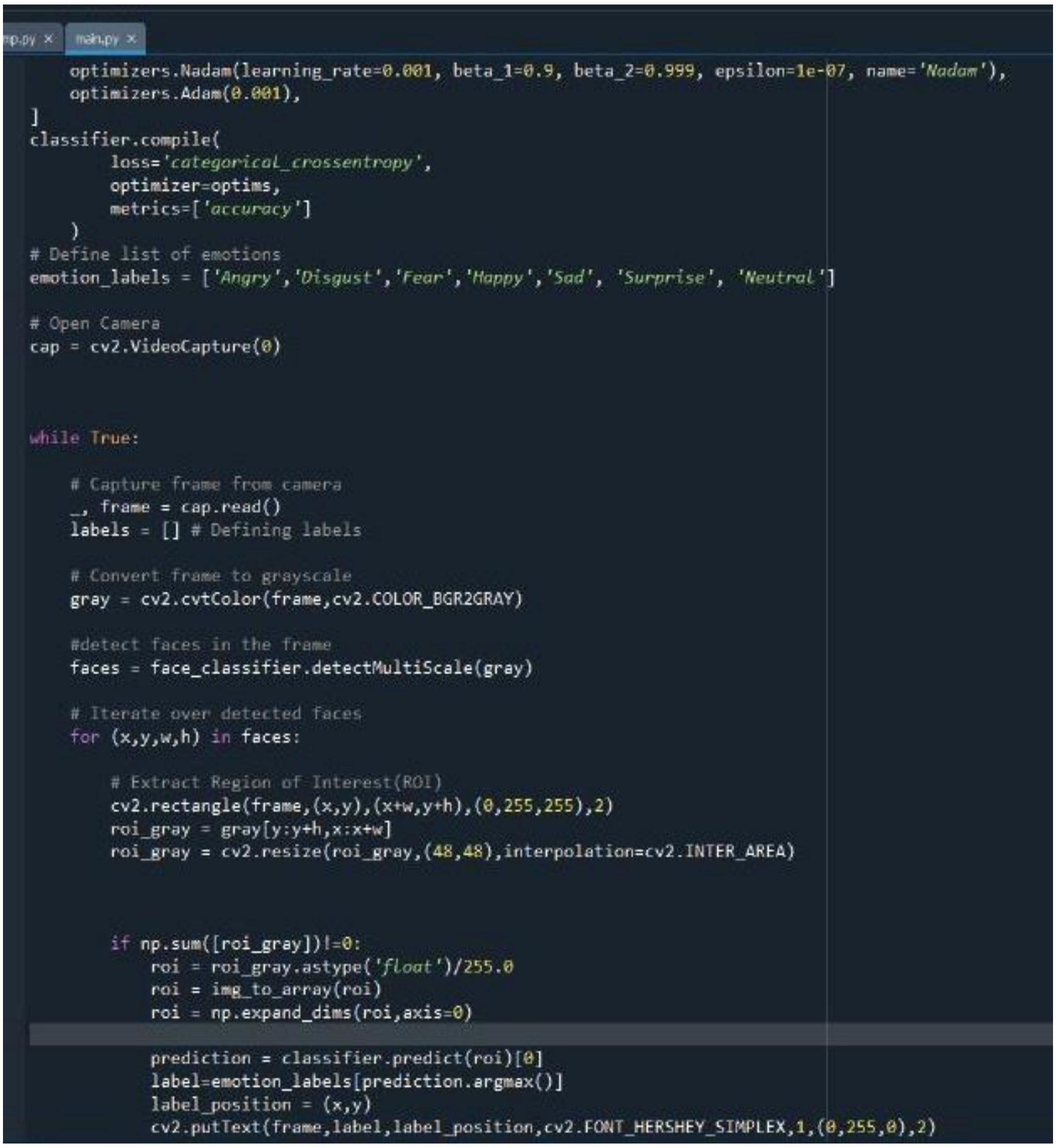
Based on the formulation and specifications mentioned, the model was implemented with the help of libraries of TensorFlow, Keras and OpenCV. FER2013 dataset consisting of large set of images was utilized for building the model.

The proposed model was compared with the existing models and the required results were achieved. FER predicts the emotion detected accurately based on the proposed model and achieved required results.

