Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) based Application Programming Interface (API) to predict Emotion over web

A PROJECT REPORT

Submitted by

PRIYADHARSHINI.M [REGISTER NO:211417104203]

NISHA.M [REGISTER NO:211417104167]

MUPPALA NAVYA SREE [REGISTER NO:211417104155]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2021

BONAFIDE CERTIFICATE

Certified that this project report "**Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) based Application Programming Interface (API) to predict Emotion over web"** is the bonafide work of **"PRIYADHARSHINI.M [REGISTER NO:211417104203] MUPPALA NAVYA SREE [REGISTER NO:211417104155] NISHA.M [REGISTER NO:211417104167]"** who carried out the project work under my supervision.

SIGNATURE

Dr.S. MURUGAVALLI, M.E., Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE
V.SATHIYA
ASSOCIATE PROFESSOR
SUPERVISOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was/ were examined in the Anna University Project Viva-Voce Examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P. CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to extend our heartfelt and sincere thanks to our Directors **Tmt.C.VIJAYARAJESWARI**, **Thiru.C.SAKTHIKUMAR,M.E.,** and **Tmt. SARANYASREE SAKTHIKUMAR B.E., M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of CSE Department, **Dr. S.MURUGAVALLI , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thank my **Project Guide Ms.V.Sathiya** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project

We would like to thank god for showering his blessing on us and we also thank our parents much for their support both emotional and financial over the years.

NAME OF THE STUDENTS

M.PRIYADHARSHINI

MUPPALA NAVYA SHREE

M.NISHA

ABSTRACT

Emotion Detection from facial expressions using AI can be a viable alternative to automatically measure consumer's engagement with their content and brands. Machine emotional intelligence is a burgeoning frontier that could have huge consequences in not only advertising, but in new startups, healthcare, wearables, education, and more. Human emotion plays an important role in the interpersonal relationship. Automatic recognition of emotion has been an active research topic from early eras and Efficiency has been a challenge. In this project we build an ***Application Programming Interface (API) to predict facial Expression*** **powered by Amazon** ***EC2 Linux instances.*** **Algorithms and models like Long Short-Term Memory** ***(LSTM) and*** **Convolutional Neural Networks** ***(CNN)*** are used for training and performance evaluation. The potential application of this Application includes Facial Expression based feedback in OTG platform's, Facial Expression detection in online interviews, Facial Expression Recognition for Security, IoT Devices, Connected users and streaming Dashboards. Effectiveness of the API is measured through several defeats and experiments by backpropagation of errors. The main goal of this project is to produce an enhanced performance level with greater accuracy.

Key words: ***Deep Learning, Cloud computing (Flask), web development, image processing, AI, port sharing, SSH, Face recognition, Emotion detection, Application Programming Interface(API), Long Short term memory(LSTM), Convolutional Neural networks (CNN), Feedback loop, Model training, Data cleaning.***

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

1.API-application programming interface

2.LSTM-long short term memory

3.SSH-secure shell

4.OTG-on the go

5.CNN-convolutional neural network

6.FCN-fully convolutional network

7.TCP-transmission control protocol

8.DLIB-digital library

9.C MAKE-cross platform

CHAPTER- I

INTRODUCTION

The development and usage of computer systems, software and networks are growing tremendously. These systems have an important role in our everyday life and they make human life much easier. Emotion plays a vital role in determining the thoughts, behavior and feeling of a human. An emotion recognition system can be built by utilizing the benefits of deep learning and different applications such as feedback analysis, face unlocking etc. can be implemented with good accuracy. Facial emotion recognition system assumes a lot of importance in this era since it can capture the human behavior, feelings, intentions etc. The conventional methods have limited speed and have less accuracy while facial emotion recognition systems using deep learning has proved to be the better one.

In this project, we use an Amazon EC2 Linux instance to create an Application Programming Interface (API) that predicts facial expression. Training and performance assessment are carried out using algorithms and models such as Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNN). A Fully Convolutional Neural Network based classifier is used to predict the facial emotions of the person in the image. An FCN (Fully Convolutional Network) is a network composed of only convolutional layers, batch norms and non-linearities, i.e., it has no fully Connected (or Dense) layers. The figure below shows how a convolutional layer applies. Whether a face exists in the image or not is determined by a Viola Jones Detector. Facial Expression Based Feedback in OTG Platforms, Facial Expression Detection in Online Interviews, Facial Expression Recognition for Security, IoT Devices, Connected Users, and Streaming Dashboards are some of the potential applications for this application. Backpropagation of errors is used to assess the API's effectiveness after multiple defeats and experiments.

1.1 Overview

The prime objective of this project is to produce an enhanced performance level with greater accuracy reducing complexity . In the proposed system, we built an *Application Programming Interface (API) to predict facial Expression* powered by Amazon *EC2 Linux instances.* Algorithms and models like Long Short-Term Memory *(LSTM) and* Convolutional Neural Networks *(CNN)* are used for training and performance evaluation. The potential application of this Application includes Facial Expression based feedback in OTG platform's, Facial Expression detection in online interviews , Facial Expression Recognition for Security, IoT Devices, Connected users and streaming Dashboards. Effectiveness of the API is measured through several defeats and experiments by backpropagation of error's.

1.2 Problem definition

Human facial expressions can be easily classified into 7 basic emotions: happy, sad, surprise, fear, anger, disgust, and neutral. Our facial emotions are expressed through activation of specific sets of facial muscles. These sometimes subtle, yet complex, signals in an expression often contain an abundant amount of information about our state of mind. Through facial emotion recognition, we are able to measure the effects that content and services have on the audience/users through an easy and low-cost procedure. For example, retailers may use these metrics to evaluate customer interest. Healthcare providers can provide better service by using additional information about patients' emotional state during treatment. Entertainment producers can monitor audience engagement in events to consistently create desired content. Humans are well-trained in reading the emotions of others, in fact, at just 14 months old, babies can already tell the difference between happy and sad. But can computers do a better job than us in accessing emotional states? To answer the question, We designed a deep learning neural network that gives machines the ability

to make inferences about our emotional states to make this process fast and quick and efficient and to get high accuracy we develop an dedicated API to detect facial emotion over cloud.

CHAPTER- II

LITERATURE SURVEY

[1]. Tzuu-Hseng S. Li , Ping-Huan Kuo , Ting-Nan Tsai , Po-Chien Luan**.**journal, IEEE

for "CNN and LSTM Based Facial Expression Analysis Model for a Humanoid Robot"

methodology of Convolutional neural network, long short-term memory, transfer learning, facial expression analysis.In this paper, the robot is equipped with a camera to capture users' facial images, and it uses this system to recognize users' emotions and responds appropriately. The emotion recognition system, based on a deep neural network, learns six basic emotions: happiness, anger, disgust, fear, sadness, and surprise. The proposed model combines CNN and LSTM and exploits the advantages of both CNN and RNN. Leave-one-out cross-validation indicates that the model performance is improved significantly. The system is applied to a humanoid robot to demonstrate its practicability for improving the HRI.

[2]. Berat A. Erol, Abhijit Majumdar,Patrick Benavidez ,Paul Rad, Kim-Kwang Raymond Choo and Mo Jamshidi.journal,IEEE, for **"**Toward Artificial Emotional Intelligence for Cooperative Social Human– Machine Interaction". Methodology Assistive robotics, human–machine interactions, humanoid robot, Internet of robotic things, smart home, supervisory control. The paper presents an interactive heterogeneous robotic system with several components working together for closing the loop by detecting human emotions from facial images. The ultimate goal was designing a system of systems that fulfills user's requests by actively interacting with them, while other system components were gathering and sharing the data.

[3].D Y Liliana**,**journal,top science,for “Emotion recognition from facial expression using deep convolutional neural network” methodology of Facial expression recognition, action unit, deep learning, convolutional neural network.In this paper[3] ,the author proposed Convolutional Neural Network architecture for facial expression recognition. There are 8 classes of facial expression that are tried to recognize.Using the CK+ database we trained using different training data size and the result is the mean square error declines as the number of training data increases and it is concluded that the mean square error declines as the training data grows and the performance of the system reaches 92.81% of the accuracy rate.

[4]. Md. Forhad Ali,Nakib Aman Turzo, Mehenag Khatun , coference paper, for **“** Facial Emotion Detection Using Neural Network,coference paper” methodology of Feature Extraction, Neural Network, Emotion Recognition, Emotion Detection, Facial Recognition. With CNN, it was possible to identify emotions, type of emotion in the real image. To delineate the result and procedures more visually and this has also introduced decision tree techniques which helps to decide which emotions percentage is high and which emotions percentage is low.

[5]. Zixing Zhang, Fabien Ringeval, Fabien Ringeval, Eduardo Coutinho, Erik Marchi and Björn Schüller, journal,IEEE, “Enhanced semi-supervised learning for multimodal emotion recognition.”methodology of Semi-Supervised Learning (SSL) technique.

This system delivers a strong performance in the classification of high/low emotional arousal (UAR = 76.5%), and significantly outperforms traditional SSL methods by at least 5.0% (absolute gain).

[6]. Y. Fan, X. Lu, D. Li, and Y. Liu.journal,IEEE,for "Video-based emotion recognition using CNN-RNN and C3D hybrid networks" methodology of CNN-RNN and C3D hybrid networks.This paper , a video-based emotion recognition system submitted to the EmotiW 2016 Challenge. The core module of this system is a hybrid network that combines recurrent neural networks (RNN) and 3D convolutional networks (C3D) in a late-fusion fashion. RNN and C3D encode appearance and motion information in different ways. Specifically, RNN takes appearance features extracted by convolutional neural networks (CNN) over individual video frames as input and encodes motion later, while C3D models appearance and motion of video simultaneously. Combined with an audio module, our system achieved a recognition accuracy of 59.02% without using any additional emotion-labeled video clips in the training set, compared to 53.8% of the winner of EmotiW 2015. Extensive experiments show that combining RNN and C3D together can improve video-based emotion recognition noticeably.

[7]Tawsin Uddin Ahmed,Sazzad Hossain, Mohammad Shahadat ,Raihan.journal science for "Facial Expression Recognition using Convolutional Neural Network with Data Augmentation" mrthodology of Convolutional neural network, data augmentation, validation accuracy, emotion detection.The objective of this paper is to develop a facial expression recognition system based on convolutional neural network with data augmentation. This approach enables classifying seven basic emotions consisting of angry, disgust, fear, happy, neutral, sad and surprise from image data. Convolutional neural networks with data augmentation leads to higher validation accuracy than the other existing models (which is 96.24%) as well as helps to overcome their limitations.

