MINOR PROJECT REPORT
ON

THESPIS.AI

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

BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY

Submitted By

Archi Agrawal
03815603119

Rishabh Sharma
04015603119

Taiyaba Zaheer
04615603119

Under the guidance of
Ms. Charul Dewan, Departmental In-charge, IT department

Department of Information Technology
Dr. Akhilesh Das Gupta Institute of Technology & Management
Guru Gobind Singh Indraprastha University
Dwarka, Delhi-110078
JAN-2023

CERTIFICATE

We hereby certify that the work that is being presented in the project report entitled **Thespis.AI** to the partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Information Technology from Dr. Akhilesh Das Gupta Institute of Technology & Management**, New Delhi. This is an authentic record of our own work carried out during a period from September, 2022 to January, 2023 under the guidance of **Ms. Charul Dewan, Departmental In-charge, IT department.**

The matter presented in this project has not been submitted by us for the award of any other degree elsewhere.

Archi Agrawal	**Rishabh Sharma**
(03815603119)	***(04015603119)***

Taiyaba Zaheer
(04615603119)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge. They are permitted to appear in the Major Project External Examination.

(Charul Dewan)	**(Charul Dewan)**
Assistant Professor	**Departmental In-charge, IT**

The B. Tech Major Project Viva-Voce Examination of **Archi Agrawal (Enrollment No: 03815603119), Rishabh Sharma (Enrollment No: 04015603119) and Taiyaba Zaheer (Enrollment No: 04615603119),** has been held on **…………………….**

Prof. (Dr.) Anil Kumar, IT Deptt.	**(Signature of External Examiner)**
(Project Coordinator, IT Deptt.)	

ACKNOWLEDGEMENT

We would like to acknowledge the contributions of the following persons without whose help and guidance this report would not have been completed.

We acknowledge the counsel and support of our project guide **Ms. Charul Dewan, Assistant Professor and Departmental In-charge, IT department,** with respect and gratitude, whose expertise, guidance, support, encouragement, and enthusiasm has made this report possible. Their feedback vastly improved the quality of this report and provided an enthralling experience. We are indeed proud and fortunate to be supervised by her. We are thankful for her constant encouragement, valuable suggestions and moral support and blessings.

We are immensely thankful to our esteemed, **Prof. (Dr.) Sanjay Kumar, Director, Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi** for his never-ending motivation and support.

We shall ever remain indebted to **Prof. (Dr.) Anil Kumar, Project Coordinator, IT department** and faculty and staff members of Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi.

Finally, yet importantly, we would like to express our heartfelt thanks to our beloved parents for their blessings, our friends/classmates for their help and wishes for the successful completion of this project.

Archi Agrawal
(03815603119)

Rishabh Sharma
(04015603119)

Taiyaba Zaheer
(04615603119)

ABSTRACT

Expressing through human facial emotions is still the most fundamental way of communication. There has been an active research over the last few decades in this field but it is still challenging due to the high intra-class variation, cross-cultural variability, nonverbal cues, high intra-class variation and variations in head pose, illumination, and occlusions. Thespis.AI is a step in the direction of more accurately and efficiently recognizing human facial emotions of a diverse population. The aim of Thespis.AI is to identify human facial emotions in real time. In this project, we have identified faces using haarcascade algorithm for face detection and have implemented an ensemble of various deep convolutional neural networks models and pre-trained transfer learning models with a voting technique for facial expression recognition (FER) training to achieve greater accuracy. To make better predictions, the hyperparameter tuning of the proposed models were performed and approaches like data augmentation, class weighting, adding auxiliary data, and ensembling were used. This model was tested on three datasets including the challenging FER2013, JAFFE and CK+, achieving a state-of-the-art 76.2% accuracy on the publicly available FER2013 test set, outperforming all existing publications. Additionally, we showcase a flask-based web application which runs our FER models on a public URL on device in real time and recognizes the emotions displayed by the human in front of the webcam.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

1.1 INTRODUCTION

Emotions are an inevitable portion of any inter-personal communication. They can be expressed in many different forms which may or may not be observed with the naked eye. Since AI systems are often deployed in areas inhabited or frequented by humans, it is of utmost importance that a machine can recognize and interact with them. Interaction with humans is not limited to conversation but relies heavily on facial expressions and body language. This makes it necessary for machines to be able to accurately identify human emotions. Therefore, with the right tools, any indications preceding or following them can be subject to detection and recognition. There has been interest in human emotion recognition in various fields including, but not limited to, human-computer interaction, animation, medicine, and security. Emotion recognition can be performed using different features, such as face, speech, EEG, and even text. Among these features, facial expressions are one of the most popular, due to a number of reasons; they are visible, they contain many useful features for emotion recognition, and it is easier to collect a large dataset of faces (than other means for human recognition).

The aim of Thespis.AI is to identify human facial emotions in real time. The web application uses computer vision and deep learning to accurately identify human facial emotions. The application uses a web camera to capture live video and displays the percentage of each emotion detected. The percentages are displayed in the form of a bar graph. The application is capable of classifying seven different facial emotions.

Existing facial emotion recognition systems are sometimes challenged with issues pertaining to accuracy and efficiency. Especially when challenged with people of diverse ethnic backgrounds and gender identities, systems tend to falter. Thespis.AI is a step in the direction of more accurately and efficiently recognizing human facial emotions of a diverse population.

While human-machine interaction is an ever-expanding field, the application does find some direct utilities in today’s world.

1.2 BASIC TERMS OF PROJECT

1.2.1 WHAT IS ARTIFICIAL INTELLIGENCE?

While a number of definitions of artificial intelligence (AI) have surfaced over the last few decades, John McCarthy offers the following definition in his 2004 paper, "It is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable."

1.2.2 WHAT IS MACHINE LEARNING?

According to IBM, machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy. Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, and to uncover key insights in data mining projects. These insights subsequently drive decision making within applications and businesses, ideally impacting key growth metrics.

1.2.3 WHAT IS DEEP LEARNING?

Deep learning is a subset of machine learning, which is essentially a neural network with three or more layers. These neural networks attempt to simulate the behavior of the human brain—albeit far from matching its ability—allowing it to “learn” from large amount of data. While a neural network with a single layer can still make approximate predictions, additional hidden layers can help to optimize and refine for accuracy.

Deep learning drives many artificial intelligence (AI) applications and services that improve automation, performing analytical and physical tasks without human intervention. Deep learning technology lies behind everyday products and services (such as digital assistants, voice-enabled TV remotes, and credit card fraud detection) as well as emerging technologies.

1.2.4 WHAT IS COMPUTER VISION?

Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs — and take actions or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand.

1.2.5 TYPES OF WEB DEVELOPMENT

Web development refers to the creating, building, and maintaining of websites. It includes aspects such as web design, web publishing, web programming, and database management. It is the creation of an application that works over the internet i.e. websites.

Web Development can be classified into two ways:

- **Frontend Development**: The part of a website that the user interacts directly is termed as front end. It is also referred to as the 'client side' of the application.
- **Backend Development**: Backend is the server side of a website. It is the part of the website that users cannot see and interact. It is the portion of software that does not come in direct contact with the users. It is used to store and arrange data.

1.3 LITERATURE OVERVIEW

Facial Emotion Recognition (FER) is a research area that aims to develop algorithms that can automatically detect and interpret human emotions from facial expressions. The field has seen a significant amount of research in recent years, with a growing number of studies focusing on developing deep learning-based approaches for FER. These approaches have demonstrated improved performance compared to traditional methods, particularly in terms of handling variations in facial expressions and facial pose.

In 1971, Ekman and Friesen proposed seven basic emotions for all cultures, including angry, disgust, fear, happy, neutral, sad, and surprise. By analyzing facial component expression/appearance, emotion can be recognized. One of the fundamental methods used to classify facial expression is based on Facial Action Coding System (FACS). FACS includes visual information of facial muscles known as Action Units (AUs). It is straightforward to notice that most of selected facial regions are located around eyes, nose, or mouth. Generally, the muscle around these regions tends to activate to describe our feelings. As a result, researchers have been focusing on detecting and analyzing them for emotion recognition.

One of the most popular deep learning-based approaches for FER is the use of convolutional neural networks (CNNs). CNNs have been used to extract features from facial images, which are then used for emotion classification. Other deep learning-based approaches that have been used for FER include recurrent neural networks (RNNs) and long short-term memory (LSTM) networks.

The state-of-the-art algorithm on CK and CK+ datasets is a CNN with four Inception modules. To overcome the limit of the above datasets in terms of head poses, or capturing environment, other datasets were provided with spontaneous condition such as DISFA or AM-FED. Since the number of data in these datasets is small, traditional learning methods such as SVM and handcrafted features can achieve comparable results.

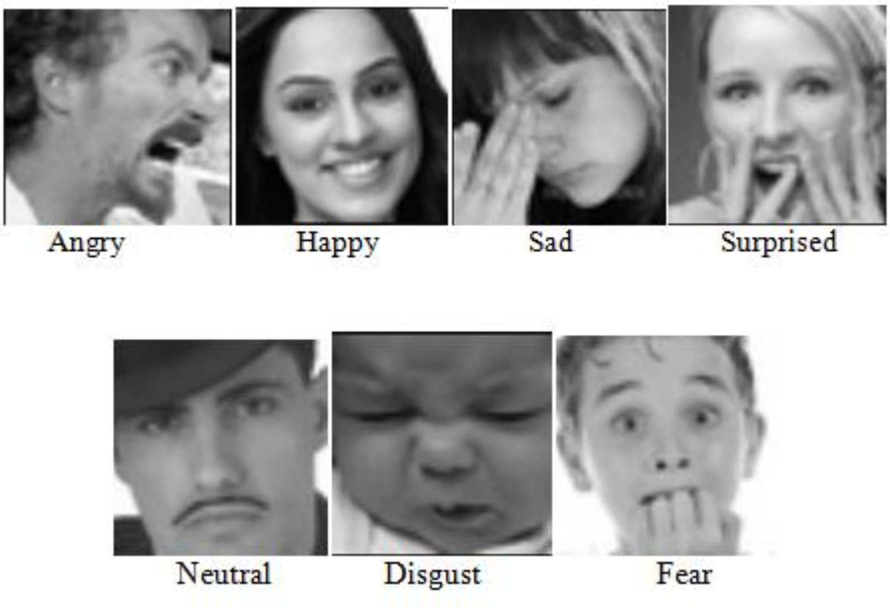
Xie(2018) developed a model based on Deep Comprehensive Multi patches Aggregation CNNs. In this work, two branches of CNNs were used. One branch of CNN was used for extracting the local features from patches and the other branch of CNNs were used for obtaining the holistic features from the entire face sample and these features were combined to create a feature vector and given to the classifier for expression classification. Experimentation was performed on CK+, JAFFE datasets and attained 93.46, 94.75% accuracy respectively. Mayya(2016) developed a new method for recognizing emotions using DCNNs. In their work, the first face was detected from dataset images and given those frontal face images to CNN for extracting features. SVM with a grid search was used for classification. Proposed models were evaluated on CK+, JAFEE and achieved 97, 98.12% accuracy respectively.

In the FER2013 challenge of ICML 2013, Tang introduced a CNN jointly learned with linear support vector machine (SVM) for facial expression recognition. With a simple CNN and an SVM instead of softmax classifier, this model outperformed others and won the first place in the challenge. Inspired by the success of GoogLeNet, Mollahosseini et al. have proposed an architecture containing four Inception modules. However, their research cannot lead to a better performance on FER2013 dataset. In 2016, Zhou et al. proposed multi-scale CNNs. This model consists of three other networks with different input sizes. In addition, they used late fusion to get final classification results. By combining multiple CNNs and modifying loss function, Yu et al. obtained a higher accuracy compared to previous approaches. Similarly, Kim et al. have introduced a multiple CNNs for facial expression recognition in the wild. Another multi-scale CNN was proposed by Wang et al. In that work, the authors used the entire feature maps in the network for classification. However, using all generated features without selection may reduce the overall performance due to trivial information in shallow layers of the network.

A CNN feature-based FER was developed by Gonzalez-Lozoya in 2020 which achieved an accuracy of 70.7% on the FER2013 dataset. Facial features were extracted using CNN. Model generalization was improved by mixing different dataset images. Another study from 2020, proposed a new method for facial expression recognition based on the transfer learning technique and achieved an accuracy of 72.5% on the FER2013 dataset. An improvement on facial expression recognition was achieved by combining SIFT and CNN features CNN features are merged with the SIFT features to increase the FER accuracy. This work was tested on FER2013 and CK+ datasets and attained 73.4% and 99.1% accuracy respectively. Another study from 2021, used a deep learning model with a new method called "Multimodal Fusion" which combines multiple modalities, such as facial expressions, speech, and physiological signals. The study achieved an accuracy of 74.1% on the FER2013 dataset.