Chapter 8

DISCUSSION OF RESULTS

This section discusses the results from the trained model.



```
    optimizers.Nadam(learning_rate=0.001, beta_1=0.9, beta_2=0.999, epsilon=1e-07, name='Nadam'),
    optimizers.Adam(0.001),
]
classifier.compile(
        loss='categorical_crossentropy',
        optimizer=optims,
        metrics=['accuracy']
    )
# Define list of emotions
emotion_labels = ['Angry','Disgust','Fear','Happy','Sad', 'Surprise', 'Neutral']

# Open Camera
cap = cv2.VideoCapture(0)



while True:

    # Capture frame from camera
    _, frame = cap.read()
    labels = [] # Defining labels

    # Convert frame to grayscale
    gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)

    #detect faces in the frame
    faces = face_classifier.detectMultiScale(gray)

    # Iterate over detected faces
    for (x,y,w,h) in faces:

        # Extract Region of Interest(ROI)
        cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,255),2)
        roi_gray = gray[y:y+h,x:x+w]
        roi_gray = cv2.resize(roi_gray,(48,48),interpolation=cv2.INTER_AREA)



        if np.sum([roi_gray])!=0:
            roi = roi_gray.astype('float')/255.0
            roi = img_to_array(roi)
            roi = np.expand_dims(roi,axis=0)

            prediction = classifier.predict(roi)[0]
            label=emotion_labels[prediction.argmax()]
            label_position = (x,y)
            cv2.putText(frame,label,label_position,cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)
```

Fig 8.1 Opencv code for real time emotion recognition

Fig 8.1 represents the opencv code for real time emotion recognition.

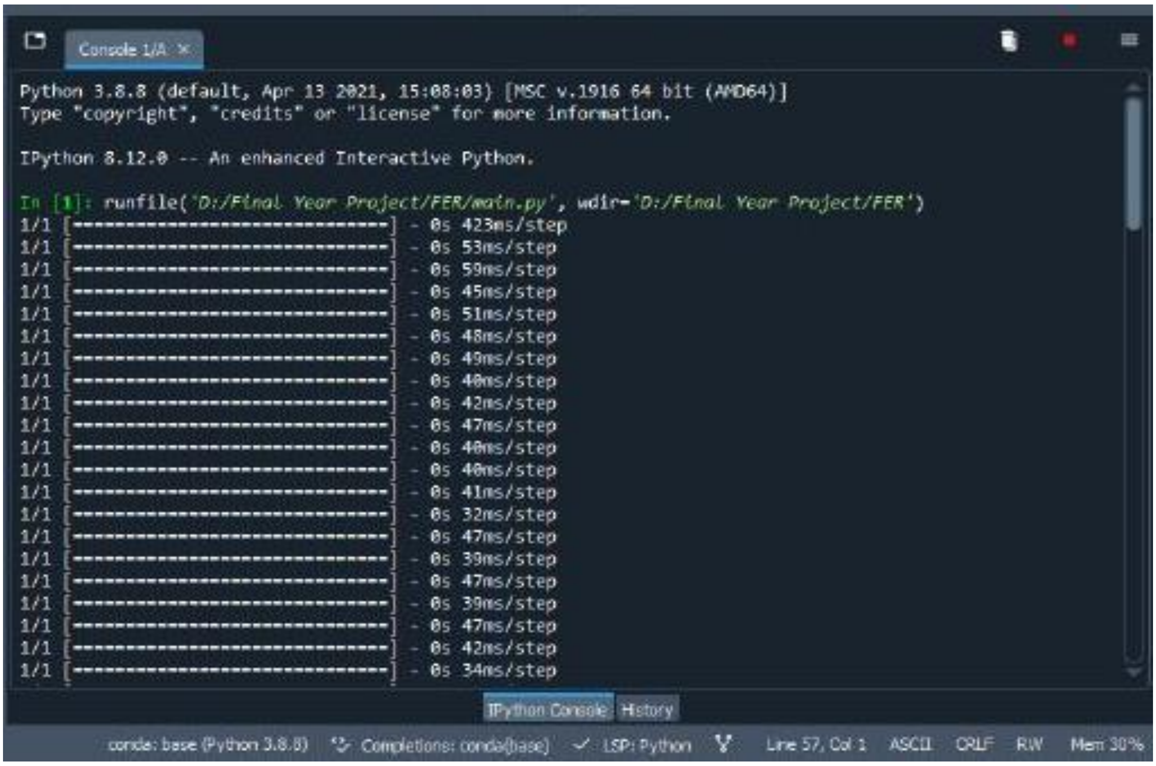


Fig 8.2 Implementation of opencv code

Fig 8.2 represents the implementation of the above opencv code.