CHAPTER -III

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In the existing system, the robot is made with a camera to capture users' facial images, and it uses this system to recognize users' emotions and respond appropriately. First, a convolutional neural network (CNN) is used to extract visual features by learning on a large number of static images. Second, a long short-term memory (LSTM) recurrent neural network is used to determine the relationship between the transformation of facial expressions in image sequences. Third, CNN and LSTM are combined to exploit their advantages in the proposed model. The performance of the proposed system is verified through leave-one-out cross-validation and compared with that of other models. The system is applied to a humanoid robot to demonstrate its practicability for improving the HRI. It involves a complex process

and since the robot is made the cost of the project is also high and hence we came up with the only software solution which uses API based emotion detector with deep learning technique reducing complexity.

3.2 PROPOSED SYSTEM:

Prime Objective:

The prime objective of this project is to produce an enhanced performance level with greater accuracy reducing complexity.

In the proposed system, we built an *Application Programming Interface (API) to predict facial Expression* powered by Amazon *EC2 Linux instances*. Algorithms and models like Long Short-Term Memory *(LSTM) and* Convolutional Neural Networks *(CNN)* are used for training and performance evaluation. The potential application of this Application includes Facial Expression based feedback in OTG platform's, Facial Expression detection in online interviews , Facial Expression Recognition for Security, IoT Devices, Connected users and streaming Dashboards. Effectiveness of the API is measured through several defeats and experiments by backpropagation of error's.

3.3 REQUIREMENT ANALYSIS AND SPECIFICATIONS

3.3.1 Input Requirements:

Emotion data set, trained ml model, camera to capture facial emotion, API socket between the application interface.

3.3.2 Output Requirements:

facial emotion from API.

3.3.3 Functional requirement:

Trained emotion data model with CNN, LSTM along with API gateway.

3.4 TECHNOLOGY STACK

HARDWARE REQUIREMENTS:

- ➢ Processor : core i5/17/i8
- ➢ RAM : 4GB
- ➢ CPU : 2 x 64-bit 2.8GHZ 8.00 GT/s
- ➢ Disk storage : 500 GB.

3.5 SOFTWARE REQUIREMENTS:

- ➢ Operating system : Windows 7/8/10(64bit) / Unix/ Linux
- ➢ Additional :
    - Cloud Hosting
    - Flask
    - Python
    - Front end (Bootstrap, Html, CSS)

CHAPTER -IV

SYSTEM DESIGN

UML DIAGRAMS:

4.2 USE CASE DIAGRAM:

End users like developer, online service provider, general user can access the emotion API and the given input moves to AWS gateway then then it proceeds to predict emotion. From there it directs to trained model, the model compares the input with emotion dataset and reverts the result back to predict emotion. LSTM feedback loop helps in training the model for increasing its accuracy. Trainer can access tained model and emotion datasets.

use case diagram, should have the following items identified.

- Functionalities to be represented as use case.

- Actors.
- Relationships among the use cases and actors.

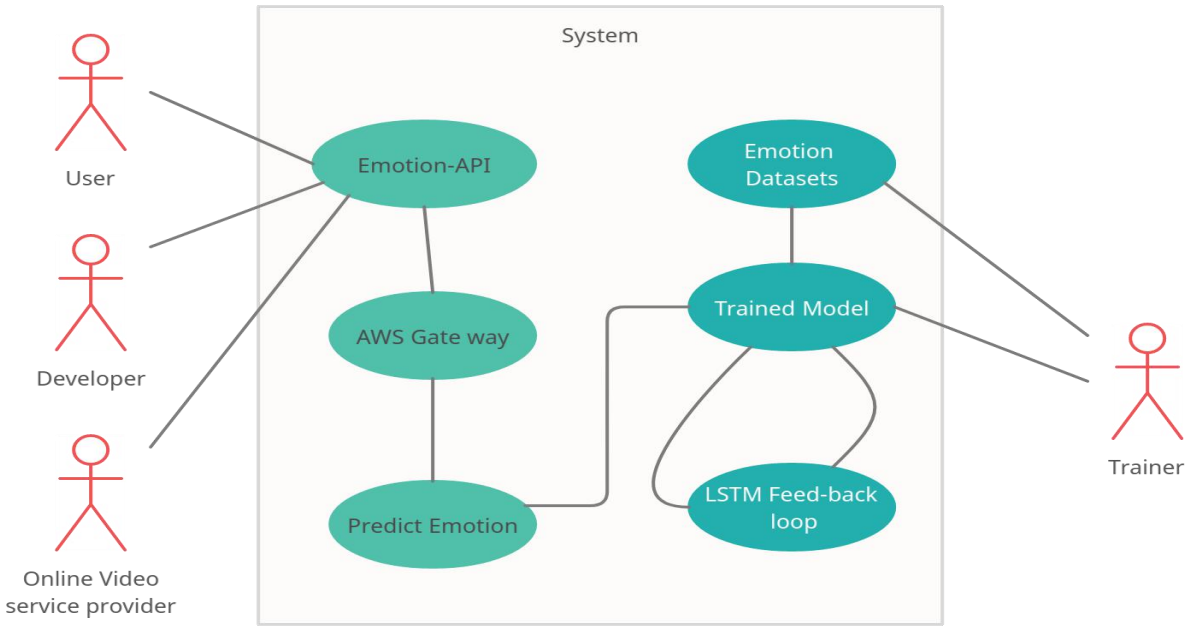


Fig 4.2 Use case diagram

Actor-

- User,
- developer,
- Online video service provider.

Functionalities-

- Emotion prediction and training the model through LSTM Feed-back loop.

4.3 CLASS DIAGRAM:

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely

used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Relationships:

A relationship is a general term covering the specific types of logical connections found on class and object diagrams. UML defines the following relationships:

Dependency

A dependency is a semantic connection between dependent and independent model elements.

Association

An association represents a family of links. A binary association (with two ends) is normally represented as a line.

Multiplicity

This association relationship indicates that (at least) one of the two related classes make reference to the other. This relationship is usually described as "A has a B" (a mother cat has kittens, kittens have a mother cat).

0	No instances (rare)
0..1	No instances, or one instance

1	Exactly one instance
1..1	Exactly one instance
0..*	Zero or more instances
*	Zero or more instances
1..*	One or more instances

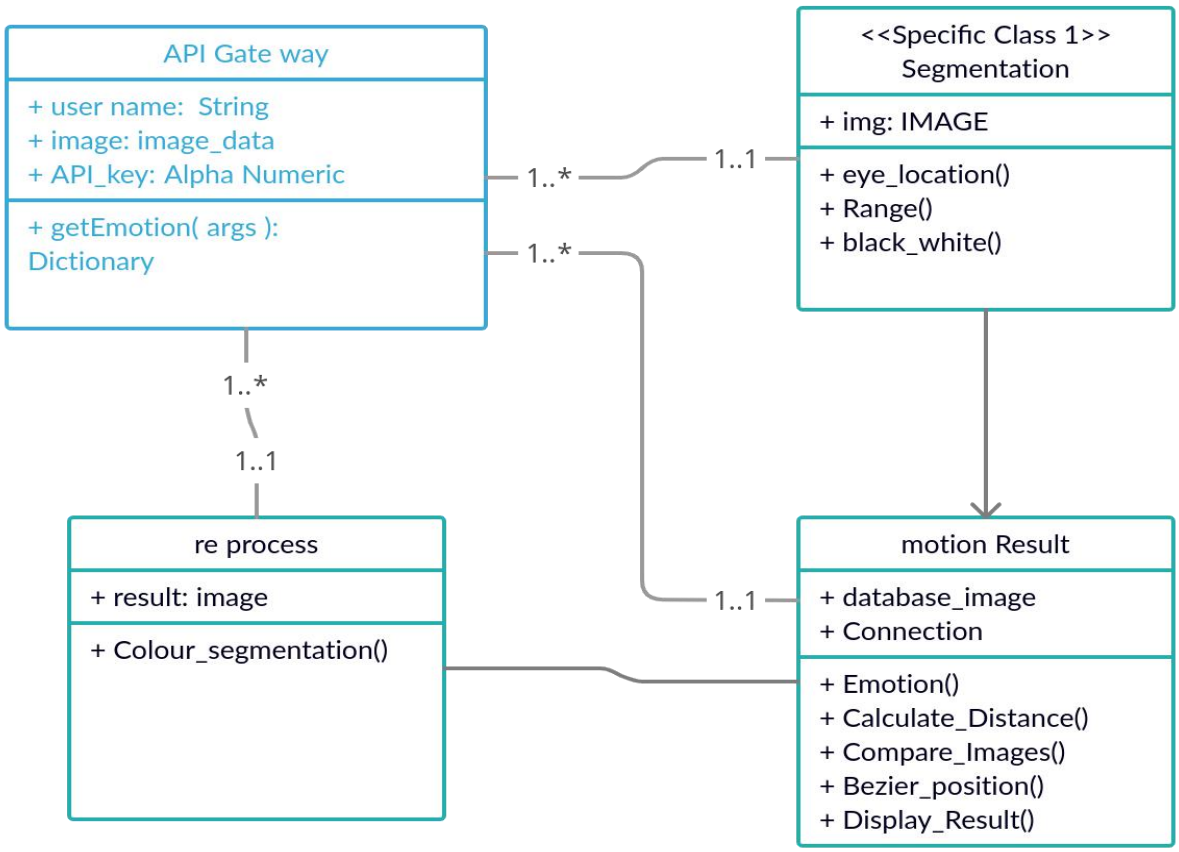


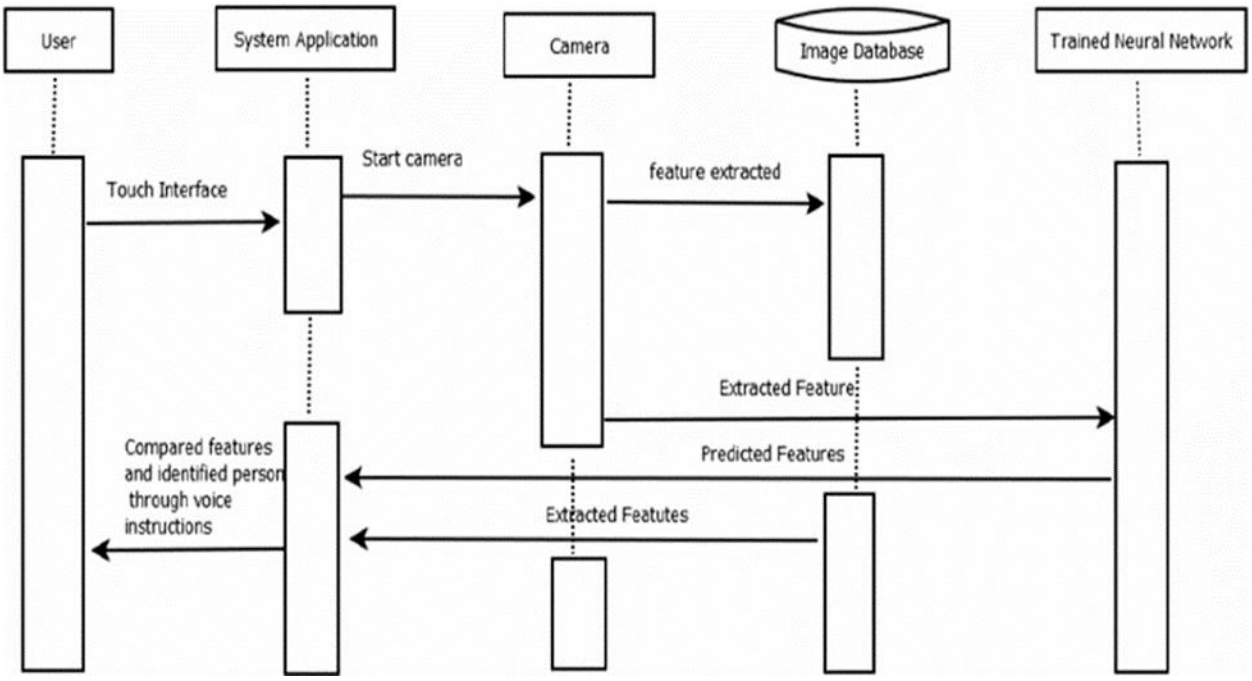
Fig 4.3 Class diagram

Class name: Functions:

- API Gateway - getemotions(args)
- Image(Segmentation) - eye_location(), Range(), black_white()
- Re process - Colour_segmentation()
- Motion result - Emotion(), Calculate_Distance(), Compare_Images(), Beizer_pos(),Display_Result (),Display_Result().

