Face and Mask Detection using OpenCV2 and Python

Manali Pathak[1] Prof. Nehal Rajput[2]
[2]Head of Department
[1,2]Department of Computer Engineering
[1,2]Swarrnim Startup & Innovation University, Bhoyan Rathod, Adalaj Kalol Highway, Gandhinagar, Gujarat India

Abstract— To protect ourselves from the Covid-19 crisis, we must wear masks and sanities ourselves. As a result, wearing the mask is essential for defending yourself against the infection. Here, we've developed software that allows us to determine whether or not an individual is wearing a mask on their face.

Keywords: Covid-19, Mask Detection, OpenCV2, Python

I. INTRODUCTION

The coronavirus, a newly discovered disease, is now infecting the majority of humans. It is highly infectious and, in extreme cases, can result in death. So, in order to protect ourselves, the WHO has developed certain guidelines that require us to wear masks on and sanitise our hands on a regular basis. Though some people mistake it for a common illness and are unaware of the causes after being sick. To shield them and the rest of society, we're developing a project that would require them to wear masks at all times. They will be penalised if they violate those laws, and they will be notified by receiving an electronic mail letter detailing the time span and date that they did not obey the law. In the present world there are various methods through which the computer can automatically detect what the object is? how is the object ? And can even give description of the image. Hence, here the computer detects on its own and is given the command to take the action on their own. This is done by using Artificial Intelligence. There are several fields in which AI is further classified such as Machine Learning which consist of Deep learning, Reinforcement learning and much more. In Machine Learning the machine is given some data to learn on its own and then accordingly it starts behaving. Hence the similar supervised learning is adopted by our program. There will be some images that will be trained, and then that model will be saved and considered further.

II. FRAMEWORK

We have used python for developing the program, numpy, pandas, keras, sklearn, os, cv2, matplotlib/seaborn, tkinter, SMTP, face_recognization and sqlite3 packages.Database is used for storing date and time when violation made. Detailed description of the packages which we have used are as follows:

- NumPy :- It is used for converting array of image into a proper format
- Pandas: - To collect data from a csv file where the name and the email id of the user will be stored.
- Keras: - To build a model stacking several layers on it.
- Sklearn: - In separating the data into train and test and preprocess the data, furthermore to check accuracy of our model.
- Os: - To walk through the directories present in the folder.
- Cv2 :- This is going to read the images and will be converting into an array form. Moreover it will also help us to enable live cameras in our system.
- Matplotlib/Seaborn: - For plotting some figures.
- Tkinter: - To make GUI that will be helpful to manage and see the fines of a person if they have paid or not.
- SMTP :- To send email to those person who were not wearing
- Face_recognization:- to recognize the person.
- Sqlite3 :- the database that we are going to use to store the data

Our project will be a classification issue in which the person's identity and whether or not they have worn a mask will be determined.

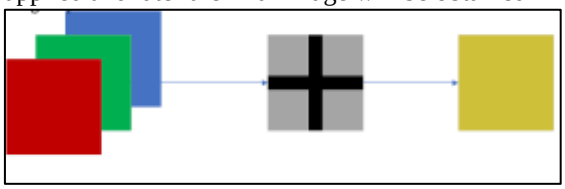
We used a variety of Machine Learning, Deep Learning, and Transfer Learning algorithms to see which of the approaches had the most accurate responses.

Initially we have tried the perceptron algorithm which is the base algorithm of all Machine Learning as well as of Deep learning. The methodology that we have followed is Neural Networks. Where the input data is the image dataset and our target variables is whether the person has worn a mask or not. Furthermore, we have tried AdaBoostClassifier, RandomForestClassifier, DecisionTreeClassifier, Logistic Regression, LinearSVC. Additionally, we have implemented Transfer Learning Models which consist of Deep learning concepts such as CNN, Max Pooling, AveragePooling and some other layers, transfer learning was initially performed on the ImageNet dataset and generated some valuable outputs hence these models are taken into account. So the Transfer Learning Models that we have implemented are Lenet-5, AlexNet, VGG-16 and GoogleNet.