It's important to note that FER2013 dataset has limited number of images per class and it is created on a controlled environment, so, the results on FER2013 dataset may not generalize well to real-world scenarios. Therefore, it's important to evaluate the performance of facial emotion recognition models on other datasets and in different scenarios.

Ozcan(2020) use transfer learning with hyperparameter optimization for FER on static images. They utilized hyperparameter optimization to increase the accuracy of the model. This work was experimented on JAFFE and ERUFER datasets. Gogić(2020) developed a joint optimization framework for FER using local binary features and shallow networks with improved execution time. The hybrid deep learning model was developed by Garima and Hemraj (2020) for facial expression recognition. Here, the primary emotion being sad or joy was identified by one CNN and secondary CNN recognizes the secondary emotion of the image. This work was tested on FER2013, and JAFFE datasets. All the mentioned works produced good results on human-based datasets but these models are sensitive to the illumination and specific poses present in that dataset because these models are evaluated on a single kind of dataset.



Universal Facial Emotions

Figure1.1

1.4 MOTIVATION

The motivation behind the project of facial emotion recognition (FER) is to develop a computer vision system that can automatically detect and interpret human emotions from facial images and videos. FER can have many potential applications in various fields such as video surveillance, human-computer interaction, and healthcare. In video surveillance, for example, FER can be used to monitor public spaces for suspicious behavior or to detect emotional states of drivers in cars to ensure safety. In human-computer interaction, FER can be used to make more natural and intuitive interactions between humans and machines, and in healthcare, FER can be used to detect and monitor emotional states of patients with mental disorders.

Another motivation behind the FER project is that emotional information is conveyed through the face and it is an important aspect of human communication. Therefore, the ability to automatically detect and interpret emotions from facial images and videos can be beneficial for human-human and human-computer interactions. Additionally, FER can also help in understanding human behavior and emotions better, which can be useful in fields such as psychology, neuroscience, and sociology.

Moreover, to further expand our knowledge and carry out a research work on a crucial field in artificial intelligence motivated us to work on this project. We are deeply heartened to be able to develop a web application with an enhanced accuracy than the previously proposed works.

In summary, the motivation behind the project of facial emotion recognition is to develop a computer vision system that can automatically detect and interpret human emotions from facial images and videos, which can have many potential applications in various fields such as video surveillance, human-computer interaction, and healthcare. Additionally, the ability to automatically detect and interpret emotions from facial images and videos can be beneficial for human-human and human-computer interactions, and for understanding human behavior and emotions better.

1.5 ORGANIZATION OF PROJECT REPORT

We aimed to designed a comprehensive, lucid and factually correct report for the project so presented. Since the application is AI-based, special emphasis in the report was laid on the future scope of the technology and the research we undertook while designing it. Research being the most central part of deep learning and AI, the research paper so written was carefully included in the report.

Snapshots of the project were included to give the reader a clear picture of what the application does. In the same sentiment, methodology was presented through various UML diagrams to enforce OOAD principles as well as present the work in an industrially relevant manner. Special care was taken while drawing the UML diagrams to give a clear picture of what the application does and how it achieves it. It was made sure that the idea presented in the abstract was clear and shed light to the importance of the work. It was of utmost importance to us that every respected designator was mentioned without whose guidance the work would have been lackluster.

The report so compiled was arranged in the most logical order and the index was updated to make each section, table and figure easily accessible.

METHODOLOGY ADOPTED

1.1 STRATEGY

The first phase of this project was to define a strategy for evolving the idea into a successful web application. In this phase we tried to:

- Identify our users
- Assess resources, objectives and capabilities
- Research the competition
- Established the Web App goals

1.2 OBJECT ORIENTED ANALYSIS AND DESIGN

1.2.1 DFD

A Data Flow Diagram (DFD) is a graphical representation of the flow of data through an information system. It shows how information is input to and output from the system, the sources and destinations of that information and where that information is stored.

- **Context Level Diagram: (0 level diagram)**

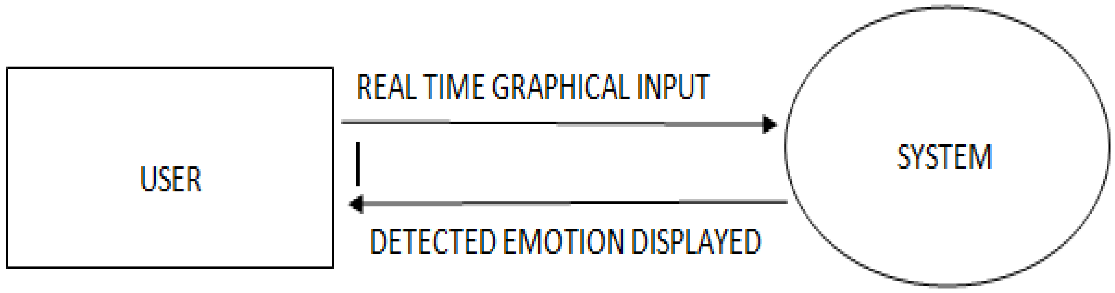


Figure 2.1

- **Level 1 DFD**

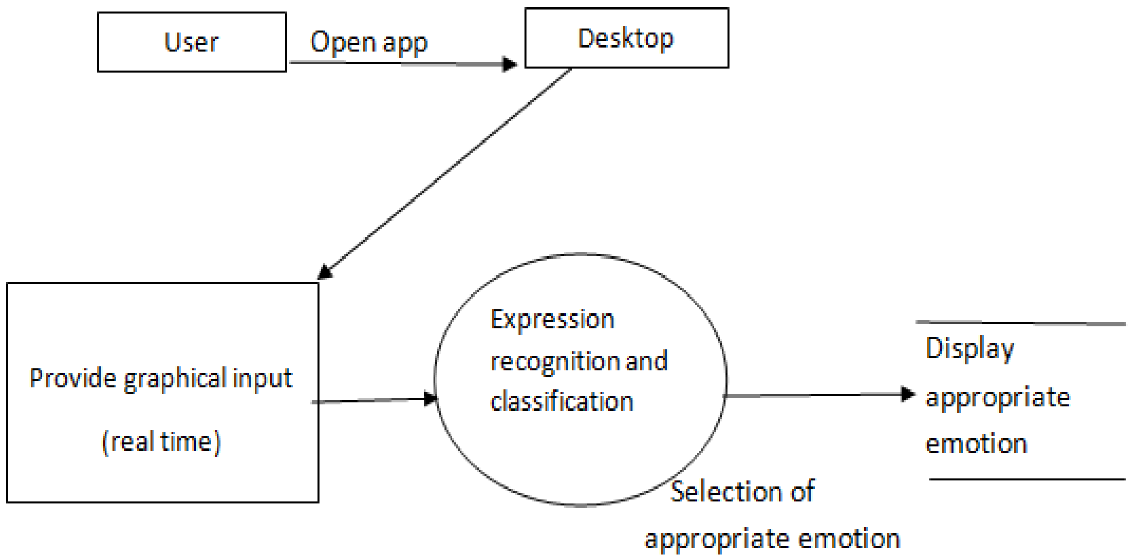

User
Open app
Desktop
Provide graphical input
(real time)
Expression recognition and classification
Display appropriate emotion
Selection of appropriate emotion

Figure 2.2

2.2.2 Use Case Diagram

Use case diagrams are a type of diagram that depicts the interactions between users (or "actors") and a system, in order to visualize and understand the system's requirements. They are used in the field of software engineering, and are part of the Unified Modeling Language (UML) standard for modeling software systems. A use case diagram typically consists of a number of elements, including: Actors, Use Cases, System Boundary, Relationships.

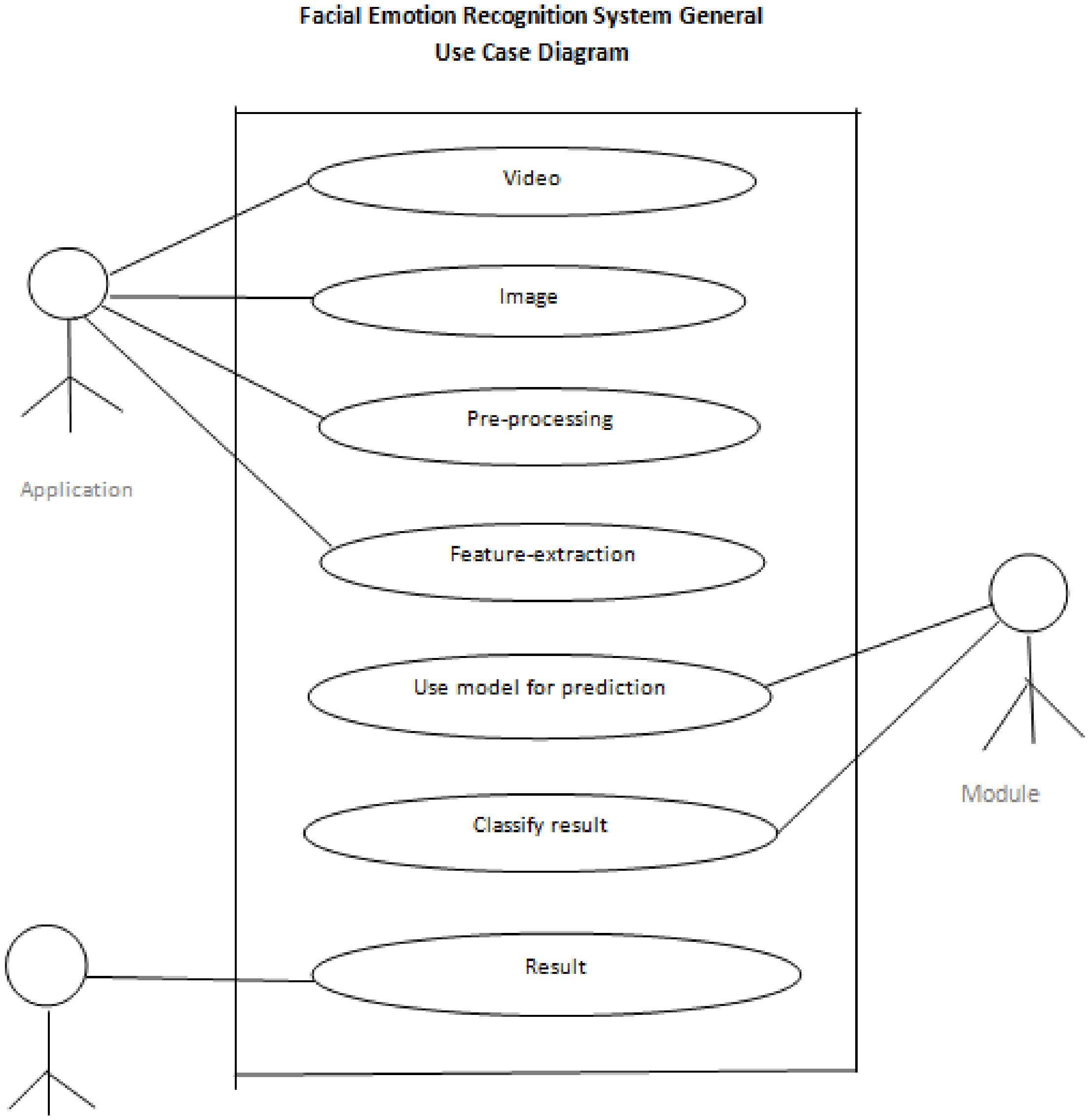


Figure 2.3

2.2.3 Description of the Use Case and Actor

A use case diagram is a dynamic or behavior diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform. In this context, a "system" is something being developed or operated, such as a web site. The "actors" are people or entities operating under defined roles within the system. The “scenario” is a specific sequence of actions and interactions between actors and the system. "Use case” is a collection of related success and failure scenarios, describing actors using the system to support a goal.

2.2.4 Sequence Diagram

A sequence diagram is a type of diagram that represents the interactions between objects or components in a system, showing the order in which those interactions occur. It is used to model the dynamic behavior of a system, and is part of the Unified Modeling Language (UML) standard for modeling software systems. A sequence diagram is composed of a series of objects or components represented by rectangles, and shows how messages or interactions are exchanged between those objects, as well as the order in which they occur.