4.4 SEQUENCE DIAGRAM:

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.



Class roles/Participant - User, System Application, Camera, Trained Neural Network.

Lifelines - …… (Symbol)

Message - Touch Interface, Start Camera, feature extracted, predicted features, Compared features and identified person through voice instructions.

CHAPTER- V

SYSTEM ARCHITECTURE

5.1 ARCHITECTURE DIAGRAM:

An architectural diagram is a diagram of a system that is used to abstract the overall outline of the software system and the relationships, constraints, and boundaries between components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap. The below Architecture diagram depicts the overall system implemented in the project.

The Amazon API Gateway requests data from expression API and the image clip involved for feature Extraction steps by HaarCascade Grayscale conversion .

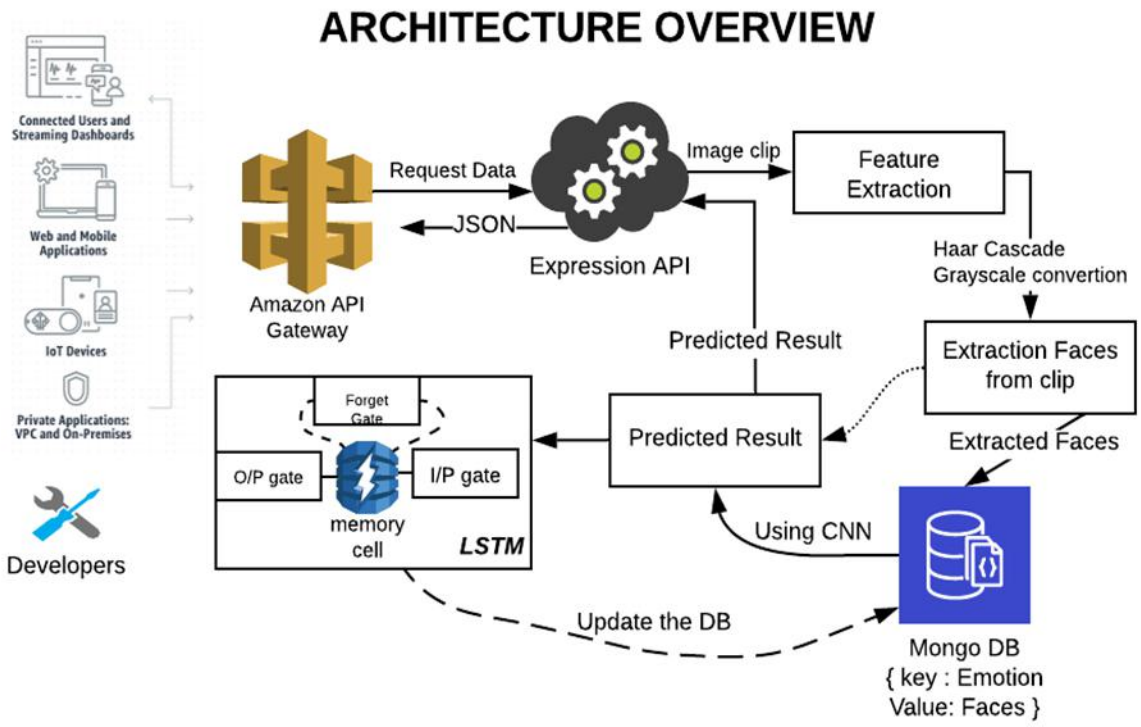


Fig 4.1 Architecture diagram

The Amazon API Gateway requests data from expression API and the image clip involved for feature Extraction steps by HaarCascade Grayscale conversion .

Then the extracted face is saved in MangoDB. Using the deep learning algorithm the results are predicted. The predicted results are sent to LSTM and Expression API.

The predicted results using JSON are sent to the Expression API Gateway. After processing from LSTM the Database is updated in MangoDB with key: emotions and value: face

5.2 MODULE DESCRIPTION-CREATING CNN LSTM MODEL

This project involves three modules,

1. Collecting, preparing and processing image Data

2. Training the model using CNN
3. Using LSTM to build up state, update weights and Back Propagate errors

5.2.1 Collecting, preparing and processing image Data:

a. Collecting Data:

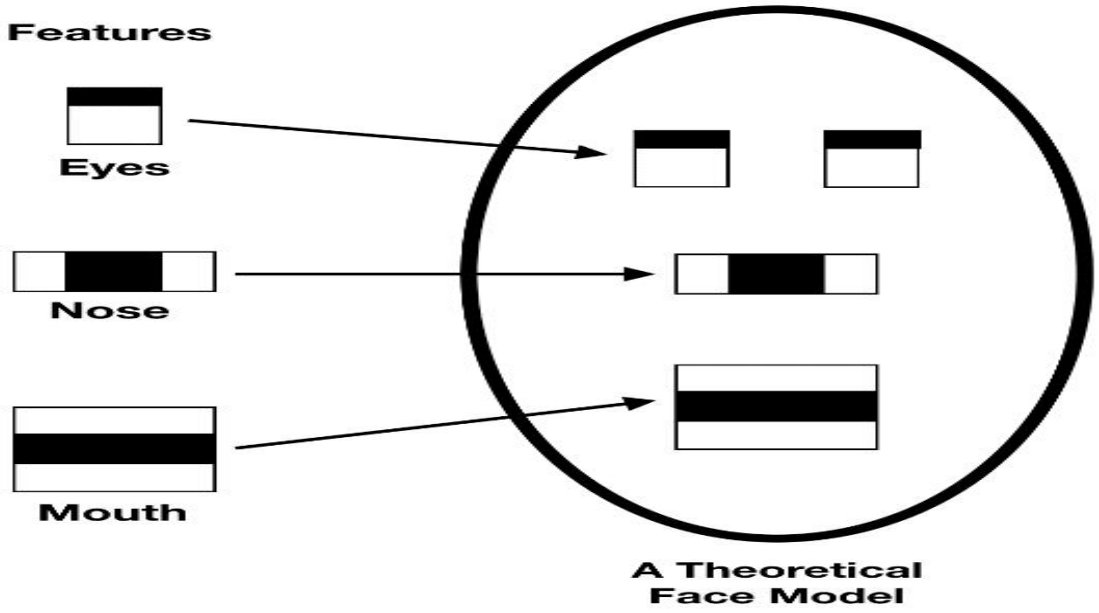
The process of gathering data depends on the type of project we desire to make. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data.We can also use some free data sets which are present on the internet. Kaggle and UCI machine learning repository are the repositories that are used the most for making Deep learning models. In this project we use self-collected data to train the model, additional to that we have also used fer2013 facial dataset to improve our accuracy.

b. Feature Extraction:

Feature extraction is a part of the dimensionality reduction process, in which an initial set of the raw data is divided and reduced to more manageable groups. So when you want to process it will be easier. The most important characteristic of these large data sets is that they have a large number of variables. Feature extraction helps to reduce the amount of redundant data from the data set.In the end, the reduction of the data helps to build the model with less machine's efforts and also increase the speed of learning and generalization steps in the machine learning process.In this project the faces are extracted from clip using grayscale conversion algorithm.

Haar Cascade classifier is an effective object detection approach. This is basically a machine learning based approach where a cascade function is trained from

a lot of images both positive and negative. Based on the training it is then used to detect the objects in the other images.



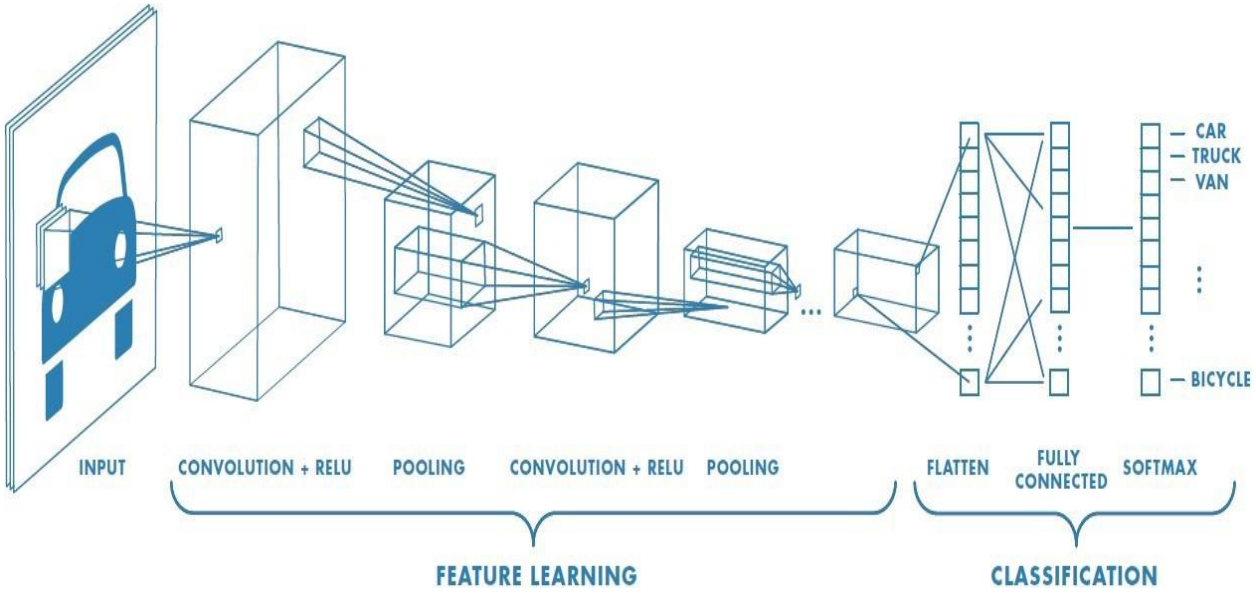
5

5.2.2 CNN based Training

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

In this project we use CNN to train the model based on Facial datasets. The model is later exported so that Further training can be done over cloud along with LSTM.