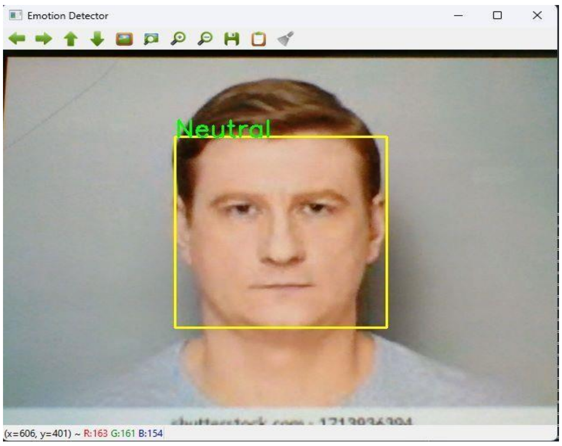


Fig 8.3 Result of the implementation of opencv code

Fig 8.3 represents the result of the implementation of the above opencv code. This turns on the camera and real time facial emotion recognition is carried out.

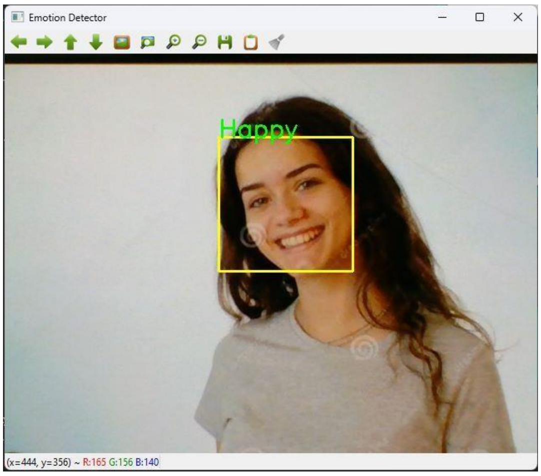


Fig 8.4 Example of happiness emotion being recognized

Fig 8.4 is an example when the recognized emotion is happiness.

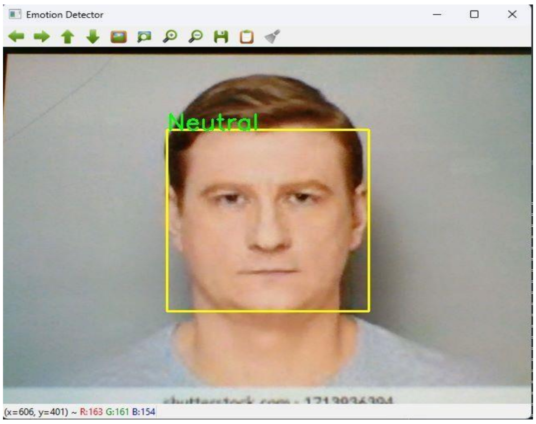


Fig 8.5 Example of neutral emotion being recognized

Fig 8.5 is an example when the recognized emotion is neutral.

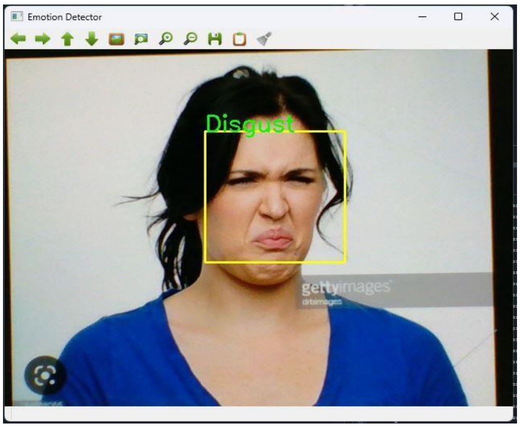


Fig 8.6 Example of disgust emotion being recognized

Fig 8.6 is an example whenthe recognized emotion is disgust. Disgust is comparatively a difficult emotion to be recognized.