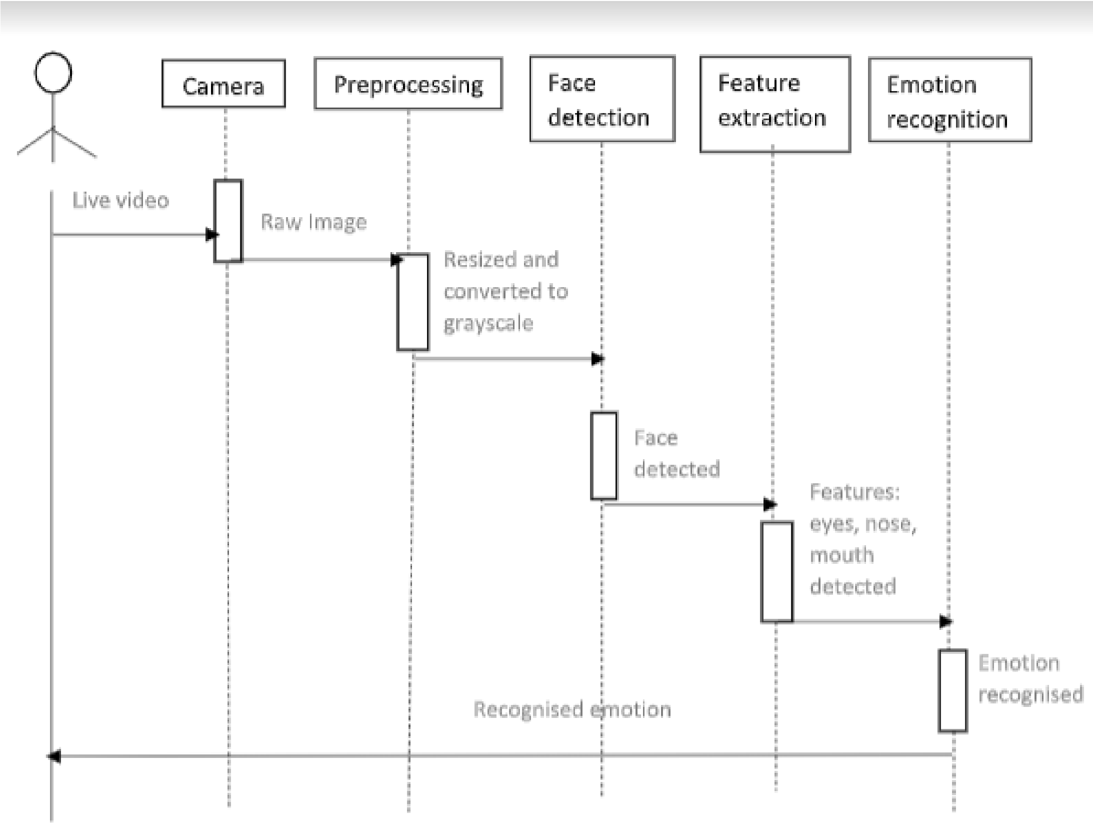


Figure 2.4

2.2.5 Activity Diagram

An activity diagram is a type of flowchart that depicts the flow of activities or actions within a system. It is used to model the dynamic behavior of a system, and is part of the Unified Modeling Language (UML) standard for modeling software systems. An activity diagram is composed of a series of interconnected activities represented by rounded rectangles, and shows how activities flow from one to another, as well as decision points, loops and parallel activities.

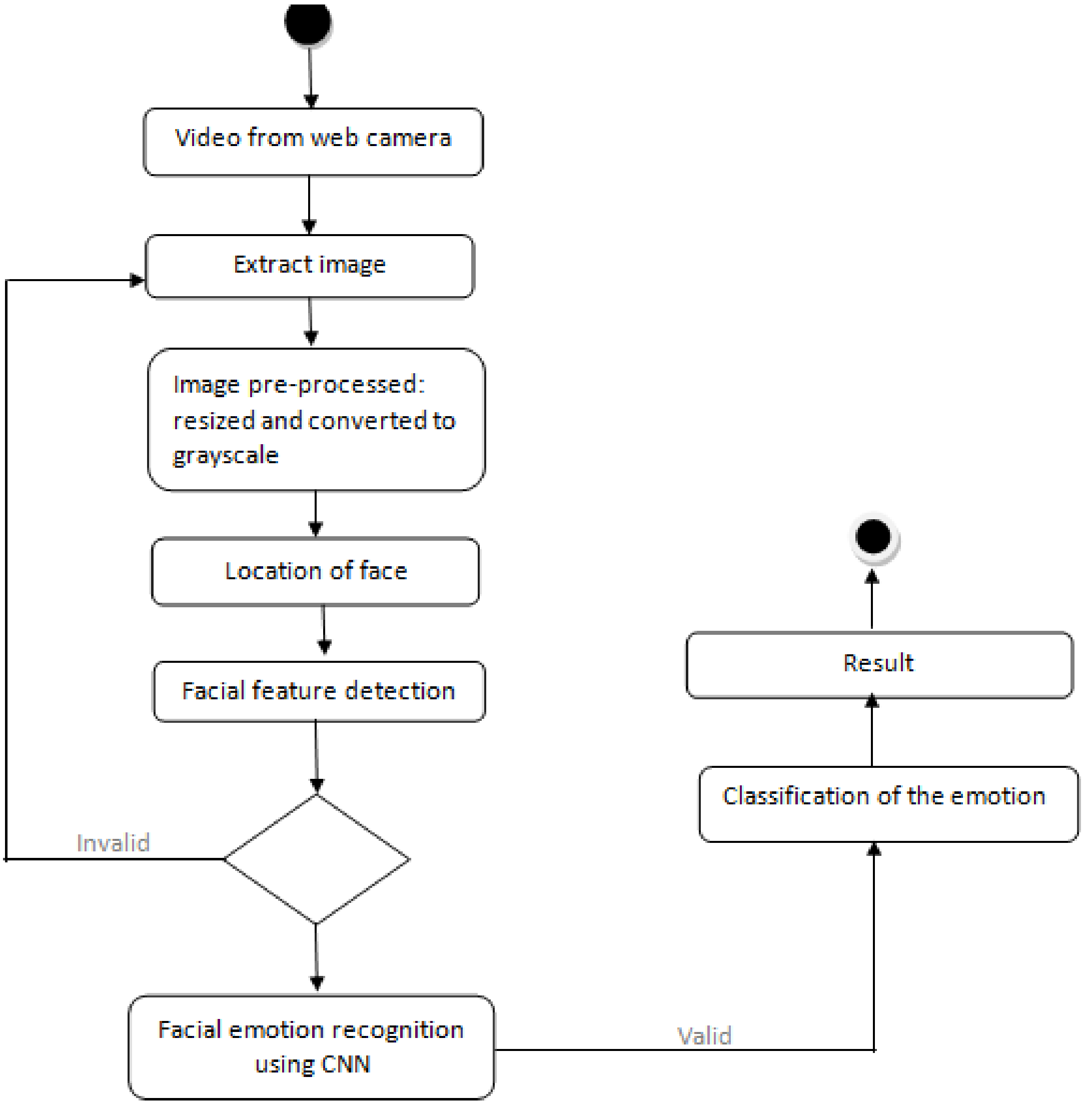

Video from web camera
Extract image
Image pre-processed: resized and converted to grayscale
Location of face
Facial feature detection
Invalid
Facial emotion recognition using CNN
Valid
Classification of the emotion
Result

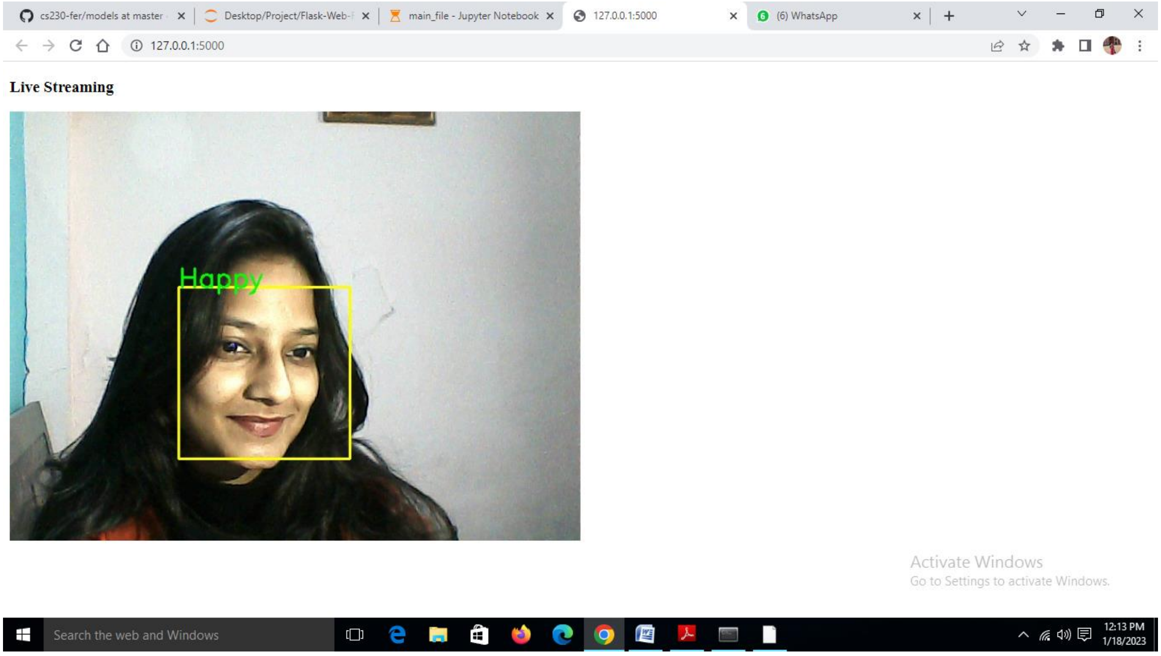
Figure 2.5

1.3DIGITAL ANALYSIS

Layout

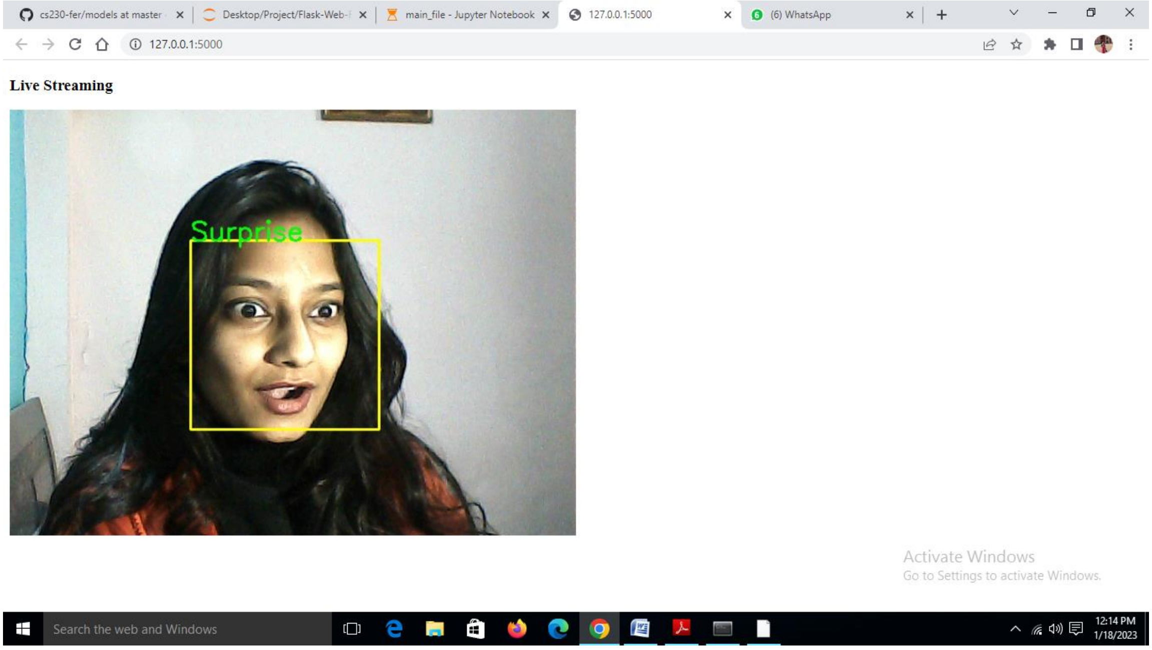
1. The system captures an image of a face using a camera.
2. The image is passed through a pre-processing module where it is resized and converted to grayscale.
3. The processed image is then passed through a face detection module which detects the location of the face in the image.
4. The detected face is then passed through a facial feature detection module which detects the key facial features such as eyes, nose, and mouth.
5. The facial features are then passed through an emotion recognition module which uses machine learning algorithms to classify the emotions expressed on the face.
6. The detected emotions are then displayed on the screen.
7. The system keep continuously capturing images and the process repeats

Snapshots of working project


cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Live Streaming
Happy
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:13 PM
1/18/2023