5.2.3 Exporting the Model to the Cloud

The model generated after CNN is transferred to cloud (AZURE) and configuration is done through SSH.

→ Connecting to server:

$ ssh -X *<user Name>@<IP address>*

$ ssh -X Emotion_API@192.178.43.71

→ Pushing the Model to the Cloud:

$ scp *<file name> <use Name>@<server ip>:\root\<fileName>*

SSH provides a secure channel over an unsecured network by using a client–server architecture, connecting an SSH client application with an SSH server. The protocol specification distinguishes between two major versions, referred to as SSH-1 and SSH-2. The standard TCP port for SSH is 22. SSH is generally used to access Unix-

like operating systems, but it can also be used on Microsoft Windows. Windows 10 uses OpenSSH as its default SSH client and SSH server SSH was designed as a replacement for Telnet and for unsecured remote shell protocols such as the Berkeley rsh and the related rlogin and rexec protocols. Those protocols send information, notably passwords, in plaintext, rendering them susceptible to interception and disclosure using packet analysis. The encryption used by SSH is intended to provide confidentiality and integrity of data over an unsecured network, such as the Internet.

5.2.4 Using Dlib and Cmake for system compatibility:

Dlib is a modern C++ toolkit containing machine learning algorithms and tools for creating complex software in C++ to solve real world problems. It is used in both industry and academia in a wide range of domains including robotics, embedded devices, mobile phones, and large high performance computing environments. Dlib's open source licensing allows you to use it in any application, free of charge.

To follow or participate in the development of dlib subscribe to dlib on github. Also be sure to read the how to contribute page if you intend to submit code to the project.

5.2.5 Connecting the cloud and deploying API using Flask :

Flask-SocketIO gives Flask applications access to low latency bi-directional communications between the clients and the server. The client-side application can use any of the SocketIO official clients libraries in Javascript, C++, Java and Swift, or any compatible client to establish a permanent connection to the server.

SYSTEM IMPLEMENTATION

CHAPTER -VI

SYSTEM IMPLEMENTATION

6.1 CODING-Model Training

```
#create 3 channel numpy array of [num_examples, 3, 48, 48]
#type for np.array dim 3 should be uint8 for conversion to pil image
```

```
train_imgs = []
test_imgs = []

for image in data['pixels']:
    img = image.split()
    img = np.array([float(i) for i in img], dtype=np.uint8)
    img = np.reshape(img, (48,48))
    img = np.array([img]*3)
    train_imgs.append(img)
train_imgs = np.array(train_imgs)
train_labels = [np.array([i]) for i in data['emotion'].to_numpy()]

for image in test_data['pixels']:
    img = image.split()
    img = np.array([float(i) for i in img], dtype=np.uint8)
    img = np.reshape(img, (48,48))
    img = np.array([img]*3)
    test_imgs.append(img)
test_imgs = np.array(test_imgs)
import torch
import torch.nn as nn
from torchvision import transforms
from torch.utils.data import Dataset, DataLoader, TensorDataset
class FaceDataset(Dataset):
"""Custom data set for faces and labels input as numpy array"""
    def __init__(self, samples, labels=None, transform=None):

        self.samples = samples
```

```
        self.labels = labels
        self.transform = transform

    def __len__(self):
        return len(self.samples)

    def __getitem__(self, idx):
        if torch.is_tensor(idx):
            idx = idx.tolist()
        if self.labels == None:
            sample = {'image': torch.from_numpy(self.samples[idx])}
        else:
            sample = {'image': torch.from_numpy(self.samples[idx]), 'label':
torch.from_numpy(self.labels[idx])}

        if self.transform:
            sample['image'] = self.transform(sample['image'])

        return sample

#compose transforms and datasets

transform = transforms.Compose([transforms.ToPILImage(),
transforms.Resize((224, 224)),
                                transforms.RandomHorizontalFlip(p=0.5),
                                transforms.RandomRotation(20),
                                transforms.ToTensor(),
```

```
                    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229,
0.224, 0.225])])

transform1 = transforms.Compose([transforms.ToPILImage(),
                    transforms.ToTensor(),
                    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229,
0.224, 0.225])])

temp_dataset = FaceDataset(train_imgs, train_labels, transform=transform)

split = int(np.round(len(temp_dataset)*0.8))

train_dataset, val_dataset = torch.utils.data.random_split(temp_dataset, [split,
len(temp_dataset) - split])
test_dataset = FaceDataset(test_imgs, transform=transform1)

train_loader = DataLoader(train_dataset, batch_size = 24, shuffle = True)
val_loader = DataLoader(val_dataset, batch_size = 24, shuffle = True)
test_loader = DataLoader(test_dataset, batch_size = 24, shuffle = True)
import matplotlib.pyplot as plt

example = next(iter(train_dataset))

plt.imshow(example['image'])
print(example['label'])

import torch
import torch.nn as nn
```

```
import torchvision.models as models

model = models.resnet18(pretrained=True)
#print(resnet18)

Downloading: "https://download.pytorch.org/models/resnet18-5c106cde.pth" to
/root/.cache/torch/hub/checkpoints/resnet18-5c106cde.pth

for param in model.parameters():
    param.requires_grad = False

model.fc = nn.Sequential(nn.Linear(512, 256),
                         nn.ReLU(),
                         nn.Linear(256, 100),
                         nn.ReLU(),
                         nn.Linear(100, 7))

criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr = 0.01)

#print(model)



#training loop
epochs = 20
device = 'cuda' if torch.cuda.is_available() else 'cpu'
print(f"Training on Device: {device}")
```

```
for epoch in range(epochs):
    running_loss = []

    model.train()
    model.to(device)
    n=0
    for item in train_loader:
        n+=1
        optimizer.zero_grad()

        image = item['image'].to(device)
        label = item['label'].to(device)
        label = label.squeeze()
        output = model(image)
        loss = criterion(output, label)

        loss.backward()
        if n%200 == 0:
            _, pred = torch.max(output.data, 1)
            #print(f"Output: {output}")
            #print(f"Label: {label}")
            #print(f"Pred: {pred}")

        running_loss.append(loss.item())

        optimizer.step()

        #validation loop
```

```
    with torch.no_grad():
        model.eval()
        total = 0
        correct = 0
        for item in val_loader:
            image = item['image'].to(device)
            label = item['label'].to(device)
            label = label.squeeze()
            output = model(image)
            _, pred = torch.max(output.data, 1)

            total+= label.size(0)
            correct += (pred == label).sum().item()

        print(f"Epoch: {epoch}")
        print(f"Training Loss: {sum(running_loss)/len(running_loss)}")
        print(f"Validation Accuracy: {float(correct)/total}")
```

6.2 Coding-Server side Scripting:

I. Create an AWS Container.

```
{
    "containerDefinitions": [
    {
```

```
    "name": "container-using-efs",
    "image": "amazonlinux:2",
    "entryPoint": [
        "sh",
        "-c"
    ],
    "command": [
        "ls -la \mount\efs"
    ],
    "mountPoints": [
        {
        "sourceVolume": "myEfsVolume",
        "containerPath": "\mount\efs",
        "readOnly": true
        }
    ]
  }
  ],
  "volumes": [
  {
    "name": "myEfsVolume",
    "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "rootDirectory": "\path\to\my\data",
"transitEncryption": "ENABLED",
        "transitEncryptionPort": integer,
        "authorizationConfig": {
        "accessPointId": "fsap-1234",
```

```
            "iam": "ENABLED"
            }
        }
    }
    ]
}
```

II. Create an EFS push.

$ yum install amazon-efs-utils

$ systemctl enable --now amazon-ecs-volume-plugin

6.3. coding-Dlib and Cmake script:

I. Configuration file:

```
if(NOT TARGET dlib-shared AND NOT dlib_BINARY_DIR)
  # Compute paths
  get_filename_component(dlib_CMAKE_DIR
"${CMAKE_CURRENT_LIST_FILE}" PATH)
  include("${dlib_CMAKE_DIR}/dlib.cmake")
endif()

set(dlib_LIBRARIES dlib::dlib)
set(dlib_LIBS       dlib::dlib)
set(dlib_INCLUDE_DIRS "/usr/local/include" "")

mark_as_advanced(dlib_LIBRARIES)
mark_as_advanced(dlib_LIBS)
```

```
mark_as_advanced(dlib_INCLUDE_DIRS)

# Mark these variables above as deprecated.
function(__deprecated_var var access)
  if(access STREQUAL "READ_ACCESS")
     message(WARNING "The variable '${var}' is deprecated!  Instead, simply use
target_link_libraries(your_app dlib::dlib).  See
http://dlib.net/examples/CMakeLists.txt.html for an example.")
  endif()
endfunction()
variable_watch(dlib_LIBRARIES __deprecated_var)
variable_watch(dlib_LIBS __deprecated_var)
variable_watch(dlib_INCLUDE_DIRS __deprecated_var)
```