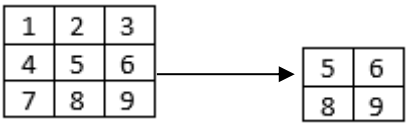
- Perceptron: - An algorithm that has inputs and performs aggregation on it and then at last returns the output.
- Logistic Regression: - An algorithm that is an extended version of perceptron which includes activation function before output. The activation function balances the data.
- AdaBoostClassifier: - it is built by using classifiers that are performing poorly to increase the accuracy. In every step, the weights are set for each classifier in each training sample.
- DecisionTreeClassifier :- An algorithm that predicts the data based on Information Gain, Gini Index and entropy where the training variable is considered and classified and predicts the output.
- RandomForestClassifier :- it consist of many decision trees that helps us to predict the data, the benefit of using RandomForestClassifier is that it uses many Decision trees to predict a single output
- LinearSVC :- It separates the data based on the hyper-plane As transfer learning uses deep learning, the layers in the model that we are going to use and are used are

Convolutional Neural Networks, Flatten, Dense, Batch-Normalization, Activation, Dropout, Max-pooling and Average-Pooling.

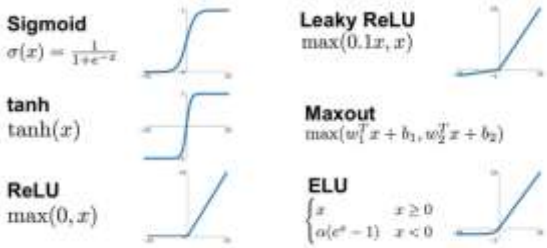
- Convolutional Neural Network :- In our project we will be using RGB mode images where the filters will be applied and later the final image will be obtained



- Max Pooling:- it extracts high intensity value from the image



- Average Pooling :- It is similar to Max pooling but the mean is performed over the image data and the output is generated.
- Activation :- there are various types of activation functions such as sigmoid, tanh, relu, erelu, lrelu



We are going to use relu in our model.

- Batch Normalization :- It normalizes the data from 0 to 1 also it increases the speed.
- Flatten :- it converts the 2D or 3D image to 1D form
- Dense :- It helps to predict the output.

So the transfer learning algorithms we have implemented are Lenet-5, AlexNet, VGG-16 and GoogleNet.

- The architecture of Lenet-5 consists of

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 32, 32, 6)	456
average_pooling2d (AveragePo	(None, 31, 31, 6)	0
conv2d_1 (Conv2D)	(None, 27, 27, 16)	2416
average_pooling2d_1 (Average	(None, 13, 13, 16)	0
conv2d_2 (Conv2D)	(None, 9, 9, 120)	48120
flatten (Flatten)	(None, 9720)	0
dense (Dense)	(None, 84)	816564
dense_1 (Dense)	(None, 2)	170

- The architecture of AlexNet consist of

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 8, 8, 96)	34944
batch_normalization (BatchNo	(None, 8, 8, 96)	384
activation (Activation)	(None, 8, 8, 96)	0
max_pooling2d (MaxPooling2D)	(None, 4, 4, 96)	0
conv2d_4 (Conv2D)	(None, 4, 4, 256)	614656
batch_normalization_1 (Batch	(None, 4, 4, 256)	1024
activation_1 (Activation)	(None, 4, 4, 256)	0
max_pooling2d_1 (MaxPooling2	(None, 2, 2, 256)	0
conv2d_5 (Conv2D)	(None, 2, 2, 384)	885120
batch_normalization_2 (Batch	(None, 2, 2, 384)	1536
activation_2 (Activation)	(None, 2, 2, 384)	0
conv2d_6 (Conv2D)	(None, 2, 2, 384)	1327488
batch_normalization_3 (Batch	(None, 2, 2, 384)	1536
activation_3 (Activation)	(None, 2, 2, 384)	0
conv2d_7 (Conv2D)	(None, 2, 2, 256)	884992
batch_normalization_4 (Batch	(None, 2, 2, 256)	1024
activation_4 (Activation)	(None, 2, 2, 256)	0
max_pooling2d_2 (MaxPooling2	(None, 1, 1, 256)	0
flatten_1 (Flatten)	(None, 256)	0
dense_2 (Dense)	(None, 4096)	1052672
batch_normalization_5 (Batch	(None, 4096)	16384
activation_5 (Activation)	(None, 4096)	0
dropout (Dropout)	(None, 4096)	0
dense_3 (Dense)	(None, 4096)	16781312
batch_normalization_6 (Batch	(None, 4096)	16384
activation_6 (Activation)	(None, 4096)	0
dropout_1 (Dropout)	(None, 4096)	0
dense_4 (Dense)	(None, 1000)	4097000
batch_normalization_7 (Batch	(None, 1000)	4000
activation_7 (Activation)	(None, 1000)	0
dropout_2 (Dropout)	(None, 1000)	0
dense_5 (Dense)	(None, 2)	2002
batch_normalization_8 (Batch	(None, 2)	8
activation_8 (Activation)	(None, 2)	0