Emotion Detected: Happy

Figure 2.6

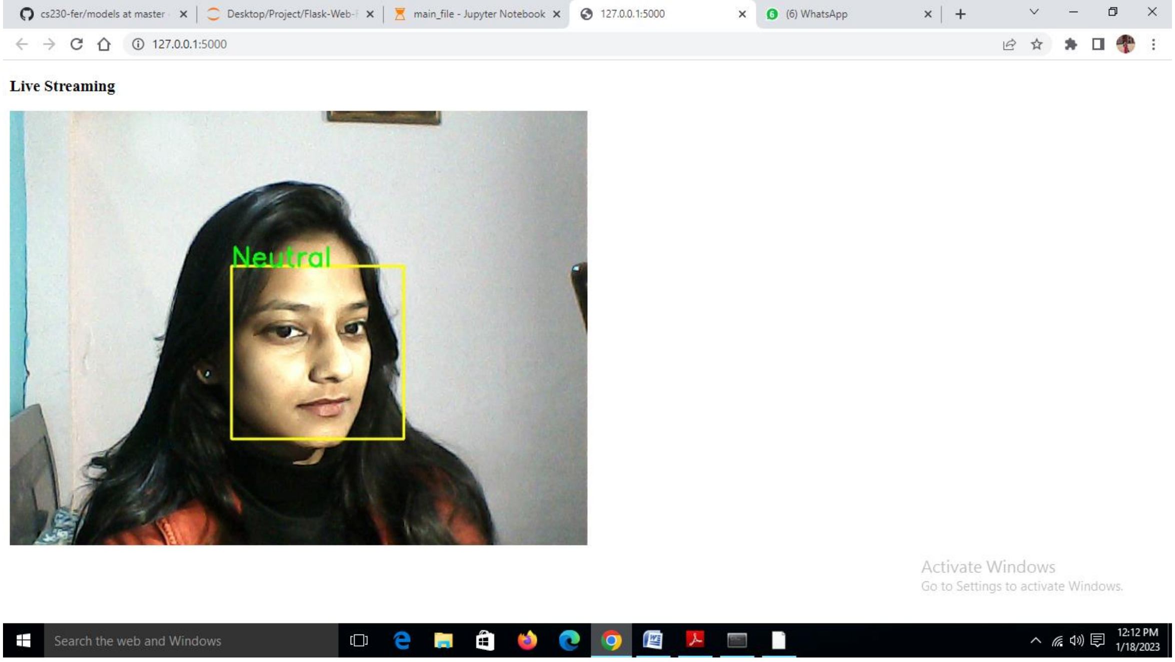


Live Streaming

cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Surprise
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:14 PM
1/18/2023

Emotion Detected: Surprise

Figure 2.7

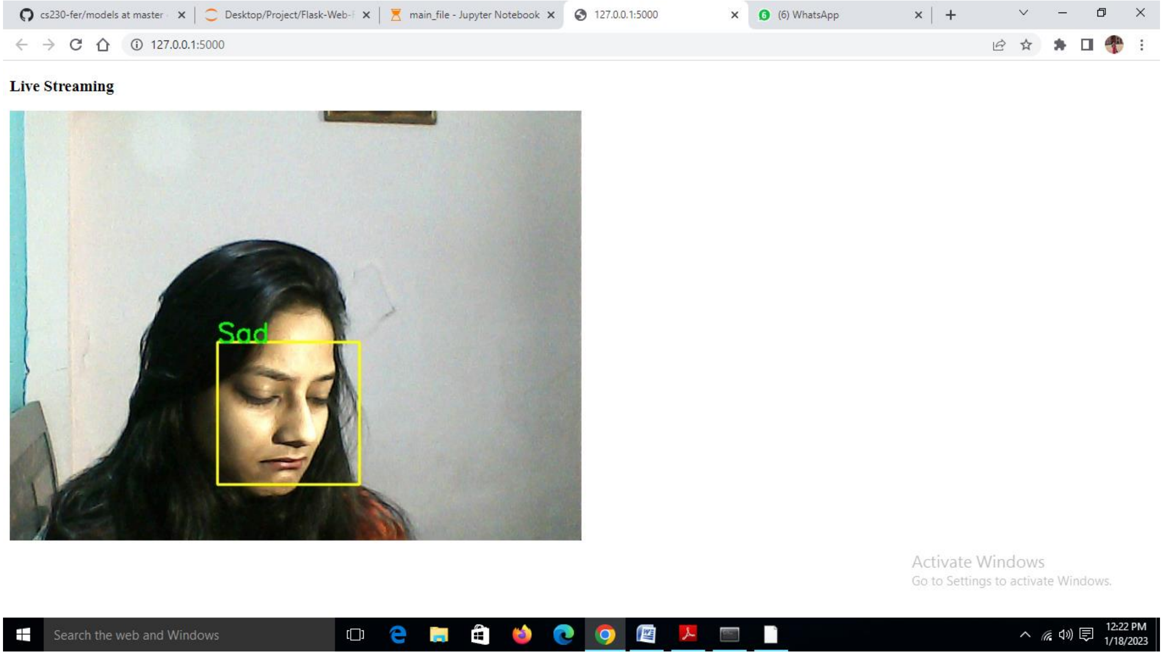


Live Streaming

cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Neutral
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:12 PM
1/18/2023

Emotion Detected: Neutral

Figure 2.8

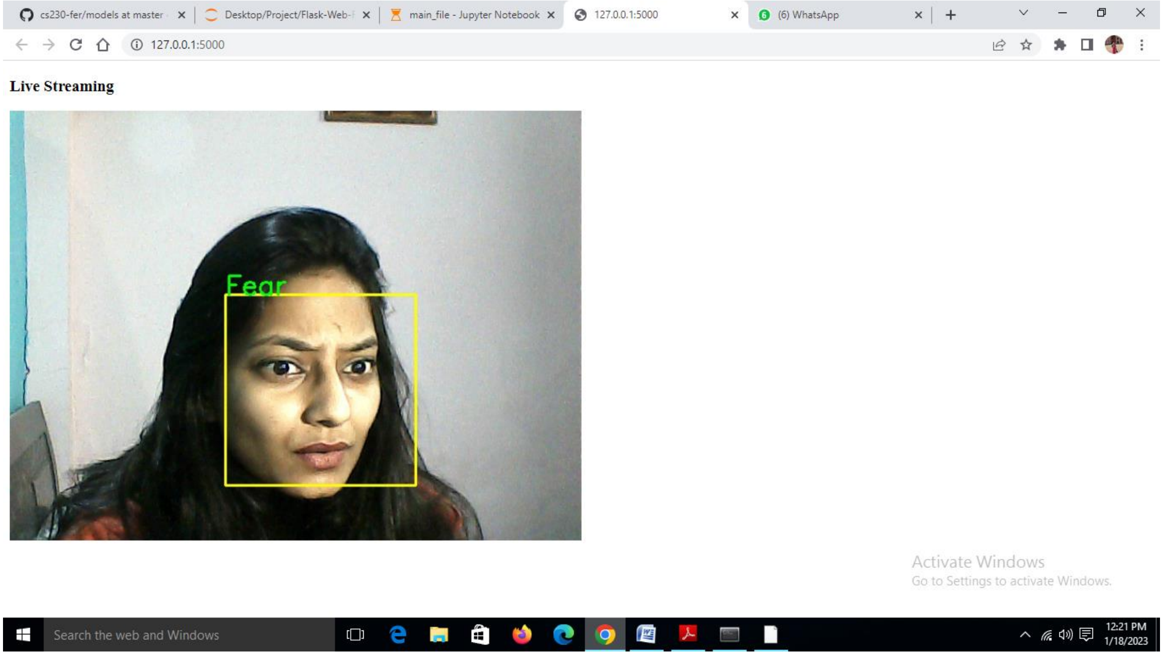


cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Sad
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:22 PM
1/18/2023

Live Streaming

Emotion Detected: Sad

Figure 2.9

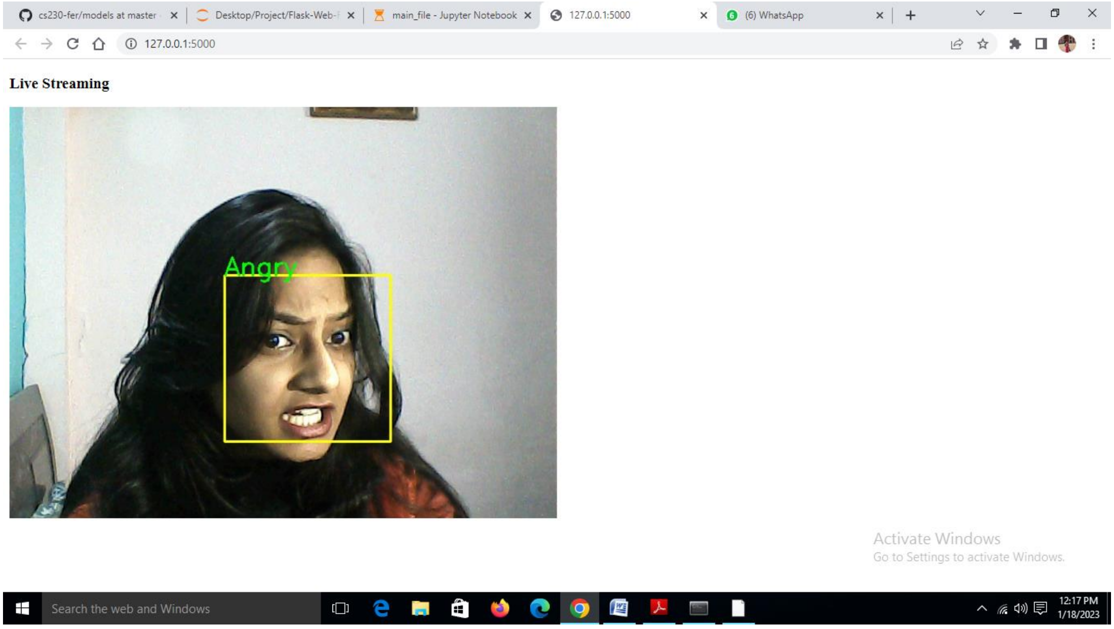


cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Fear
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:21 PM
1/18/2023

Live Streaming

Emotion Detected: Fear

Figure 2.10



cs230-fer/models at master
Desktop/Project/Flask-Web-
main_file - Jupyter Notebook
127.0.0.1:5000
(6) WhatsApp
127.0.0.1:5000
Angry
Activate Windows
Go to Settings to activate Windows.
Search the web and Windows
12:17 PM
1/18/2023

Live Streaming

Emotion Detected: Angry

Figure 2.11

DESIGNING AND RESULT ANALYSIS

3.1 PROJECT INSIGHTS

3.1.1 FRONT END AND BACK END

Front end and back end: Frontend and Backend are two most popular terms used in web development. These terms are very crucial for web development but are quite different from each other. Each side needs to communicate and operate effectively with the other as a single unit to improve the website's functionality.

3.1.2 FRONT END LANGUAGES USED IN OUR APPLICATION:

- **HTML:** The Hyper Text Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript

- **CSS:** Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.

- **JavaScript:** JavaScript (JS) is a lightweight interpreted or JIT-compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js.

3.1.3 BACK END LANGUAGES USED IN OUR APPLICATION

- **Python:** Python is a high-level, general-purpose and a very popular programming language. Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse.

- **Flask:** Flask is a micro web framework written in Python. It is classified as a micro-framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself.

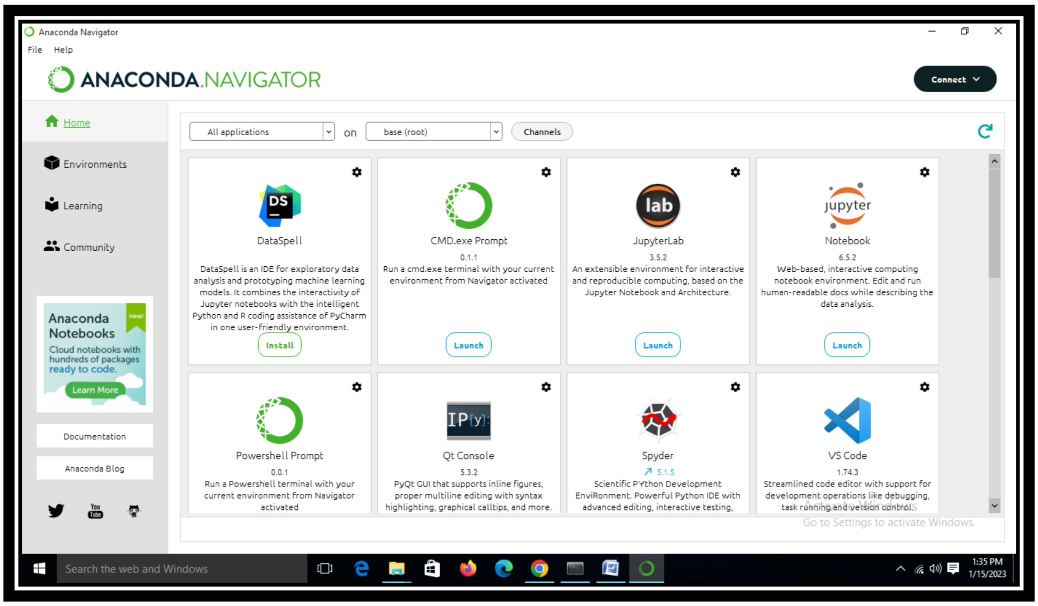
3.1.4 SOFTWARES USED IN OUR APPLICATION

- **NGROK**

  While running flask apps from local machine we need to use LOCAL-HOST, but services such as Google Colab provides virtual machine hence we do not access the local-host. NGROK software provides a way to share our local host server with anyone, anywhere, using a secure tunnel to our local machine.