I. CMAKE :

```
# Install script for directory: <directory>
# Set the install prefix
if(NOT DEFINED CMAKE_INSTALL_PREFIX)
  set(CMAKE_INSTALL_PREFIX "/usr/local")
endif()
string(REGEX REPLACE "/$" "" CMAKE_INSTALL_PREFIX
"${CMAKE_INSTALL_PREFIX}")

# Set the install configuration name.
if(NOT DEFINED CMAKE_INSTALL_CONFIG_NAME)
  if(BUILD_TYPE)
     string(REGEX REPLACE "^[^A-Za-z0-9_]+" ""
     CMAKE_INSTALL_CONFIG_NAME "${BUILD_TYPE}")
```

```
  else()
      set(CMAKE_INSTALL_CONFIG_NAME "Release")
  endif()
  message(STATUS "Install configuration:
\"${CMAKE_INSTALL_CONFIG_NAME}\"")
endif()

# Set the component getting installed.
if(NOT CMAKE_INSTALL_COMPONENT)
  if(COMPONENT)
      message(STATUS "Install component: \"${COMPONENT}\"")
      set(CMAKE_INSTALL_COMPONENT "${COMPONENT}")
  else()
      set(CMAKE_INSTALL_COMPONENT)
  endif()
endif()

# Install shared libraries without execute permission?
if(NOT DEFINED CMAKE_INSTALL_SO_NO_EXE)
  set(CMAKE_INSTALL_SO_NO_EXE "1")
endif()

# Is this installation the result of a cross compile?
if(NOT DEFINED CMAKE_CROSSCOMPILING)
  set(CMAKE_CROSSCOMPILING "FALSE")
endif()

# Set default install directory permissions.
```

```
if(NOT DEFINED CMAKE_OBJDUMP)
  set(CMAKE_OBJDUMP "/usr/bin/objdump")
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR
NOT CMAKE_INSTALL_COMPONENT)
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/lib" TYPE
STATIC_LIBRARY FILES
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/buil
d/dlib/libdlib.a")
endif()
if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR
NOT CMAKE_INSTALL_COMPONENT)
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/include/dlib"
TYPE DIRECTORY FILES
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/dlib/
" FILES_MATCHING REGEX "/[^/]*\\.h$" REGEX "/[^/]*\\.cmake$" REGEX
"/[^/]*\\_tutorial\\.txt$" REGEX "/cassert$" REGEX "/cstring$" REGEX "/fstream$"
REGEX "/iomanip$" REGEX "/iosfwd$" REGEX "/iostream$" REGEX "/istream$"
REGEX "/locale$" REGEX "/ostream$" REGEX "/sstream$" REGEX
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/buil
d/dlib" EXCLUDE)
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR
NOT CMAKE_INSTALL_COMPONENT)
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/include/dlib"
TYPE FILE FILES
```

```
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/config.h")
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR NOT CMAKE_INSTALL_COMPONENT)
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/include/dlib" TYPE FILE FILES
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/revision.h")
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR NOT CMAKE_INSTALL_COMPONENT)
  if(EXISTS "$ENV{DESTDIR}${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib/dlib.cmake")

file(DIFFERENT EXPORT_FILE_CHANGED FILES
      "$ENV{DESTDIR}${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib/dlib.cmake"
      "/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/CMakeFiles/Export/lib/cmake/dlib/dlib.cmake")
      if(EXPORT_FILE_CHANGED)
      file(GLOB OLD_CONFIG_FILES "$ENV{DESTDIR}${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib/dlib-*.cmake")
      if(OLD_CONFIG_FILES)
```

```
      message(STATUS "Old export file
\"$ENV{DESTDIR}${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib/dlib.cmake\"
will be replaced.  Removing files [${OLD_CONFIG_FILES}].")
      file(REMOVE ${OLD_CONFIG_FILES})
      endif()
      endif()
  endif()
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib"
TYPE FILE FILES
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/buil
d/dlib/CMakeFiles/Export/lib/cmake/dlib/dlib.cmake")
  if("${CMAKE_INSTALL_CONFIG_NAME}" MATCHES
"^([Rr][Ee][Ll][Ee][Aa][Ss][Ee])$")
      file(INSTALL DESTINATION
"${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib" TYPE FILE FILES
"/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/buil
d/dlib/CMakeFiles/Export/lib/cmake/dlib/dlib-release.cmake")
  endif()
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR
NOT CMAKE_INSTALL_COMPONENT)



  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/lib/cmake/dlib"
TYPE FILE FILES
```

```
    "/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/config/dlibConfig.cmake"
    "/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/config/dlibConfigVersion.cmake"
    )
endif()

if("x${CMAKE_INSTALL_COMPONENT}x" STREQUAL "xUnspecifiedx" OR NOT CMAKE_INSTALL_COMPONENT)
  file(INSTALL DESTINATION "${CMAKE_INSTALL_PREFIX}/lib/pkgconfig" TYPE FILE FILES "/home/humanz/Desktop/Tcs_handson/Flask_proj/Attendance_system_flask/dlib/build/dlib/dlib-1.pc")
endif()
```

6.4 Coding API with POST Request Flask:

I. Testing API with POST Request:

```
import matplotlib.pyplot as plt
import requests
```

```
import cv2
import os

#Emotion_reply={}
def getEmotionWithImage():
    #img=cv2.imread('./image_and_results/Navya/Angry.jpeg')
    url = 'http://127.0.0.1:5000/Emotion'
    my_img = {'image': open('./image_and_results/Navya/smile.jpeg', 'rb')}
    r = requests.post(url, files=my_img)

    # convert server response into JSON format.
    Emotion_reply=r.json()
print(Emotion_reply)
    plotEmotionWithPercent(Emotion_reply)
    #print(r)

#------------------------Ploting the Emotion----------------------

def plotEmotionWithPercent(Emotion_reply):
    # labels for bars
    tick_label = list(Emotion_reply['emotion'].keys())

    # x-coordinates of left sides of bars
    #left = list(range(1,len(tick_label)+1))

    # heights of bars
    height=[]
    for emval in tick_label:
```

```
        height.append(Emotion_reply['emotion'][emval])

        #import matplotlib.pyplot as plt

        fig, ax = plt.subplots()

        #total = 90000
        langs = tick_label
        langs_users_num = height

        #percent = langs_users_num/total*100

        new_labels = [i+'  {:.2f}%'.format(j) for i, j in zip(langs, langs_users_num)]

        plt.barh(langs, langs_users_num, color='lightskyblue', edgecolor='blue')
        plt.yticks(range(len(langs)), new_labels)
        plt.tight_layout()
    for spine in ax.spines.values():
        spine.set_visible(False)
        ax.set_title('Dominant Emotion - '+Emotion_reply['dominant_emotion'])
        ax.axes.get_xaxis().set_visible(False)
        ax.tick_params(axis="y", left=False)
        plt.show()

    def getEmotionWithVideo():

        file_name=""
        video_capture = cv2.VideoCapture(1) #replace with 0 for first camera
```

```
        while True:
        ret, frame = video_capture.read()
        frame = cv2.resize(frame,(800,500))
        frame = cv2.putText(frame, 'Press \"Q\" to detect Emotion', (50,50),
cv2.FONT_HERSHEY_SIMPLEX,
                    1,(0,0,255), 2, cv2.LINE_AA)
        cv2.imshow("Frame",frame)
        if cv2.waitKey(1) & 0xFF == ord('q'):
          file_name=os.getcwd()+"/detect_im.jpg"
          cv2.imwrite(file_name,frame)
          break
#--------------------Send image to API-----------------------
        url = 'http://127.0.0.1:5000/Emotion'
        my_img = {'image': open(file_name, 'rb')}
        r = requests.post(url, files=my_img)

        # convert server response into JSON format.
        Emotion_reply=r.json()
        print(Emotion_reply)
        plotEmotionWithPercent(Emotion_reply)

#-----------------Wait for user to press any key--------------------
        cv2.waitKey(0)
        video_capture.release()
cv2.destroyAllWindows()

if __name__=="__main__":
        getEmotionWithImage()
```

```
    #getEmotionWithVideo()
```

II. Initiating Server from Flask:

```
from flask import Flask,render_template,request,flash,url_for,redirect,jsonify
from werkzeug.utils import secure_filename
import E_model_loader
import os

#####################################################
#l=learner.load_learner("./models/level1.pth") #
#####################################################

app=Flask(__name__)
app.secret_key = 'h432hi5ohi3h5i5hi3o2hi'

#create a route
'"@app.route('/')
def home():
    return render_template('index.html')"'
Emotion={}
@app.route('/Emotion',methods=['GET','POST'])
def result():
    if request.method == 'POST':
    img = request.files['image']
    path = os.getcwd()+"/detect_im.jpg"
    img.save(path)
```

```
        Emotion=E_model_loader.detect_emotion(path)
        print(Emotion)
        #return jsonify({'msg': 'success', 'size': [img.width, img.height]})
        return jsonify(Emotion)
        else:
        return jsonify({"err": "Only for API purporse"})
        #return jsonify(Emotion)

'"@app.route('/model')
def model():
        return render_template('model.html')"'
```

CHAPTER -VII

SYSTEM TESTING

Emotion Detection:

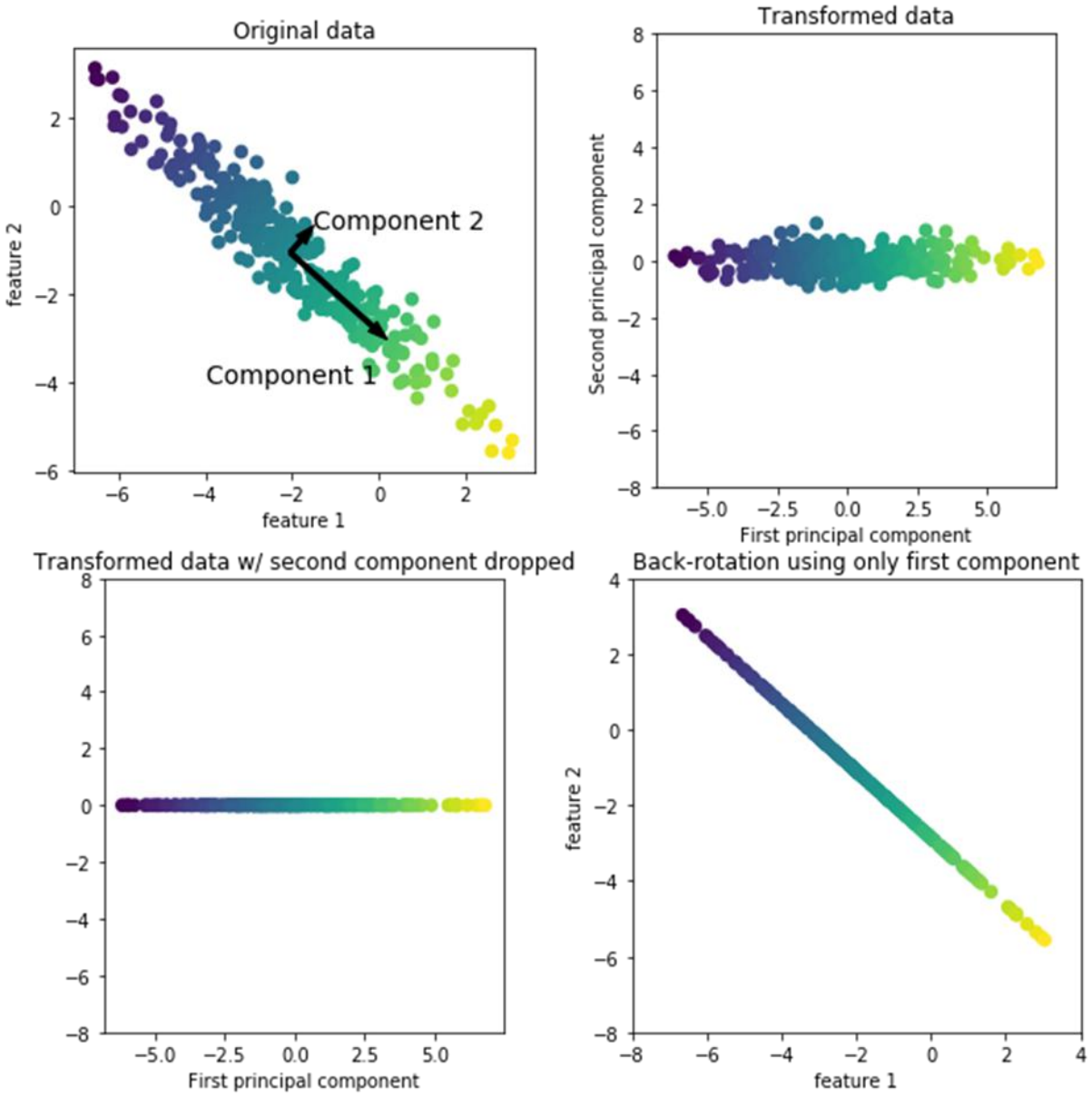
HOST ID	MODULE NAME	INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT	REMARKS
TC01	Emotion Detection	Happy Emotion	Detection of the emotion with the message as HAPPY.	Pass	Detected successfully
TC02	Emotion Detection	Sad emotion	Detection of the emotion with the message as SAD.	Pass	Detected successfully
TC03	Emotion Detection	No emotion	Detection of the emotion with the message as NEUTRAL.	Pass	No accurate data, so detected as Neutral.