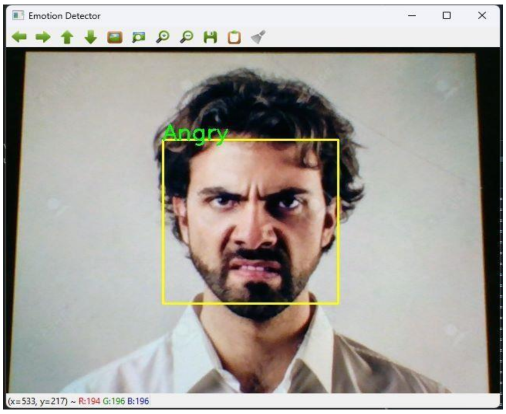


Fig 8.7 Example of anger emotion being recognized

Fig 8.7 is an example of when the recognized emotion is anger.

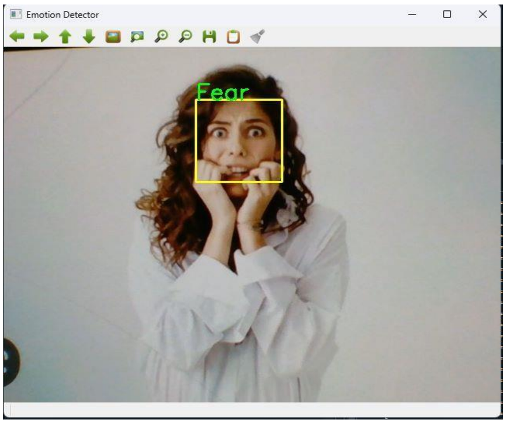


Fig 8.8 Example of fear emotion being recognized

Fig 8.8 is an example when the recognized emotion is fear. The result of neutral and anger emotions keeps toggling as these two emotions vary from person to person.

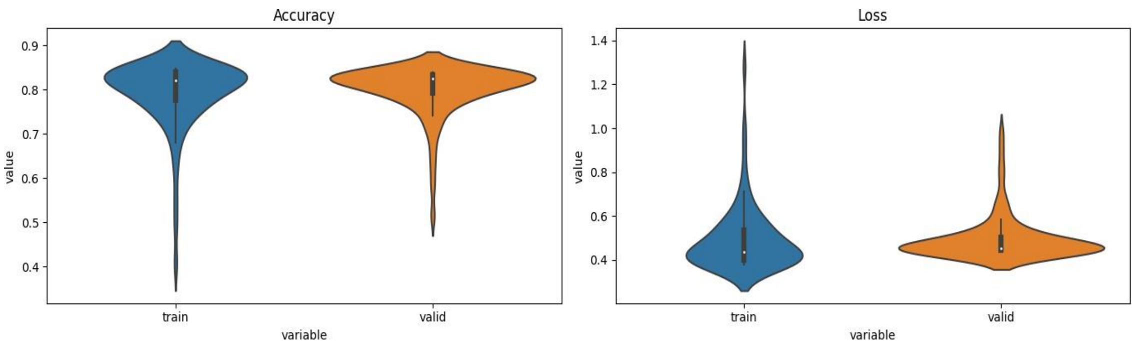


Fig 8.9 Violin Plot of accuracy and loss functions

Fig 8.9 is a representation of the plot of accuracy and loss functions of both the trainingand testing data.