- **ANACONDA**

  Anaconda is an open-source package and environment management system that runs on Windows, macOS, and Linux. Conda quickly installs, runs, and updates packages and their dependencies. It also easily creates, saves, loads, and switches between environments on local computer. We have specifically used **Jupyter Notebook** on Conda for running python files and implementing opencv library.



DataSpell

DataSpell is an IDE for exploratory data analysis and prototyping machine learning models. It combines the interactivity of Jupyter notebooks with the intelligent Python and R coding assistance of PyCharm in one user-friendly environment.

CMD.exe Prompt 0.1.1

Run a cmd.exe terminal with your current environment from Navigator activated

JupyterLab 3.5.2

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook 6.5.2

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Powershell Prompt 0.0.1

Run a Powershell terminal with your current environment from Navigator activated

Qt Console 5.3.2

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder 5.1.5

Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing,

VS Code 1.74.3

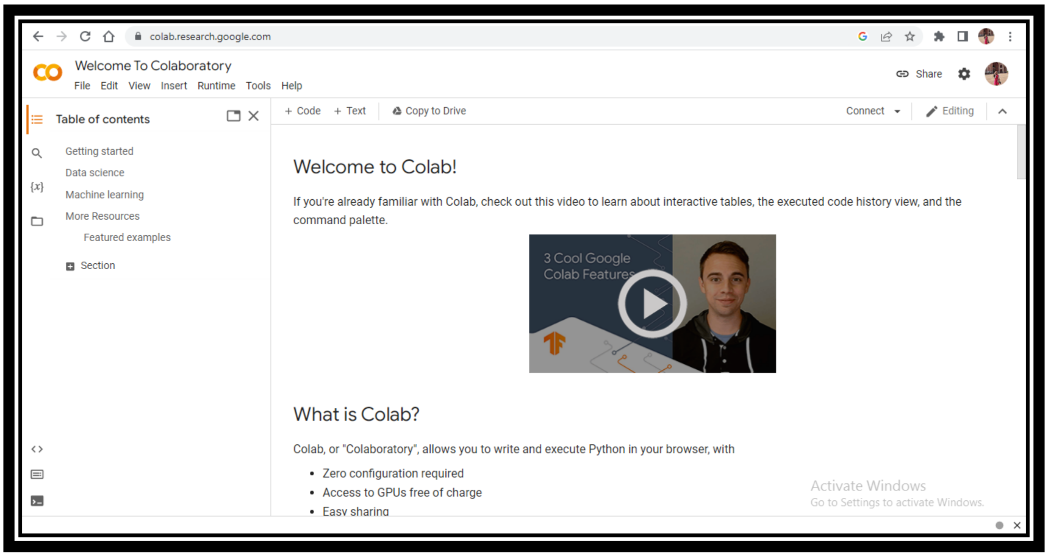
Streamlined code editor with support for development operations like debugging, task running and version control.

Anaconda Navigator

Figure 3.1

- ## <u>GOOGLE COLAB</u>

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. We used Colab hosted Jupyter notebook service that requires no setup to use, for training our model free of charge, accessing computing resources like GPUs.



Welcome to Colab!

If you're already familiar with Colab, check out this video to learn about interactive tables, the executed code history view, and the command palette.

What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

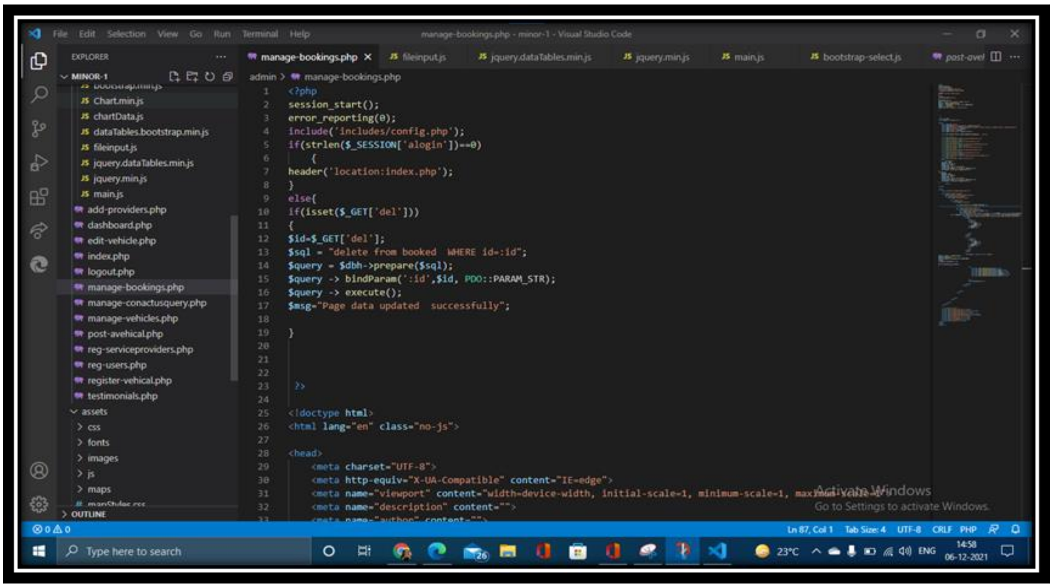
- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Google Colab

Figure 3.2

- **VISUAL STUDIO CODE**

Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git.



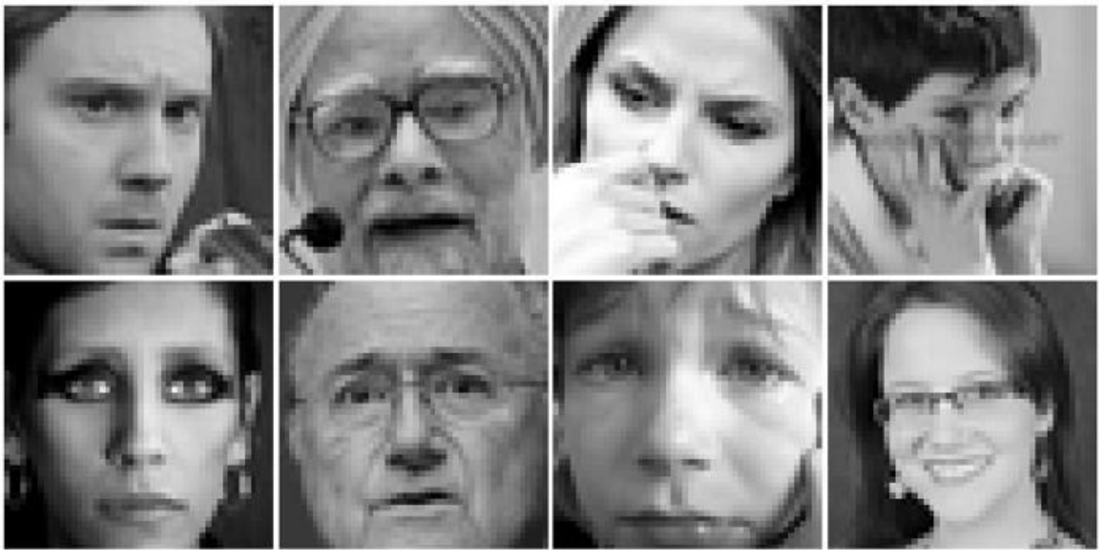
```
<?php
session_start();
error_reporting(0);
include('includes/config.php');
if(strlen($_SESSION['alogin'])==0)
    {
header('location:index.php');
}
else{
if(isset($_GET['del']))
{
$id=$_GET['del'];
$sql = "delete from booked  WHERE id=:id";
$query = $dbh->prepare($sql);
$query -> bindParam(':id',$id, PDO::PARAM_STR);
$query -> execute();
$msg="Page data updated  successfully";

}



 ?>

<!doctype html>
<html lang="en" class="no-js">

<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1, minimum-scale=1, max
    <meta name="description" content="">
```

Visual Studio Code
Figure 3.3

3.1.5 Datasets used for Training

- **FER2013:** The images in the FER2013 dataset were collected from a variety of sources, including the internet, and they were pre-processed and aligned to a standard size of 48x48. The dataset consists of 35,887 grayscale images of faces, each labeled with one of seven possible emotion categories: angry, disgusted, fearful, happy, sad, surprised, and neutral and is divided into three sets: a training set of 28,709 images, a validation set of 3,589 images, and a test set of 3,589 images.

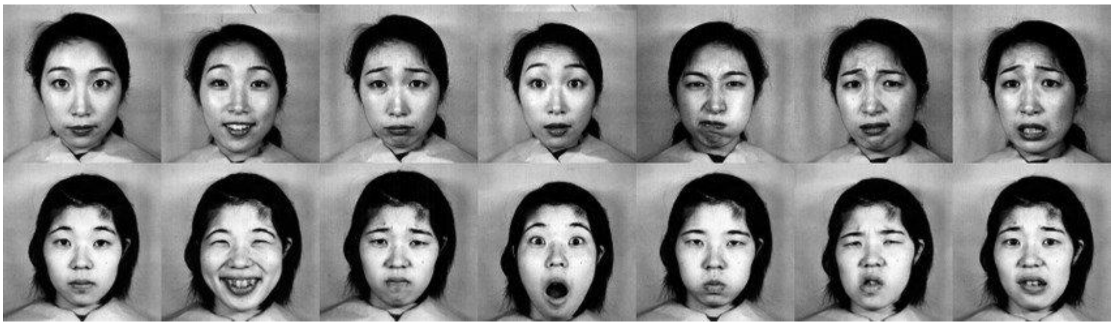
  The dataset is considered to be challenging due to its diverse set of images with variations in pose, lighting, and expression intensity, achieving only 65±5% human-level accuracy and the highest performing published works achieving 75.8% test accuracy. FER2013 is, however, not a balanced dataset, as it contains only 547 images labeled 'disgust', which is 91% less than the average distribution of other 6 emotions.



Images from FER2013 Dataset

Figure 3.4

- **JAFFE Dataset:** The Japanese Female Facial Expression (JAFFE) is a relatively small dataset containing 213 256x256 grayscale images of 10 Japanese women. It covers 7 facial expressions, 6 basic and 1 neutral with several images for each expression. The six basic facial expressions covered are anger, fear, sadness, happiness, surprise, and disgust.



Images from JAFFE Dataset

Figure 3.5

- **Extended Cohn-Kanade Dataset:** The extended Cohn-Kanade dataset (CK+) was compiled by taking snapshots of each of the 7 expressions posed in the 593 video sequences of the original dataset. The compiled dataset contains 981

48x48 grayscale images of subjects ranging from 18 to 50 years of age with variety of genders and heritage. The images are classified into emotions like anger, disgust, contempt, sadness, fear, surprise and happiness.