PERFORMANCE ANALYSIS

The performance of the CNNs is used extract visual features by learning on a large number of static images. The performance of the proposed system is verified through

leave-one-out cross-validation and compared with that of other models. If the validation loss decreases then the accuracy will increase. The system is applied to a humanoid robot to demonstrate its practicability for improving the HRI. It involves a complex process and since the robot is made the cost of the project is also high and hence we came up with the only software solution which uses API based emotion detector with deep learning technique reducing complexity. Effectiveness of the API is measured through several defeats and experiments by backpropagation of error's. In this experiment we found that using static images average 95% results are correctly matched with the images during training we are getting 95% accuracy on testing.

```
y_hat = grid.predict(X_test)
grid.plot_sample(target_names, y_hat)
```



Analysed Report

CONCLUSION

CHAPTER -VIII

8.1 CONCLUSION:

By Performing CNN on model we are able to get around ~84.2% of accuracy. To increase accuracy and capability of the project we have introduced LSTM (long short term memory) which constantly monitors the performance of the system in prediction and performs addition and removal of dataset from training the model. This repeated process increases the efficiency and performance of the system with good results. This is similar to feed back loop which provides stability to the system

8.2 FUTURE SCOPE:

The future work involves analysis of various emotions and bringing this feature more efficiently and quicker. Focusing in time consumption plays a vital role in future.

APPENDICES

APPENDICES

SCREEN SHOTS:

INPUT

OUTPUT

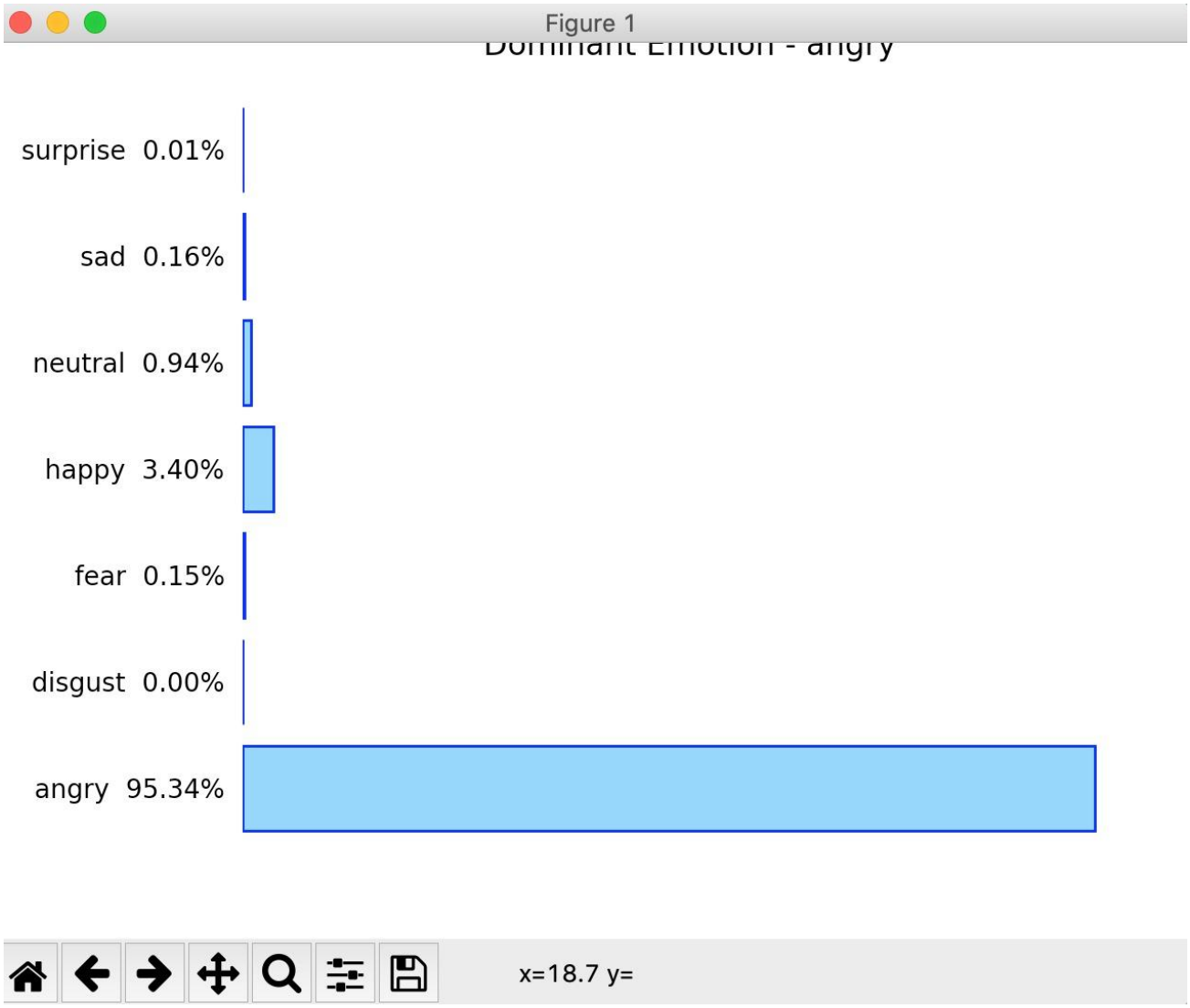


Figure 1
surprise 0.01%
sad 0.16%
neutral 0.94%
happy 3.40%
fear 0.15%
disgust 0.00%
angry 95.34%
x=18.7 y=

OUTPUT

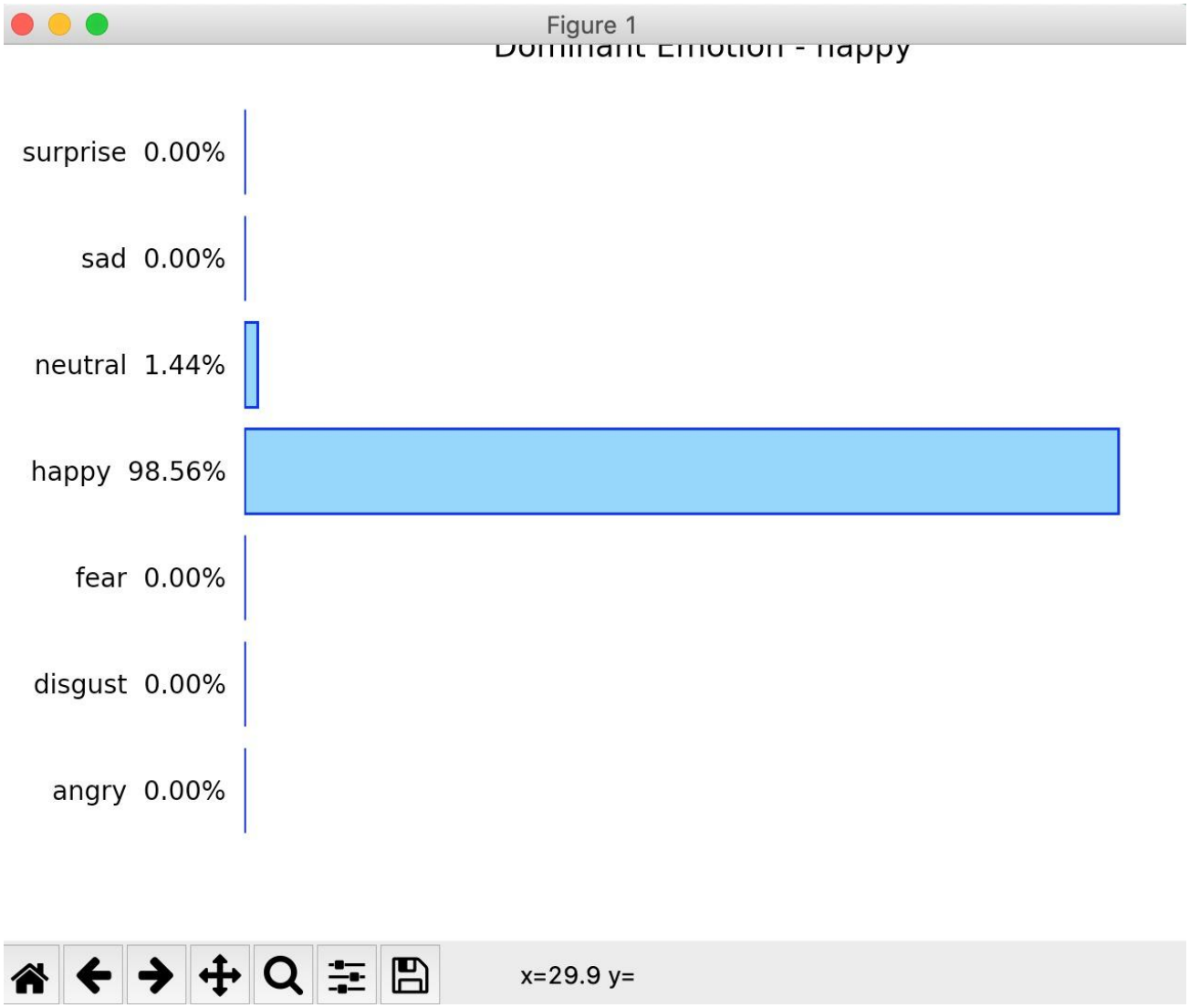


Figure 1
surprise 0.00%
sad 0.00%
neutral 1.44%
happy 98.56%
fear 0.00%
disgust 0.00%
angry 0.00%
x=29.9 y=