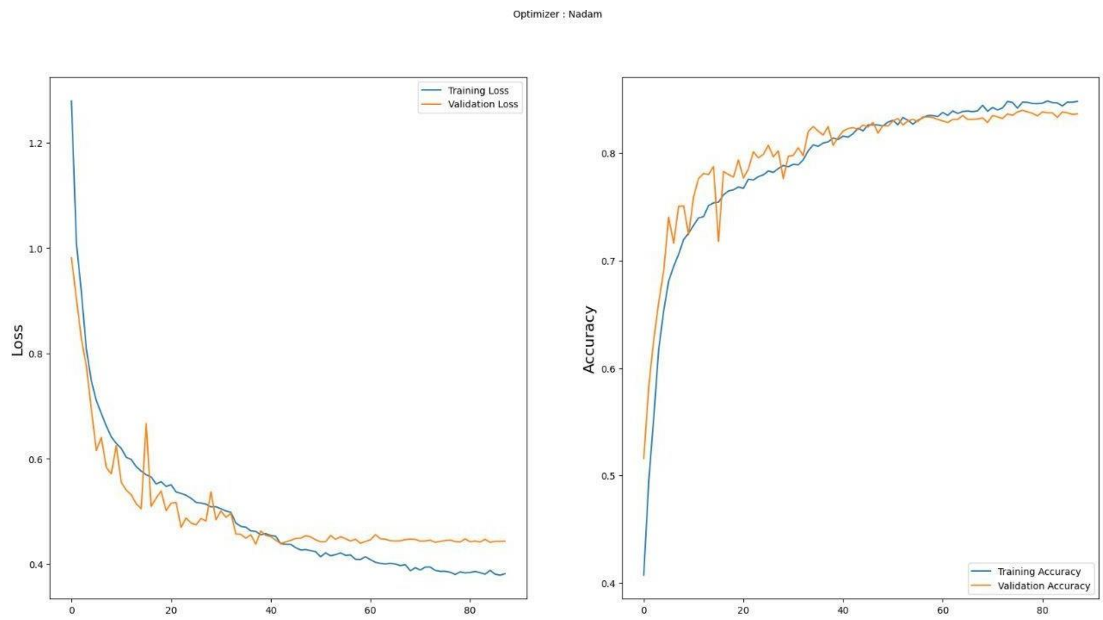


Fig 8.10 Line graph representing the accuracy and lossfunctions

Fig 8.10 represents the same data in the form of a line graph.

8.1 Summary

The deep convolutional neural network (CNN) model on the Fer2013 dataset with an 84% accuracy is a significant achievement in the field of image recognition and emotiondetection. The Fer2013 dataset is a challenging dataset consisting of 48x48 pixel grayscale images of faces categorized into seven emotions: anger, disgust, fear, happiness, sadness, surprise, and neutral. The dataset contains 35,887 training images, 3,589 validation images, and 3,589 test images, making it a suitable dataset for training deep learning models.

The DCNN model was trained using a combination of convolutional, pooling, and dense layers. The convolutional layers were used to extract features from the input images, and the pooling layers were used to reduce the spatial dimensions of the feature maps. The dense layers were used to classify the emotions in the images. The model was trained using the Adam optimizer with a learning rate of 0.001 and a batch size of 64. The training was stopped after 100 epochs to prevent overfitting.

The 84% accuracy achieved by the DCNN model on the Fer2013 dataset is impressive, considering the complexity of the dataset and the challenges associated with image recognition and emotion detection. The model's performance was evaluated using various metrics such as accuracy, precision, recall, and F1-score. The confusion matrix was also used to visualize the model's performance and identify areas for improvement.

One of the key strengths of the Deep Convolutional Neural Network (DCNN) model is its ability to learn and extract features from images automatically. This makes the model suitable for various applications such as facial recognition, emotion detection, and object recognition. The model's ability to learn and generalize from the training data is also critical, as it enables the model to make accurate predictions on new and unseen data.

Another strength of the DCNN model is its scalability. As the size of the dataset increases, the DCNN model can be scaled up by adding more convolutional and dense layers. This allows the model to learn and extract more complex features from the input images, leading to improved accuracy.

Despite the impressive performance of the DCNN model, there are still some areas for improvement. One of the key limitations of the model is its interpretability. The model's ability to extract features automatically makes it difficult to understand the specific features that the model is using to make predictions. This can make it challenging to debug the model and identify areas for improvement.

Another limitation of the model is its sensitivity to the quality and diversity of the training data. If the training data is biased or lacks diversity, the model may not be able to generalize well to new and unseen data. This highlights the importance of data preprocessing and augmentation techniques to improve the quality and diversity of the training data.

In conclusion, the DCNN model on the Fer2013 dataset with an 84% accuracy is a significant achievement in the field of image recognition and emotion detection. The model's ability to learn and extract features automatically makes it suitable for various applications, and its scalability allows it to handle larger and more complex datasets. However, the model's interpretability and sensitivity to the quality and diversity of the training data are still areas for improvement. Future research should focus on developing techniques to improve the interpretability of deep learning models and address the challenges associated with bias and lack of diversity in training data.

Chapter 9

CONCLUSION AND FUTURE ENHANCEMENTS

9.1 Conclusion

Autism spectrum disorder (ASD) is a neurodevelopmental disorder that affects communication and social interaction. It is characterized by difficulty with social communication, such as difficulty with eye contact, facial expression, and body language, and difficulty with social interaction, such as difficulty with initiating and maintaining conversations or engaging in play with others.

While there is no current cure for ASD, Treatment for ASD may include a combination of speech and language therapy, occupational therapy, and behavioural therapy.