Images from CK+ Dataset
Figure 3.6

Datasets	Happy	Angry	Sad	Neutral	Disgust	Fear	Surprise
FER2013	8989	4953	6077	6198	547	5121	4002
JAFFE	30	30	31	30	29	33	30
CK+	90	90	90	90	90	90	90S

Distribution of Emotions in Dataset
Table 1

3.1.6 Model Architecture

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 46, 46, 64)        640

 conv2d_1 (Conv2D)           (None, 46, 46, 64)        36928

 batch_normalization (BatchN  (None, 46, 46, 64)       256
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 23, 23, 64)       0
 )
```

```
dropout (Dropout)           (None, 23, 23, 64)        0

conv2d_2 (Conv2D)           (None, 23, 23, 128)       73856

batch_normalization_1 (Batc  (None, 23, 23, 128)      512
hNormalization)

conv2d_3 (Conv2D)           (None, 23, 23, 128)       147584

batch_normalization_2 (Batc  (None, 23, 23, 128)      512
hNormalization)

max_pooling2d_1 (MaxPooling  (None, 11, 11, 128)      0
2D)

dropout_1 (Dropout)         (None, 11, 11, 128)       0

conv2d_4 (Conv2D)           (None, 11, 11, 256)       295168

batch_normalization_3 (Batc  (None, 11, 11, 256)      1024
hNormalization)

conv2d_5 (Conv2D)           (None, 11, 11, 256)       590080

batch_normalization_4 (Batc  (None, 11, 11, 256)      1024
hNormalization)

max_pooling2d_2 (MaxPooling  (None, 5, 5, 256)        0
2D)

dropout_2 (Dropout)         (None, 5, 5, 256)         0

conv2d_6 (Conv2D)           (None, 5, 5, 512)         1180160

batch_normalization_5 (Batc  (None, 5, 5, 512)        2048
hNormalization)

conv2d_7 (Conv2D)           (None, 5, 5, 512)         2359808

batch_normalization_6 (Batc  (None, 5, 5, 512)        2048
hNormalization)

max_pooling2d_3 (MaxPooling  (None, 2, 2, 512)        0
2D)