tion

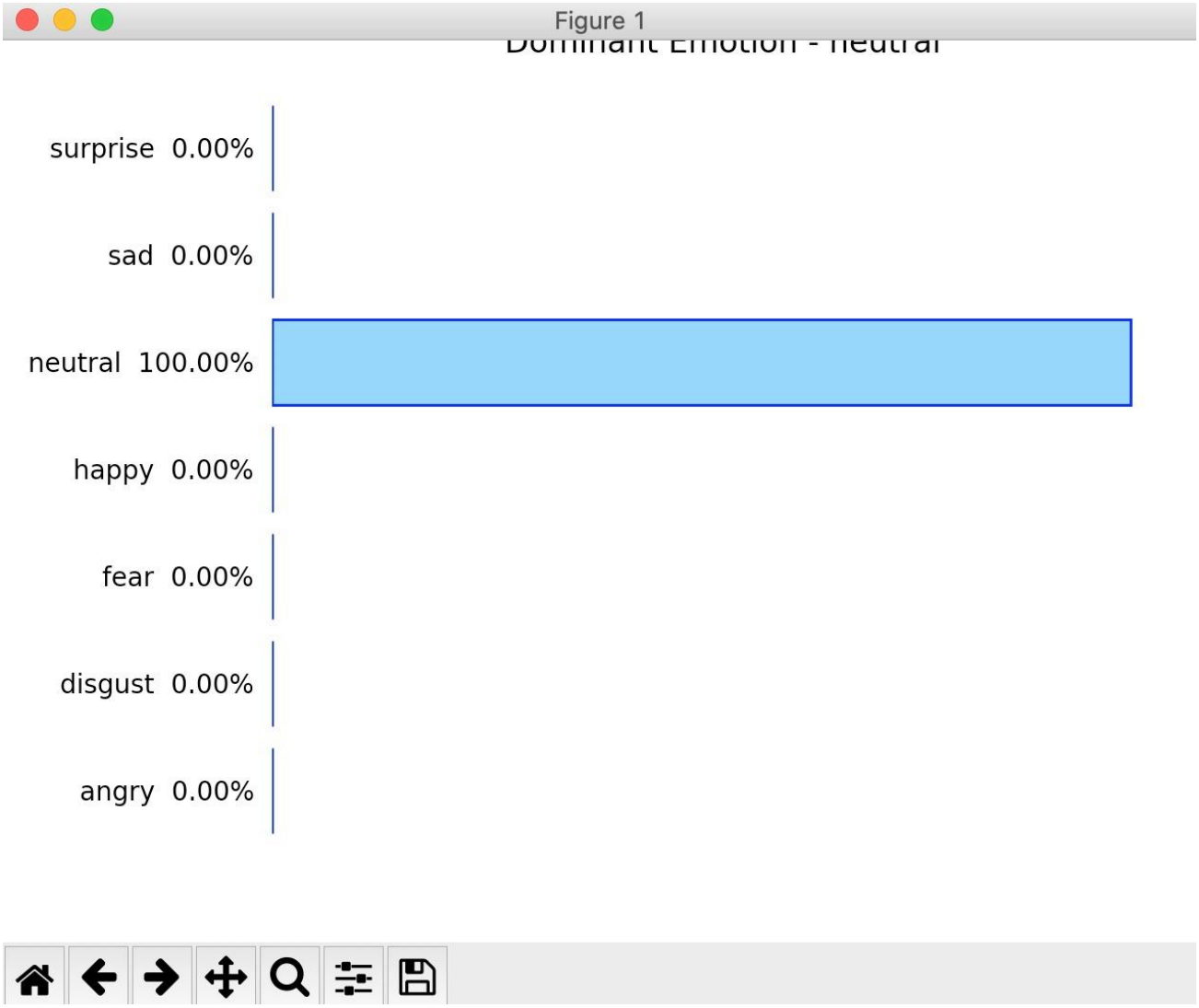


Figure 1
surprise 0.00%
sad 0.00%
neutral 100.00%
happy 0.00%
fear 0.00%
disgust 0.00%
angry 0.00%

PUBLICATIONS

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 04 | Apr 2021 www.irjet.net p-ISSN: 2395-0072

CONVOLUTIONAL NEURAL NETWORKS (CNN) AND LONG SHORT TERM-MEMORY (LSTM) BASED APPLICATION PROGRAMMING INTERFACE (API) TO PREDICT EMOTION OVER WEB.

M Priyadharshini[1], M Nisha[2], Muppala Navya Sree[3], v sathya[4]

[1]Student, B.E Computer science and Engineering, Panimalar Engineering College, Tamilnadu, India
[2]Student, B.E Computer science and Engineering, Panimalar Engineering College, Tamilnadu, India
[3]Student, B.E Computer science and Engineering, Panimalar Engineering College, Tamilnadu, India
[4]Associate Professor, Dept. of Computer Science Engineering, Panimalar Engineering College, Tamilnadu, India

---

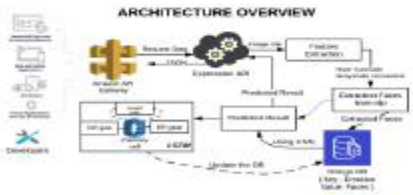
Abstract - *Human emotion plays an important role in the interpersonal relationship. Automatic recognition of emotion has been an active research topic from early years and efficiency has been a challenge. In this project we build an Application Programming Interface (API) to predict facial expression powered by Amazon EC2 Linux instance. Algorithms and models like Long Short Term-Memory (LSTM) and Convolutional Neural Networks (CNN) are used for training and performance evaluation. The potential application of this application includes Facial Expression based feedback in OTG platforms, Facial Expression detection in online interviews, Facial Expression Recognition for security, IOT devices, Connected users and streaming dashboards. Effectiveness of the API is measured through several defects and experiments by backpropagation of errors. The main goal of this project is to produce an enhanced performance level with greater accuracy.*

Key Words: Deep Learning, Cloud Computing (Flask), Web Development, Image Processing, AI, Port Sharing, SSH, Face Recognition, Emotion Detection, Application Programming Interface (API), Long Short Term-Memory (LSTM), Convolutional Neural Networks (CNN), Feedback Loop, Model Training, Data Cleaning.

1. INTRODUCTION

The development and usage of computer systems, software and networks are growing tremendously. These systems have an important role in our everyday life and they make human life much easier. Emotion plays a vital role in determining the thoughts, behavior and feeling of a human. An emotion recognition system can be built by utilizing the benefits of deep learning and different applications such as feedback analysis, face unlocking etc. Can be implemented with good accuracy. Facial emotion recognition system assumes a lot of importance in the era since it can capture the human behavior, feelings, intensions etc. The conventional methods have limited speed and have less accuracy while facial emotion recognition systems using deep learning has proved to be the better one. In this project, we use an Amazon EC2 Linux instance to create an Application Programming Interface (API) that predicts facial expression. Training and performance assessment are carried out using algorithms and models such as Long Short Term-Memory (LSTM) and Convolutional Neural Networks (CNN). A Fully Convolutional Neural Network based classifier is used to predict the facial emotions of the person in the image. An FCN (Fully Convolutional Networks) is a network composed of only convolutional layers, batch norms and non-linearities, i.e., it has no fully Connected (or Dense) layers. Whether s face exists in the image or not is determined by a Viola Jones Detector. Facial Expression Based Feedback in OTG Platforms, Facial Expression Detection in Online Interviews, Facial Expression Recognition for Security, IoT Devices, Connected Users, and Streaming Dashboards are some of the potential applications for this application. Backpropagation of errors is used to assess the API's effectiveness after multiple defeats and experiments.

1.1 Collecting, preparing and processing image Data



The process of gathering data depends on the type of project we desire to make. The data set can be collected from various sources such as a file, database, sensor and many other such sources but the collected data cannot be used directly for performing the analysis process as there might be a lot of missing data, extremely large values, unorganized text data or noisy data. We can also use some free data sets which are present on the internet. Kaggle and UCI machine learning repository are the repositories that are used the most for making Deep Learning models. In this project we use self-collected data to train the model, additional to that we have also used fer2013 facial dataset to improve our

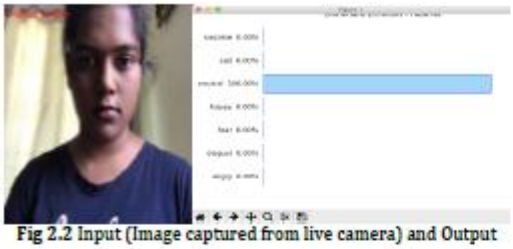
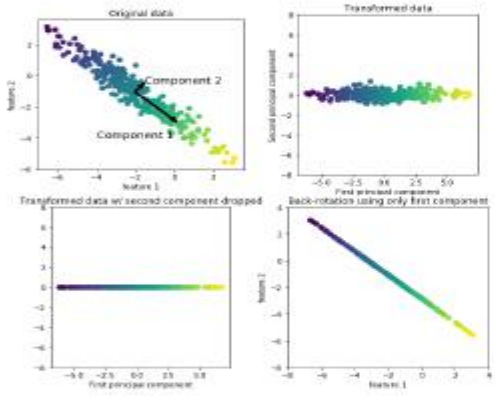


Fig 2.2 Input (Image captured from live camera) and Output

2.2 Performance Analysis

Y_hat = grid.predict(X_test)

Grid.plot_sample(target_names, y_hat)



3. CONCLUSION

By Performing CNN on model, we are able to get around ~84.2% of accuracy. To increase accuracy and capability of the project we have introduced LSTM (Long Short Term-Memory) which constantly monitors the performance of the system in prediction and performs addition and removal of dataset from training the model. This repeated process increases the efficiency and performance of the system with good results. This is similar to feedback loop which provides stability to the system.

REFERENCES

[1]. Dada EG, Bassi JS, Chiroma H, Abdulhamid SM, Adetunmbi AO, Ajibuwa OE. Machine Learning for email spam filtering: review, approaches and open research problems. Heliyon 2019; 5(6):e01802. https://doi.org/10.1016/j.heli

[2]. Xie M. Development of artificial intelligence and effects on the financial system. J Phys Conf 2019;1187:032084. https://doi.org/10.1088/1742-6596/1187/3/032084.

[3] Hegazy O, Soliman OS, Salam MA. A machine learning moel for stock market prediction. Int J Comput Sci Telecommun 2014;4(12):16-23.

[4] Beckmann JS, Lew D. Reconciling evidence-based medicine and precision medicine in the era of big data: challenges and opportunities. Genome Med 2016;8(1):134-9.

[5] Weber GM, Mandl KD, Kohane IS. Finding the missing link for big biomedical data. Jama 2014;311(24):2479-80.

[6] Loconcole C, Chiaaradia D, Bevilacqua V, Frisoli A. Real-time emotion recognition: an improved hybrid approach for classification performance. Intelligent Computing Theory 2014:320-31.

[7] Huang X, Kortelainen J, Zhao G, Li X, Moilanen A, Seppeanen T, Pietikainen M.Multimodal emotion analysis from facial expression and electroencephalogram. Comput Vis Image Understand 2016;147:114-24. https://doi.org/10.1016/j. cviu.2015.09.015.

[8] Raheel A, Majid M, Anwar SM. Facial Expression Recognition based on electroencephalography. In: 2019 2nd international conference on computing, Mathematics and engineering technologies (iCoMET), Sukkur, Pakisthan; 2019.