By utilizing the improvement in technologies such as computer vision, Artificial Neural Networks, etc, this project aims to improve the methods currently used in behavioural therapy for children diagnosed with ASD.

The proposed system uses a Deep Convolutional Neural Network (DCNN) architecture to create a Neural Network capable of detecting and classifying the emotion of the subject it is evaluating from live video/images of the subject. This model is then packaged in a user-friendly web application that references the model with images of people around the child. By using the proposed system to analyze facial expressions, body language, and other nonverbal cues, it can provide real-time feedback and guidance to children with ASD, helping them to identify and understand the emotions of others. The system can also be trained to accurately recognize a widerange of emotions and can be used in a variety of settings, including classrooms, therapy sessions,and home environments.

In the context of detecting emotions from facial expressions, DCNN networks can be trained to recognize patterns in sequences of facial images and classify them into different emotion categories. This can be done by feeding the DCNN network a dataset of images labelledwith the corresponding emotion and training the network to predict the emotion of an image based on the patterns it learns from the dataset.

One key advantage of using DCNNs for emotion detection is they are very effective at learning complex features from raw data, such as facial expressions, which is critical for emotion detection. This is because DCNNs can automatically extract and encode the most relevant and discriminative features of facial expressions, making them more accurate than traditional machine learning models that rely on handcrafted features.

In conclusion, the proposed system shows a lot of promise in improving the means by which Children diagnosed with ASD learn and train to better understand and participate in socialinteractions thereby helping them live more productive and fulfilling lives.

9.2 Future Enhancements

9.2.1 Optimizing the application to run on lower-powered systems to increase accessibility.

9.2.2 Designing a system to monitor the environment over the long term to generate larger personal datasets aiding in generating a more accurate model which can account for growthand mental development.

9.2.3 Establish a larger number of classifications for emotions other than the standard emotions in order to increase the sensitivity of the system to minute changes in detected facial expressions.

9.2.4 Design and develop a fully end-to-end software and hardware framework that can be easilyused by patients.

9.2.5 Explore the use of transfer learning to enhance the base - model and retrain for different use cases of classification that can be used for training children with ASD.

REFERENCES

[1] Hui Ding, Shaohua Kevin Zhou and Rama Chellappa, "FaceNet2ExpNet: Regularizing a Deep Face Recognition Net for Expression Recognition", Sept 2016, arXiv.1609.06591

[2] S. Gupta, P. Kumar, and R. K. Tekchandani, "Facial emotion recognition based real-time learner engagement detection system in online learning context using deep learning models," Multimedia Tools and Applications. Springer Science and Business Media LLC, Sep. 09, 2022. doi: 10.1007/s11042-022-13558-9.

[3] H. Li, N. Wang, X. Ding, X. Yang, and X. Gao, "Adaptively Learning Facial Expression Representation via C-F Labels and Distillation," IEEE Transactions on Image Processing, vol. 30. Institute of Electrical and Electronics Engineers (IEEE), pp. 2016–2028, 2021. doi: 10.1109/tip.2021.3049955.

[4] Jiawei Shi, Songhao Zhu and Dongsheng Wang, "Learning to Amend Facial Expression Representation via De-albino and Affinity", Oct 2021, arXiv.2103.10189

[5] R. Pavez, J. Diaz, J. Arango-Lopez, D. Ahumada, C. Mendez-Sandoval, and F. Moreira, "Emo-mirror: a proposal to support emotion recognition in children with autism spectrum disorders," Neural Computing and Applications. Springer Science and Business Media LLC, Oct. 08, 2021. doi: 10.1007/s00521-021-06592-5.

[6] J. M. Garcia-Garcia, V. M. R. Penichet, M. D. Lozano, and A. Fernando, "Using emotion recognition technologies to teach children with autism spectrum disorder how to identify and express emotions," Universal Access in the Information Society, vol. 21, no. 4. Springer Science and Business Media LLC, pp. 809–825, Jun. 19, 2021. doi: 10.1007/s10209-021-00818-y.