dropout_3 (Dropout)         (None, 2, 2, 512)         0

flatten (Flatten)           (None, 2048)              0

dense (Dense)               (None, 512)               1049088

dropout_4 (Dropout)         (None, 512)               0

dense_1 (Dense)             (None, 256)               131328

dropout_5 (Dropout)         (None, 256)               0

dense_2 (Dense)             (None, 128)               32896
```

```
 dropout_6 (Dropout)         (None, 128)              0

 dense_3 (Dense)             (None, 7)                903

=================================================================
Total params: 5,905,863
Trainable params: 5,902,151
Non-trainable params: 3,712
```

Architecture of Deep CNN model
Table 2

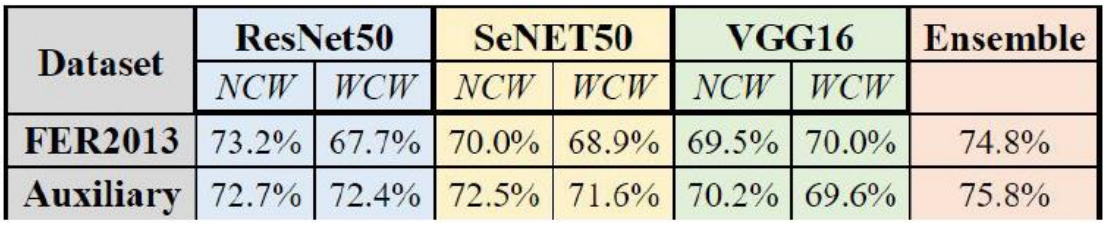
3.2 RESULTS AND DISCUSSION

Most of the publications which achieved state-of-the-art accuracies on FER2013 utilized auxiliary training data.

Model	**Accuracy**
(Human-Level)	65±5%
Tang[4]	71.2%
Pramerdorfer[2]	75.2%
Baseline	64%
Five-Layer Model	66.3%
VGG16	70.2%
InceptionV3	73.4%
SeNet50	72.5%
ResNet50	73.2%
Ensemble	75.8%
Proposed model	76.2%

Results on FER2013
Table 3

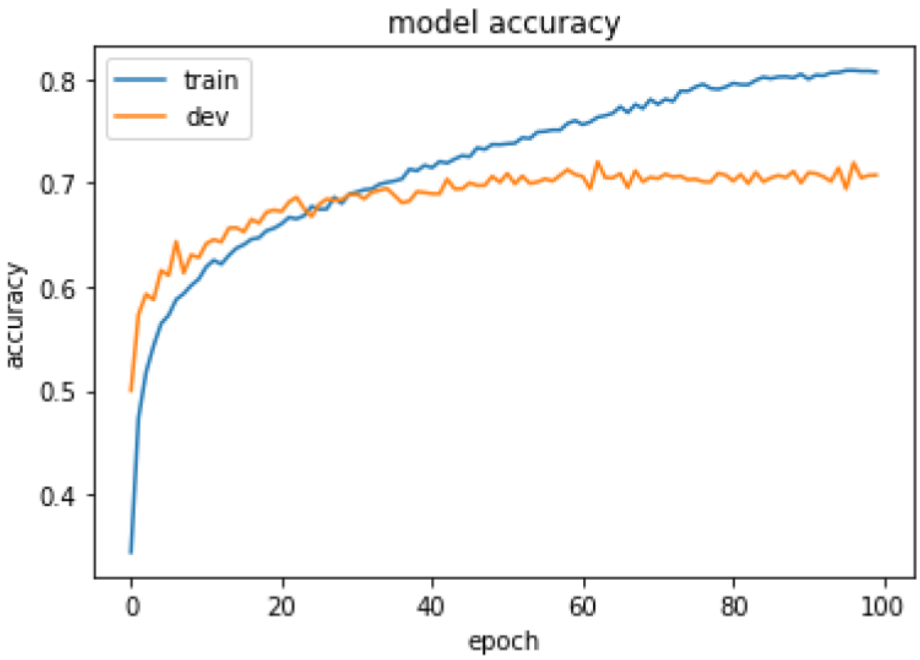
The table below demonstrates our accuracy gains from employing auxiliary data with care taken to avoid dataset bias. It also depicts our success in implementing class weighting, which significantly increased accuracies on frequently misclassified emotions. Hyperparameter turning of the deep CNN model and pre-trained models helped achieved the best accuracy yet stated. Ensembling several models with and without class weights improved the overall accuracy.



Dataset	ResNet50		SeNET50		VGG16		Ensemble
	NCW	*WCW*	*NCW*	*WCW*	*NCW*	*WCW*	
FER2013	73.2%	67.7%	70.0%	68.9%	69.5%	70.0%	74.8%
Auxiliary	72.7%	72.4%	72.5%	71.6%	70.2%	69.6%	75.8%

Table 3. Accuracies for different methods with and without auxiliary data.

Given the high complexity of transfer learning models and relatively small size of our datasets, we also experienced overfitting while training. Although we added 50% dropout for the last three layers, our ResNet50 transfer learning model quickly overfit to the training set with dev accuracy starting to flatten after only 30 epochs.



Learning curve for ResNet50 WCW transfer learning model
Figure 3.7

It is worth mentioning that because our models exceeded human-level accuracy, error analysis was particularly challenging for some misclassifications, such as the fear image discussed prior. Additionally, because emotions are highly subjective, Bayes error is high and it is often the case that an image can have multiple interpretations.

Auxiliary Data & Data Preparation

Although several FER datasets are available online, they vary widely in image size, color, and format, as well as labeling and directory structure. During training, we loaded images in batches of 64 from disk (to avoid memory overflow) and utilized Keras data generators to automatically resize and format the images.

Data Augmentation

We researched and experimented with commonly used techniques in existing FER papers and achieved our best results with horizontal mirroring, ±10 degree rotations, ±10% image zooms, and ±10% horizontal/vertical shifting.

Class Weighting

To alleviate the class imbalance problem, we applied class weighting inversely proportional to the number of samples. For the disgust class, we were able to drop the misclassification rate from 61% to 34%.

SMOTE

The Synthetic Minority Over-sampling Technique (SMOTE) involves oversampling minority classes and under sampling majority classes to get the best results. Although utilizing SMOTE resulted in a perfectly balanced training dataset, our models quickly overfit to the training dataset and we decided not to experiment further.

Ensembling

We performed ensembling with soft voting of seven models to significantly improve our highest test accuracy from 75.8% to 76.2%.

MERITS, DEMERITS AND APPLICATIONS

4.1 MERITS

Facial emotion recognition has many potential benefits. Some of the main merits of our project include:

- **Improving human-computer interaction**: It can be used to improve the naturalness and expressiveness of human-computer interaction by detecting the user's emotions and adapting the system's behavior accordingly.
- **Enhancing video analysis**: It can be used to detect emotions in video sequences, which can be used in applications such as surveillance, personalization, and behavior understanding.
- **Healthcare applications**: FER can be used to monitor the emotions of patients with conditions such as depression, anxiety, and schizophrenia, which can help with the diagnosis and treatment of these conditions.
- **Facial expression analysis**: FER allows for the study of facial expressions and emotions, which can be used to better understand human behavior and emotions.
- **Emotion-based personalization**: It can be used in applications such as music, movies, and gaming to provide a more personalized experience based on the user's emotions.
- **Automated emotions recognition**: FER can help in automating the process of emotions recognition which otherwise is a time-consuming and subject to human error.
- **Multimodal approach**: It can be used in combination with other modalities such as speech, physiological signals and text to improve the performance of emotion recognition.

It has many potential applications in various fields, such as human-computer interaction, video analysis, healthcare, and personalization. With the development of new techniques and increasing amount of data, FER is expected to play an important role in many areas of research and industry.

4.2 DEMERITS

Facial emotion recognition (FER) is a complex task that has many challenges and limitations. Some of the main demerits of FER projects include:

- **Variability in facial expressions**: People express emotions in different ways, and even the same person may express the same emotion differently in different situations. This variability in facial expressions can make it difficult for FER models to accurately detect and classify emotions.
- **Limited diversity in training data**: Many FER datasets have been collected in controlled environments and may not be representative of the diversity of people and emotions in the real world. This can lead to a lack of generalizability of FER models, particularly for people with different ethnicities, ages, or genders.
- **Bias in models**: FER models can be biased towards certain groups of people, such as people with darker skin tones, which can lead to inaccurate or unfair predictions.
- **Privacy concerns**: FER systems can be used to track and analyze people's emotions without their knowledge or consent, raising privacy concerns.
- **Complexity**: FER is a complex problem that requires the integration of multiple techniques from computer vision, machine learning, and psychology. It requires large amount of data, computational resources and a lot of time for training.
- **Limited use cases**: FER has limited use cases and it's not applicable to all the scenarios, because emotions can be expressed through multiple modalities such as speech, text, physiological signals and action.

While FER has the potential to be a valuable tool in various applications, it's important to be aware of these demerits and limitations and to take steps to mitigate them in order to develop accurate and fair FER systems.

4.3 APPLICATIONS

- One of the main focuses of artificial intelligence today is human-machine interaction. Ameca is the most advanced humanoid robot, developed by Engineered Arts, a British AI based company specifically for research on human-machine interaction. Since machines are often deployed in areas inhabited or frequented by humans, it is necessary that AI is able to better understand human emotions from facial expressions.
- Surveillance and security: FER can be used to detect emotions in video sequences, which can be used in applications such as surveillance, personalization, and behavior understanding.
- Marketing and advertising: FER can be used to analyze the emotions of people viewing ads and videos, which can help to improve the effectiveness of marketing campaigns.
- Gaming and entertainment: FER can be used in gaming and entertainment applications to provide a more personalized experience based on the user's emotions.
- Companies deploy emotional recognition AI at their call centers to enhance customer service. Such solutions have many benefits. They can pick the best fitting agent for a specific client, give real-time feedback to agents and notify them when they start to lose control, and respond in kind to a frustrated customer. Artificial intelligence algorithms can also analyze incoming support tickets and identify clients on the verge of cutting ties with the company. American tech giants offer basic emotion analysis for specific sectors such as automotive, advertisers and recruiters.
- Human resources: FER can be used in recruitment and selection processes to identify candidate's emotions to increase the accuracy of predictions about candidate's suitability for a role.
- Transportation: FER can be used to monitor the emotions of drivers and passengers in cars, buses and trains to improve the safety and comfort of transportation.
- Robotics: FER can be used to make robots more expressive, empathetic, and socially aware.

- Law enforcement: FER can be used to detect emotions in suspects and victims during police interviews, which can help with investigations and court proceedings.
- This emotional artificial intelligence type can be used in interview settings to detect whether candidates are nervous, confident, genuine, etc. Emotion detection applications have been used to examine suitability of candidates for the job in remote interviews.
- Facial emotion analysis can be used to enhance experience of video games by reading the players' expressions. On the same wavelength, they can be used to read emotions of viewers of film and TV and use the data for future development. In fact, Disney has used emotion detection software to test volunteers’ reactions to a range of its films including Star Wars: The Force Awakens and Zootopia.
- Facial emotion detection can be used by automobile companies to assert driver alertness and is currently in use by BMW, Ford and Kia Motors for the same. Emotional recognition systems can be used to identify suspicious people by the police.

CONCLUSION AND SCOPE OF THE PROJECT

5.1 CONCLUSION

We developed a facial expression recognition system to detect seven facial expression based on facial mimics in real time. It proposes a fast and accurate deep learning-based approach for automatic facial expression recognition. The proposed models outperform the state of existing works onFER2013 which is challenging datasets due to intra-class variation and inter-class similarities. We explored several models including Deep CNNs and pre-trained networks based on InceptionV3, SeNet50 and ResNet50. To alleviate FER2013’s inherent class imbalance, we employed class weights, data augmentation, and auxiliary datasets. To enhance the accuracy of each model separately, hyperparameter tuning of these models were performed. By ensembling four models and tuning our architecture, we achieved an accuracy of 76.2% on the challenging FER2013 dataset, which is the highest to our knowledge. We also found through network interpretability that our models learned to focus on relevant facial features for emotion detection. Finally, we demonstrated that facial emotion recognition models could be applied in the real world by developing a flask-based web application with real-time recognition speed, minimal memory, disk, and computational requirements. The web application classified facial emotions of a person that were being captured by the webcam, into seven universal facial emotions, namely, angry, sad, happy, disgusted, surprised, fearful and neutral, which is often described as a resting face with no emotion depicted. However, there are still many challenges to be addresses, such as dealing with unseen facial expressions, cross-cultural variations, and privacy issues. Therefore, further research on FER is necessary to improve the performance and robustness of the models and to explore new applications.

5.2 SCOPE OF THE PROJECT

To further improve the accuracy of our models, we hope to utilize facial landmark detection and alignment implement attention CNNs, and retrain our network by occluding facial features irrelevant to emotion recognition. We would also like to employ more auxiliary data (in particular AffectNet which contains over a million labeled images) and balance our training dataset with methods such as ADASYN. Furthermore, we believe there is great potential for improvement with pipeline models, where commonly misclassified emotion pairs (e.g. neutral and sad) are fed to secondary networks with higher accuracy rates between those specific emotions.
To further adapt our models to the real world, we hope to integrate contemporary Psychological research, particularly the arousal-valence emotional model, and also utilize multi-label classification to better handle images with multiple possible emotion labels. Moreover, AI emotion recognition algorithms help marketers understand which ads resonate better with the target audience and what features they should include in their videos to yield a better outcome. It can be used to analyze people's emotional reactions to the adverts on the large billboards. Emotional recognition systems can be used to identify suspicious people by the police.

There are still many challenges to be addresses, such as dealing with unseen facial expressions, cross-cultural variations, and privacy issues. Moreover, more experimentation could be done with the problem of data imbalance. We would like to improve the robustness and accuracy of our web app model by including a web app dataset and also applying different data augmentation techniques to address varying camera brightness and angle issues. For better deployment results, one could use a quantized model for face detection that could run on the GPU/TPU. Additionally, we would like to apply our work to benefit humanity, such as by employing it to support shared empathy.

RESEARCH PAPER

Thespis.AI: Real Time Facial Emotion Recognition using Ensemble of Deep CNNs

Archi Agrawal[1], Taiyaba Zaheer[2], Rishabh Sharma[3]
[1,2,3]Dr. Akhilesh Das Gupta Institute of Technology and Management, Delhi

Abstract – **Emotions are an inevitable portion of any inter-personal communication. There has been an active research over the last few decades in this field but it is still challenging due to the high intra-class variation. This paper proposes an ensemble of various deep Convolutional Neural Networks models with a voting technique for facial expression recognition (FER). To make better predictions, the hyperparameters of the proposed model were tuned. This model achieved a state-of-the-art 76.2%accuracy on the publicly available FER2013 test set, outperforming all existing publications. Additionally, we showcase a web app which runs our FER models on-device in real time.**

I. INTRODUCTION

Recognizing facial expressions is fundamentally important in human communication and is an active area of research that has a wide range of potential applications. The technology is based on machine learning algorithms, and recent developments in deep learning techniques have led to significant improvements in the accuracy while categorizing emotions from facial images, exceeding human level performance. Convolution Neural Network (CNN) based models are very robust and performing well in facial expression classification tasks. In CNN, convolutional filter parameters are fine-tuned at each layer to attain high-level features to generalize and represent the desired features for recognizing the unseen images. This has allowed for the development of groundbreaking applications in sociable robotics, medical treatment, driver fatigue surveillance, student awareness estimation and other human-computer interaction systems. However, there are still challenges that need to be addressed such as cross-cultural variability, nonverbal cues, high intra-class variation, high inter-class similarities and variations in head pose, illumination, and occlusions. In this paper, we have improved the performance of emotion recognition models, and have applied them in real world situations. We took several approaches from recent publications to improve accuracy, including transfer learning, data augmentation, class weighting, adding auxiliary data, and ensembling. We also applied our results to develop a web app to run our models on-device.

II. RELATED WORKS

In the emotional analysis, facial emotion recognition and classification has been considered as a challenging task. In recent years, many authors have proposed and developed various deep learning and machine learning (ML) models for emotion recognition tasks. FER2013 was designed by Goodfellow[1] as a Kaggle competition to promote researchers to develop better FER systems. The winner, Yichuan Tang, achieved 71.2% accuracy by using the primal objective of an SVM as the loss function for training and additionally used the L2-SVM loss function. This was a new development at the time and gave great results on the contest dataset. A recent survey paper on FER by S. Li and W.Deng sheds light on the current state of deep-learning-based approaches to FER. Another paper by Pramerdorfer and Kampel [2] describes the approaches taken by six current state-of-the-art papers and ensembles their networks to achieve 75.2% test accuracy on FER2013. The highest accuracy achieves so far on FER2013 dataset is 75.8% by Amil Khanzada et. al.[11], which is, to our knowledge, the highest reported in any published journal paper. Among the six papers, Zhang et al. achieved the highest accuracy of 75.1% by employing auxiliary data and additional features: a vector of HoG features were computed from face patches and processed by the first FC layer of the CNN (early fusion). They also employed facial landmark registration, suggesting its benefits even in challenging conditions (facial landmark extraction is inaccurate for about 15% of images in the FER dataset) [5]. The paper with the second highest accuracy by Kim et al. utilized face registration, data augmentation, additional features, and ensembling [6]. From our graduate community at Stanford, we also found reports from recent CS229 and CS230 projects on FER useful as reference [7,8].

III. DATASETS

Facial Emotions Recognition is a well-studied field with numerous available datasets. We used FER2013 as our main dataset and improved accuracy on its test set and also used CK+ and JAFFE as auxiliary datasets.

FER2013 Dataset

The images in the FER2013 dataset were collected from a variety of sources, including the internet, and they were pre-processed and aligned to a standard size of 48x48. The dataset consists of 35,887 grayscale images of faces, each labeled with one of seven possible emotion categories: angry, disgusted, fearful, happy, sad, surprised, and neutral and is divided into three sets: a training set of 28,709 images, a validation set of 3,589 images, and a test set of 3,589 images.

The dataset is considered to be challenging due to its diverse set of images with variations in pose, lighting, and expression intensity, achieving only 65±5% human-level accuracy and the highest performing published works achieving 75.8% test accuracy. FER2013 is, however, not a balanced dataset, as it contains only 547 images labeled ‘disgust’, which is 91% less than the average distribution of other 6 emotions.

Fig.1.1 Images from FER2013 dataset

JAFFE Dataset

The Japanese Female Facial Expression (JAFFE) is a relatively small dataset containing 213 256x256 grayscale images of 10 Japanese women. It covers 7 facial expressions, 6 basic and 1 neutral with several images for each expression. The six basic facial expressions covered are anger, fear, sadness, happiness, surprise, and disgust.

Extended Cohn-Kanade Dataset

The extended Cohn-Kanade dataset (CK+) was compiled by taking snapshots of each of the 7 expressions posed in the 593 video sequences of the original dataset. The compiled dataset contains 981 48x48 grayscale images of subjects ranging from 18 to 50 years of age with variety of genders and heritage. The images are classified into emotions like anger, disgust, contempt, sadness, fear, surprise and happiness.

IV. MODELS

Baseline Model

In order to better understand the problem, we decided to first try to tackle this problem from scratch, building a deep CNN using eight 3x3x32 same-padding, ReLU filters, interleaved with four 2x2 MaxPool layers, and completed with a FC layer and softmax layer. We also added batch norm and 50% dropout layers to address high variance and improve our accuracy.

Five-Layer Model

One of the highest accuracy papers reported 75.8% accuracy, not using auxiliary training data or facial landmark registration. The authors achieved these results by studying six other papers and ensembling their networks. Because of the simplicity of the network, we decided to replicate their exercise of reproducing the results.

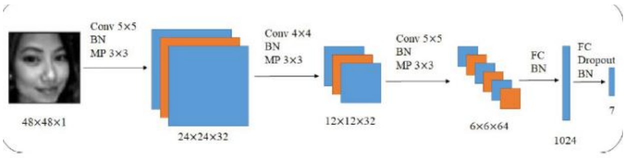


Fig.2.1. Five Layer Model Architecture

This model consists of three stages of convolutional and max-pooling layers, followed by an FC layer of size 1024

and a softmax output layer. The max-pooling layers use kernels of size 3x3 and stride 2. ReLU was utilized as the activation function. To improve performance, we also added batch norm at every layer and 30% dropout after the last FC layer. To fine tune the model, we trained it for 100 epochs, optimizing the cross-entropy loss using stochastic gradient descent with a momentum of 0.9. The initial learning rate, batch size, and weight decay are fixed at 0.1, 64, and 0.0001, respectively. The learning rate is halved if the validation accuracy does not improve for 10 epochs.

Transfer Learning

Since the FER2013 dataset is quite small and unbalanced, we found that utilizing transfer learning significantly boosted the accuracy of our model. We explored transfer learning, using the Keras VGG-Face library and each of ResNet50, SeNet50 and InceptionV3 as our pre-trained models. To match the input requirements of these new networks which expected RGB images of no smaller than 197x197, we resized and recolored the 48x48 grayscale images in FER2013 during training time.

Fine Tuning ResNet50

ResNet50 is the first pre-trained model we explored. ResNet50 is a deep residual network with 50 layers. It is defined in Keras with 175 layers. We replaced the original output layer with two FC layers of sizes 4,096 and 1,024 respectively and a softmax output layer of 7 emotion classes. We froze the first 170 layers in ResNet, and kept the rest of the network trainable. We used SGD as our optimizer with a learning rate of 0.01 and a batch size of 32. After training for 122 epochs using SGD with a 0.01 learning rate and a batch size of 128, we achieved 73.2% accuracy on the test set. We also tried to freeze the entire pre-trained network and only train the FC layers and output layer, but the model failed to fit onto the training set in the first 20 epochs despite our many attempts to adjust Hyperparameters.

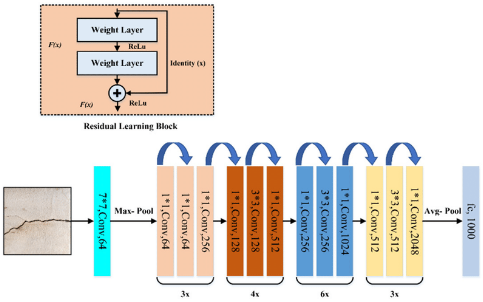


Fig.2.2. ResNet50 Architecture

Fine Tuning SeNet50

SeNet50 is another pre-trained model we explored. It is a deep residual network with 50 layers. SeNet50 has a similarstructure with ResNet50, so we didn’t spend too much time tuning this model. We trained the network on the set of parameters we used for ResNet50 and achieved 72.5% accuracy on the test set.

Fine Tuning InceptionV3

Although much shallower than ResNet50 and SeNet50 with only 16 layers, VGG16 is more complex and has many more parameters. We kept all pre-trained layers frozen and added two FC layers of size 4096 and 1024 respectively with 50% dropout. After 100 epochs of training with the Adam optimizer, we achieved an accuracy of 70.2% on the test set classified into emotions like anger, disgust, contempt, sadness, fear, surprise and happiness.

V. METHODS

Auxiliary Data & Data Preparation

Although several FER datasets are available online, they vary widely in image size, color, and format, as well as labeling and directory structure. During training, we loaded images in batches of 64 from disk (to avoid memory overflow) and utilized Keras data generators to automatically resize and format the images.

Data Augmentation

We researched and experimented with commonly used techniques in existing FER papers and achieved our best results with horizontal mirroring, ±10 degree rotations, ±10% image zooms, and ±10% horizontal/vertical shifting.

Class Weighting

To alleviate the class imbalance problem, we applied class weighting inversely proportional to the number of samples. For the disgust class, we were able to drop the misclassification rate from 61% to 34%.

SMOTE

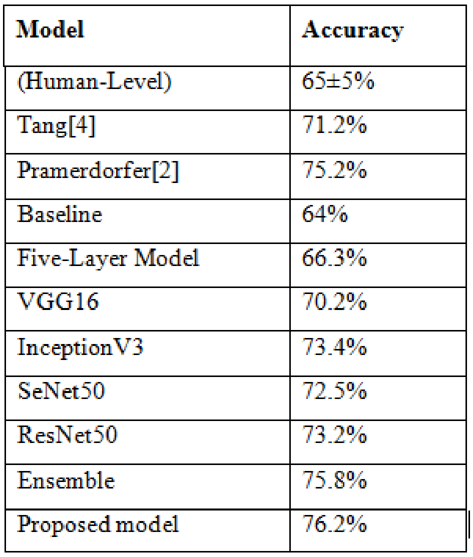
The Synthetic Minority Over-sampling Technique (SMOTE) involves oversampling minority classes and under sampling majority classes to get the best results. Although utilizing SMOTE resulted in a perfectly balanced training dataset, our models quickly overfit to the training dataset and we decided not to experiment further.

Ensembling

We performed ensembling with soft voting of seven models to significantly improve our highest test accuracy from 75.8% to 76.2%.

VI. EXPERIMENTAL RESULTS

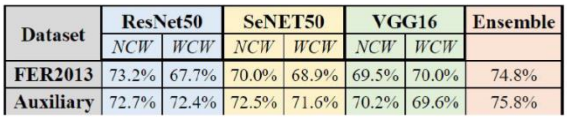
Most of the publications which achieved state-of-the-art accuracies on FER2013 utilized auxiliary training data.



Model	Accuracy
(Human-Level)	65±5%
Tang[4]	71.2%
Pramerdorfer[2]	75.2%
Baseline	64%
Five-Layer Model	66.3%
VGG16	70.2%
InceptionV3	73.4%
SeNet50	72.5%
ResNet50	73.2%
Ensemble	75.8%
Proposed model	76.2%

Fig.3.1Results on FER2013

The table below demonstrates our accuracy gains from employing auxiliary data with care taken to avoid dataset bias. It also depicts our success in implementing class weighting, which significantly increased accuracies on frequently misclassified emotions. Ensembling several models with and without class weights improved the overall accuracy.



Dataset	ResNet50		SeNET50		VGG16		Ensemble
	NCW	*WCW*	*NCW*	*WCW*	*NCW*	*WCW*	
FER2013	73.2%	67.7%	70.0%	68.9%	69.5%	70.0%	74.8%
Auxiliary	72.7%	72.4%	72.5%	71.6%	70.2%	69.6%	75.8%

Fig.3.2. Accuracies for different methods with and without auxiliary data.

Given the high complexity of transfer learning models and relatively small size of our datasets, we also experienced overfitting while training. Although we added 50% dropout for the last three layers, our ResNet50 transfer learning model quickly overfit

to the training set with dev accuracy starting to flatten after only 30 epochs.
It is worth mentioning that because our models exceeded human-level accuracy, error analysis was particularly challenging for some misclassifications, such as the fear image discussed prior. Additionally, because emotions are highly subjective, Bayes error is high and it is often the case that an image can have multiple interpretations.

VII. CONCLUSION

In addition to Deep CNNs, we also looked at pre-trained networks built on InceptionV3, SeNet50, and ResNet50. We used data augmentation, class weights, and supplementary datasets to reduce the FER2013's innate class imbalance. A maximum accuracy of 76.2%, to the best of our knowledge, was obtained on FER2013 dataset by creating an ensemble of four models described in previous sections, fine-tuning our architecture and applying a voting classifier in the end. Moreover, we discovered that network interpretability taught our models to concentrate on pertinent facial cues for emotion recognition. Additionally, we demonstrated that facial emotion recognition models could be applied in the real world by developing flask-based web application with quick real-time detection and low memory, disc, and computational needs.

VIII. FUTURE WORK

We intend to use facial landmark detection and alignment, attention CNNs, and retrain our network by occluding facial elements unrelated to emotion recognition in order to increase the accuracy of our models. We also want to use more auxiliary data (especially AffectNet, which has over a million tagged photos), and we want to use techniques like ADASYN to balance our training dataset. There are still numerous issues that need to be resolved, such as how to handle hidden facial expressions, cultural differences, and privacy concerns. By using various data augmentation approaches to address concerns with fluctuating camera brightness and angle, we would like to increase the robustness and accuracy of our web app model. Also, one might utilize a quantized face detection model that runs on the GPU/TPU for improved deployment outcomes.

IV. REFERENCES

[1] S. Li and W. Deng, “Deep facial expression recognition: A survey,” arXiv preprint: 1804.08348, 2018.

[2] Pramerdorfer, C., Kampel, M.: Facial expression recognition using convolutional neural networks: state of the art. Preprint arXiv:1612.02903v1, 2016.

[3] I. J. Goodfellow et al., “Challenges in representation learning: A report on three machine learning contests,”

[4] Y. Tang, “Deep Learning using Support Vector Machines,” in International Conference on Machine Learning (ICML) Workshops, 2013.

[5] Z. Zhang, P. Luo, C.-C. Loy, and X. Tang, “Learning Social Relation Traits from Face Images,”

[6] B.-K. Kim, S.-Y. Dong, J. Roh, G. Kim, and S.-Y. Lee, “Fusing Aligned and Non-Aligned Face Information for Automatic Affect Recognition in the Wild: A Deep Learning Approach,”

[7] Quinn M., Sivesind G., and Reis G., “Real-time Emotion Recognition From Facial Expressions”, 2017

[8] Wang J., and Mbuthia M., “FaceNet: Facial Expression Recognition Based on Deep Convolutional Neural Network”, 2018

[9] Facial Expression Recognition with Deep Learning by Amil Khanzada et al. arXiv:2004.11823

[10] Virtual facial expression recognition using deep CNN with ensemble learning, doi.org/10.1007/s12652-020-02866-3

[11] Deep Learning based Framework for Emotion Recognition using Facial Expression,doi.org/10.51846/vol5iss3pp51-57

[12]https://datarepository.wolframcloud.com/resources/FER-2013

[13]https://www.kaggle.com/datasets/msambare/fer2013

[14]https://doi.org/10.1109/BIOSMART.2019.8734245

[15]https://www.mdpi.com/1088298

[16]https://doi.org/10.48550/arXiv.1612.02903

[17]https://doi.org/10.48550/arXiv.1307.0414

REFERENCES

[1] https://doi.org/10.1007/s12652-020-02866-3

[2] https://doi.org/10.51846/vol5iss3pp51-57

[3] https://datarepository.wolframcloud.com/resources/FER-2013

[4] https://www.kaggle.com/datasets/msambare/fer2013

[5] https://doi.org/10.1109/BIOSMART.2019.8734245

[6] https://www.mdpi.com/1088298

[7] https://doi.org/10.48550/arXiv.1612.02903

[8] https://doi.org/10.48550/arXiv.1307.0414

[9] https://zenodo.org/record/3451524#.X2bg0GgzZPY

[10] http://dx.doi.org/10.28979/jarnas.1056664

[11] https://doi.org/10.1109/CVPR.2017.277

[12] https://github.com/NJNischal/Facial-Expression-Recognition-with-CNNs

[13] https://github.com/akmadan/Emotion_Detection_CNN

[14] https://github.com/krishnaik06/Flask-Web-Framework

[15] https://machinelearningmastery.com/how-to-create-a-random-split-cross-validation-and-bagging-ensemble-for-deep-learning-in-keras/

[16] https://doi.org/10.1142/S0218001419400159

[17] https://github.com/ashishpatel26/Facial-Expression-Recognization-using-JAFFE

[18] https://github.com/bmanczak/BEP

[19] https://github.com/alicex2020/Deep-Learning-Lie-Detection.git

[20] https://www.kirupa.com/html5/accessing_your_webcam_in_html5.htm

[21] https://www.freecodecamp.org/news/how-to-make-your-first-javascript-chart/

[22] https://www.w3schools.com/js/js_graphics_chartjs.asp

APPENDIX

main.ipynb