[9] Vassilis S, Herrmann Jürgen. Where do machine learning and human-computer interaction meet?. 1997.

[10] Keltner D, Ekman P. In: Lewis M, Haviland Jones JM, editors. Facial Expression of emotion, handbook of emotions. New York: Guilford Press; 2000. p. 236-49.

[11] Ekman P. Darwin and facial expression: a century ofresearch in review. United States of America: Academic Press Ishk; 2006.p.1973.

[12] Ekman P, Friesen WV. Constants across cultures in the face and emotion. J Pers Soc Psycchol 1971;17(2):124.

[13] EkmanP. Darwin and facial expresion: a century of research in review. United States of America: Academic Press Ishk; 2006.p. 1973.

[14] Ekman P, Friesen WV, Ancoli S. Facial signs of emotional experience. J Pers Soc Psychol 1980;39:1123-34.

[15] Nguyen BT, Trinh MH, Phan TV, Nguyen HD. An efficient real-time emotion detection using camera and facial landmarks. In: 2017 seventh international conference on information science and technology (ICIST); 2017. https://doi.org/10.1109/icist.2017.7926765.

[16]Lonconsole C, Miranda CR, Augusto G, Frisoli A, Orvalho V. Real-time emotion recognition novel method for geometrical facial features extraction. Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP) 2014:378-85.

[17] Palestra Giuseppe, Pettinicchio Adriaana, Del Coco marco, Ccarcagn Pierluigi, Leo Marco, Distante Ccosimo. Improved performance in facial expression recognition using 32 geometric features. In: Proceedings of the 18th international conference on image analysis and processing. ICIAP; 2015.p.518-28.

[18] Zhang J; Yin Z, Cheng P, Nichele S. Emotion recognition using multi-modal data and machine learning techniques: a tutorial and review. Information fusion. 2020.

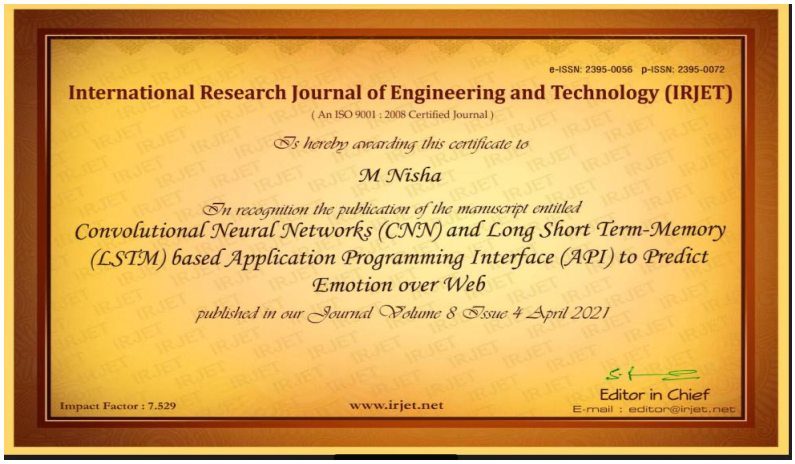
[19] WilsonPI, Fernandez J. Facial feature detection using Haar classifiers. J. Comput. Small Coll., rocnik 21, c. 2006;4. s. 127-133, ISSN 1937-4771.

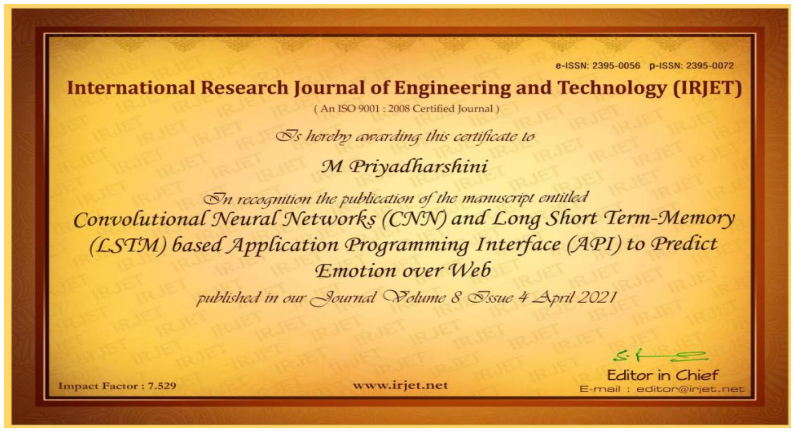
[20] Zhao G, Pietikainen M. Dynamic Texture Recognition Using Volume Local Binary Patterns. In: Vidal R, Heyden A, ma Y, editors. Dynamical Vision. WDV 2006, WDV 2005. Lecture Notes in Computer Science, vol. 4358. Berlin, Heidelberg: Springer; 2007

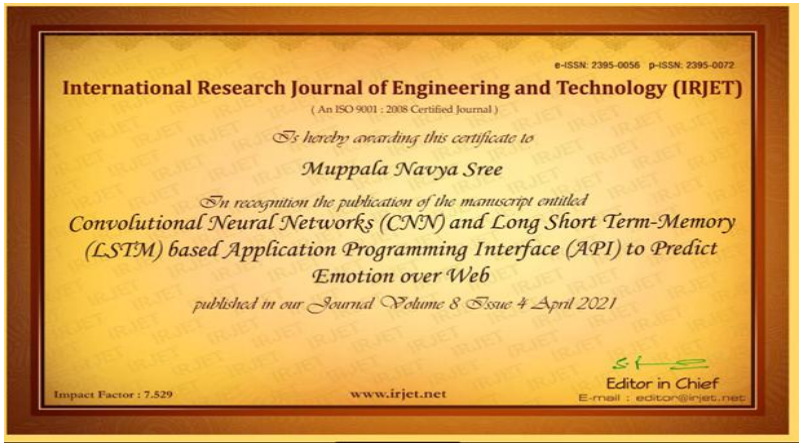
Journal name : **International Research Journal of Engineering and Technology (IRJET)**

Paper title : Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) based Application Programming Interface (API) to predict Emotion over web.

Publicationissues:Volume8,Issue4, April2021 S.No:434 https://www.irjet.net/volume8-issue4






e-ISSN: 2395-0056 p-ISSN: 2395-0072
International Research Journal of Engineering and Technology (IRJET)
(An ISO 9001 : 2008 Certified Journal)
It is hereby awarding this certificate to
Muppala Navya Sree
In recognition the publication of the manuscript entitled
Convolutional Neural Networks (CNN) and Long Short Term-Memory (LSTM) based Application Programming Interface (API) to Predict Emotion over Web
published in our Journal Volume 8 Issue 4 April 2021
Impact Factor : 7.529
www.irjet.net
Editor in Chief
E-mail : editor@irjet.net

REFERENCES

REFERENCES -IX

[1]. Dada EG, Bassi JS, Chiroma H, Abdulhamid SM, Adetunmbi AO, Ajibuwa OE. Machine learning for email spam filtering: review, approaches and open research problems. Heliyon 2019; 5(6):e01802. https://doi.org/10.1016/j.heli

[2]. Xie M. Development of artificial intelligence and effects on the financial system. J Phys Conf 2019;1187:032084. https://doi.org/10.1088/1742-6596/1187/3/032084.

[3] Hegazy O, Soliman OS, Salam MA. A machine learning model for stock market prediction. Int J Comput Sci Telecommun 2014;4(12):16–23.

[4] Beckmann JS, Lew D. Reconciling evidence-based medicine and precision medicine in the era of big data: challenges and opportunities. Genome Med 2016;8(1):134–9.

[5] Weber GM, Mandl KD, Kohane IS. Finding the missing link for big biomedical data. Jama 2014;311(24):2479–80.

[6] Loconsole C, Chiaradia D, Bevilacqua V, Frisoli A. Real-time emotion recognition: an improved hybrid approach for classification performance. Intelligent Computing Theory 2014:320–31.

[7] Huang X, Kortelainen J, Zhao G, Li X, Moilanen A, Sepp€anen T, Pietikainen M. Multimodal emotion analysis from facial expressions and electroencephalogram. Comput Vis Image Understand 2016;147:114–24. https://doi.org/10.1016/j. cviu.2015.09.015.

[8] Raheel A, Majid M, Anwar SM. Facial expression recognition based on electroencephalography. In: 2019 2nd international conference on computing, mathematics and engineering technologies (iCoMET), Sukkur, Pakistan; 2019.

[9] Vassilis S, Herrmann Jürgen. Where do machine learning and human-computer interaction meet?. 1997.

[10] Keltner D, Ekman P. In: Lewis M, Haviland Jones JM, editors. Facial expression of emotion, handbook of emotions. New York: Guilford Press; 2000. p. 236–49.

[11] Ekman P. Darwin and facial expression: a century of research in review. United State Of America: Academic Press Ishk; 2006. p. 1973.
[12] Ekman P, Friesen WV. Constants across cultures in the face and emotion. J Pers Soc Psychol 1971;17(2):124.
[13] Ekman P. Darwin and facial expression: a century of research in review. United State Of America: Academic Press Ishk; 2006. p. 1973.
[14] Ekman P, Friesen WV, Ancoli S. Facial signs of emotional experience. J Pers Soc Psychol 1980;39:1123–34.
[15] Ekman P, Friesen WV, Ancoli S. Facial signs of emotional experience. J Pers Soc Psychol 1980;39:1123–34.
[16] Nguyen BT, Trinh MH, Phan TV, Nguyen HD. An efficient real-time emotion detection using camera and facial landmarks. In: 2017 seventh international conference on information science and technology (ICIST); 2017. https://doi.org/10.1109/icist.2017.7926765.
[17] Loconsole C, Miranda CR, Augusto G, Frisoli A, Orvalho V. Real-time emotion recognition novel method for geometrical facial features extraction. Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP) 2014:378–85.
[18] Palestra Giuseppe, Pettinicchio Adriana, Del Coco Marco, Carcagn Pierluigi, Leo Marco, Distante Cosimo. Improved performance in facial expression recognition using 32 geometric features. In: Proceedings of the 18th international conference on image analysis and processing. ICIAP; 2015. p. 518–28.
[19] Zhang J, Yin Z, Cheng P, Nichele S. Emotion recognition using multi-modal data and machine learning techniques: a tutorial and review. Information fusion. 2020.
[20] Wilson PI, Fernandez J. Facial feature detection using Haar classifiers. J. Comput. Small Coll., rocník 21, c. 2006;4. s. 127-133, ISSN 1937-4771.

[21] Zhao G, Pietikainen M. Dynamic Texture Recognition Using Volume Local Binary Patterns. In: Vidal R, Heyden A, Ma Y, editors. Dynamical Vision. WDV 2006, WDV 2005. Lecture Notes in Computer Science, vol. 4358. Berlin, Heidelberg: Springer; 2007