[7] Mouath Aouayeb, Wassim Hamidouche, Catherine Soladie, Kidiyo Kpalma and Renaud Seguier, "Learning vision transformer with squeeze and excitation for facial expression recognition", Jul 2021, arXiv:2107.03107

[8] Yuhang Zhang, Chengrui Wang, Xu Ling and Weihong Deng, "Learn from all: Erasing attention consistency for Noisy Label Facial Expression Recognition", Sep 2022, arXiv:2207.10299

[9] T. Kumar Arora et al., “Optimal Facial Feature Based Emotional Recognition Using Deep Learning Algorithm,” Computational Intelligence and Neuroscience, vol. 2022. Hindawi Limited, pp. 1–10, Sep. 20, 2022. doi: 10.1155/2022/8379202.

[10] Radhey Shyam, “Facial Emotion Detection And Recognition”, May 2022, International Journal of Engineering Applied Sciences and Technology, 2022, Vol. 7, Issue 1, ISSN No. 2455-2143, Pages 176-179

[11] M. A. H. Akhand, S. Roy, N. Siddique, M. A. S. Kamal, and T. Shimamura, “Facial Emotion Recognition Using Transfer Learning in the Deep CNN,” Electronics, vol. 10, no. 9. MDPI AG, p. 1036, Apr. 27, 2021. doi: 10.3390/electronics10091036.

[12] S. Begaj, A. O. Topal and M. Ali, "Emotion Recognition Based on Facial Expressions Using Convolutional Neural Network (CNN)," 2020 International Conference on Computing, Networking, Telecommunications & Engineering Sciences Applications (CoNTESA), 2020, pp. 58-63, doi: 10.1109/CoNTESA50436.2020.9302866.

[13] Zhu D, Fu Y, Zhao X, Wang X, Yi H, “Facial Emotion Recognition Using a Novel Fusion of Convolutional Neural Network and Local Binary Pattern in Crime Investigation” Comput Intell Neurosci. 2022 Sep 22; 2022:2249417. doi: 10.1155/2022/2249417. PMID: 36188698; PMCID: PMC9522492.

[14] Pise AA, Alqahtani MA, Verma P, K P, Karras DA, S P, Halifa A, “Methods for Facial Expression Recognition with Applications in Challenging Situations”, Comput Intell Neurosci. 2022 May 25; 2022:9261438. doi: 10.1155/2022/9261438. PMID: 35665283; PMCID: PMC9159845.

[15] KH Teoh, RC Ismail1, SZM Naziri, R Hussin, MNM Isa and MSSM Basir, “Face Recognition and Identification using Deep Learning Approach”, 2021 J. Phys.: Conf. Ser. 1755 012006.

[16] G. Wan et al., “FECTS: A Facial Emotion Cognition and Training System for Chinese Children with Autism Spectrum Disorder,” Computational Intelligence and Neuroscience, vol. 2022. Hindawi Limited, pp. 1–21, Apr. 27, 2022. doi: 10.1155/2022/9213526.

[17] K. Premkumar et al., “Facial Emotion Recognition for Autism Children,” International Journal of Innovative Technology and Exploring Engineering, vol. 9, no. 7. Blue Eyes Intelligence Engineering and Sciences Engineering and Sciences Publication - BEIESP, pp. 1274–1278, May 30, 2020. doi: 10.35940/ijitee.f3772.059720.

[18] P. Rani, "Emotion Detection of Autistic Children Using Image Processing," 2019 Fifth International Conference on Image Information Processing (ICIIP), 2019, pp. 532-535, doi: 10.1109/ICIIP47207.2019.8985706.

[19] D. Acharya, Z. Huang, D. P. Paudel and L. Van Gool, "Covariance Pooling for Facial Expression Recognition," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 480-4807, doi: 10.1109/CVPRW.2018.00077.

[20] Hengshun Zhou, Debin Meng, Yuanyuan Zhang, Xiaojiang Peng, Jun Du, Kai Wang, Yu., “Exploring Emotion Features and Fusion Strategies for Audio-Video Emotion Recognition,” arXiv, 2020, doi: 10.48550/ARXIV.2012.13912.

[21] S. Begaj, A. O. Topal and M. Ali, "Emotion Recognition Based on Facial Expressions Using Convolutional Neural Network (CNN)," 2020 International Conference on Computing, Networking, Telecommunications & Engineering Sciences Applications (CoNTESA), 2020, pp. 58-63, doi: 10.1109/CoNTESA50436.2020.9302866.

[22] N. Mehendale, “Facial emotion recognition using convolutional neural networks (FERC),” SN Applied Sciences, vol. 2, no. 3. Springer Science and Business Media LLC, Feb. 18, 2020. doi: 10.1007/s42452-020-2234-1.