```
!ngrok authtoken
2K5KnWoHDYHraEQeC4E6fGzL2io_qpUBzverAjQq7Mu2QD6T


from keras.models import load_model
from time import sleep
from keras_preprocessing.image import img_to_array
from keras.preprocessing import image
import numpy as np
from flask import Flask, render_template, Response
import cv2


app = Flask(__name__)

from flask_ngrok import run_with_ngrok
run_with_ngrok(app)


camera = cv2.VideoCapture(0)


def gen_frames():

detector=cv2.CascadeClassifier('haarcascade_frontalface_d
efault.xml')
    classifier =load_model(r'model.h5')
    emotion_labels =
['Angry','Disgust','Fear','Happy','Neutral', 'Sad',
'Surprise']

    while True:
        success, frame = camera.read()  # read the camera
frame
        if not success:
            break
        else:
            labels = []
            gray = cv2.cvtColor(frame,cv2.COLOR_BGR2GRAY)
            faces = detector.detectMultiScale(gray,1.1,7)

            for (x,y,w,h) in faces:
```

```
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_S
IMPLEX,1,(0,255,0),2)
                else:
                    cv2.putText(frame,'No
Faces',(30,80),cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)

            cv2.imshow('Emotion Detector',frame)
            ret, buffer = cv2.imencode('.jpg', frame)
            frame = buffer.tobytes()
            yield (b'--frame\r\n'
                   b'Content-Type: image/jpeg\r\n\r\n' +
frame + b'\r\n')

@app.route('/')
def index():
    return render_template('index.html')
@app.route('/video_feed')
def video_feed():
    return Response(gen_frames(), mimetype='multipart/x-
mixed-replace; boundary=frame')
if __name__=='__main__':
    app.run()
```

HTML Template

```
<!DOCTYPE html>
<html>


  <body>
    <div class="container">
```

```
        <div class="row">
            <div class="col-lg-8  offset-lg-2">
                <h3 class="mt-5">Live Streaming</h3>
                <img src="{{ url_for('video_feed') }}"
     width="50%">
            </div>
        </div>
    </div>
    </body>


</html>
```

model_train.ipynb (baseline model only)

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
import seaborn as sns
import os
from keras.utils import load_img, img_to_array
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Dense,Input,Dropout,GlobalAverag
ePooling2D,Flatten,Conv2D,BatchNormalization,Activation,M
axPooling2D
from keras.models import Model,Sequential
from keras.optimizers import RMSprop,SGD,Adam
from keras.callbacks import ModelCheckpoint, EarlyStoppin
g, ReduceLROnPlateau


!unzip '/content/drive/MyDrive/TrainingSet.zip' -
d '/content/'

picture_size = 48
folder_path='/content/images/'
expression = 'disgust'
plt.figure(figsize= (12,12))
for i in range(1, 10, 1):
    plt.subplot(3,3,i)
    img = load_img(folder_path+"train/"+expression+"/"+
                  os.listdir(folder_path + "train/" + exp
ression)[i], target_size=(picture_size, picture_size))
    plt.imshow(img)
plt.show()
```

```
emotion_labels = ['Angry','Disgust','Fear','Happy','Neutral', 'Sad', 'Surprise']

batch_size  = 64
datagen_train  = ImageDataGenerator()
datagen_val = ImageDataGenerator()

batch_size  = 64
datagen_train  = ImageDataGenerator()
datagen_val = ImageDataGenerator()

train_set = datagen_train.flow_from_directory(folder_path+"train",target_size = (picture_size,picture_size), color_mode = "grayscale",batch_size=batch_size, class_mode='categorical', shuffle=True)

test_set = datagen_val.flow_from_directory(folder_path+"validation", target_size = (picture_size,picture_size), color_mode = "grayscale", batch_size=batch_size,class_mode='categorical', shuffle=False)



from tensorflow.keras.regularizers import l2


model = Sequential()
num_features = 64
num_labels = 7

model.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu', input_shape=(48, 48, 1), data_format='channels_last', kernel_regularizer=l2(0.01)))
model.add(Conv2D(num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Conv2D(2*num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(2*num_features, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))
model.add(Conv2D(2*2*num_features, kernel_size=(3, 3), ac
tivation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(2*2*num_features, kernel_size=(3, 3), ac
tivation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Conv2D(2*2*2*num_features, kernel_size=(3, 3),
activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(Conv2D(2*2*2*num_features, kernel_size=(3, 3),
activation='relu', padding='same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2), strides=(2, 2)))
model.add(Dropout(0.5))

model.add(Flatten())

model.add(Dense(2*2*2*num_features, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2*2*num_features, activation='relu'))
model.add(Dropout(0.4))
model.add(Dense(2*num_features, activation='relu'))
model.add(Dropout(0.5))

model.add(Dense(num_labels, activation='softmax'))
model.compile(loss='categorical_crossentropy',
              optimizer=Adam(lr=0.001, beta_1=0.9, beta_2
=0.999, epsilon=1e-7),
              metrics=['accuracy'])

history = model.fit_generator(train_set,
steps_per_epoch=train_set.n//train_set.batch_size,
epochs=100,validation_data = test_set,
validation_steps = test_set.n//test_set.batch_size,
shuffle=True )
```