- The architecture of VGG-16 consists of

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 32, 32, 64)	1792
conv2d_9 (Conv2D)	(None, 32, 32, 64)	36928
max_pooling2d_3 (MaxPooling2	(None, 16, 16, 64)	0
conv2d_10 (Conv2D)	(None, 16, 16, 128)	73856
conv2d_11 (Conv2D)	(None, 16, 16, 128)	147584
max_pooling2d_4 (MaxPooling2	(None, 8, 8, 128)	0
conv2d_12 (Conv2D)	(None, 8, 8, 256)	295168
conv2d_13 (Conv2D)	(None, 8, 8, 256)	590080
conv2d_14 (Conv2D)	(None, 8, 8, 256)	590080
max_pooling2d_5 (MaxPooling2	(None, 4, 4, 256)	0
conv2d_15 (Conv2D)	(None, 4, 4, 512)	1180160
conv2d_16 (Conv2D)	(None, 4, 4, 512)	2359808
conv2d_17 (Conv2D)	(None, 4, 4, 512)	2359808
max_pooling2d_6 (MaxPooling2	(None, 2, 2, 512)	0
conv2d_18 (Conv2D)	(None, 2, 2, 512)	2359808
conv2d_19 (Conv2D)	(None, 2, 2, 512)	2359808
conv2d_20 (Conv2D)	(None, 2, 2, 512)	2359808
max_pooling2d_7 (MaxPooling2	(None, 1, 1, 512)	0
flatten_2 (Flatten)	(None, 512)	0
dense_6 (Dense)	(None, 4096)	2101248
dense_7 (Dense)	(None, 4096)	16781312
dense_8 (Dense)	(None, 2)	8194

– The architecture of GoogleNet consists of

Layer (type)	Output Shape	Param #
conv2d_21 (Conv2D)	(None, 32, 32, 10)	40
conv2d_22 (Conv2D)	(None, 32, 32, 10)	910
conv2d_23 (Conv2D)	(None, 32, 32, 10)	110
conv2d_24 (Conv2D)	(None, 32, 32, 10)	2510
max_pooling2d_8 (MaxPooling2	(None, 32, 32, 10)	0
flatten_3 (Flatten)	(None, 10240)	0
dense_9 (Dense)	(None, 1200)	12288200
dense_10 (Dense)	(None, 600)	720600
dense_11 (Dense)	(None, 150)	90150
dense_12 (Dense)	(None, 2)	302

– Architecture of our model

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 31, 31, 20)	260
conv2d_1 (Conv2D)	(None, 30, 30, 15)	1215
batch_normalization (BatchNo	(None, 30, 30, 15)	60
max_pooling2d (MaxPooling2D)	(None, 15, 15, 15)	0
activation (Activation)	(None, 15, 15, 15)	0
flatten (Flatten)	(None, 3375)	0
dense (Dense)	(None, 128)	432128
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 16)	528
dense_4 (Dense)	(None, 8)	136
dense_5 (Dense)	(None, 2)	18

III. DATASET

There are images which contain mask and without mask data. The images will be converted to array during processing and there is also face_recognization that will figure out the person automatically and finally there will be the csv file which will contain the name and the email address of the person to whom the email will be sent.
Sample image of dataset.

A. With mask

This is the dataset folder that consists of images of people wearing masks.

B. Without mask

This is the dataset folder that consists of images of people not wearing masks.

C. People Dataset

A folder from which a person will be identified.

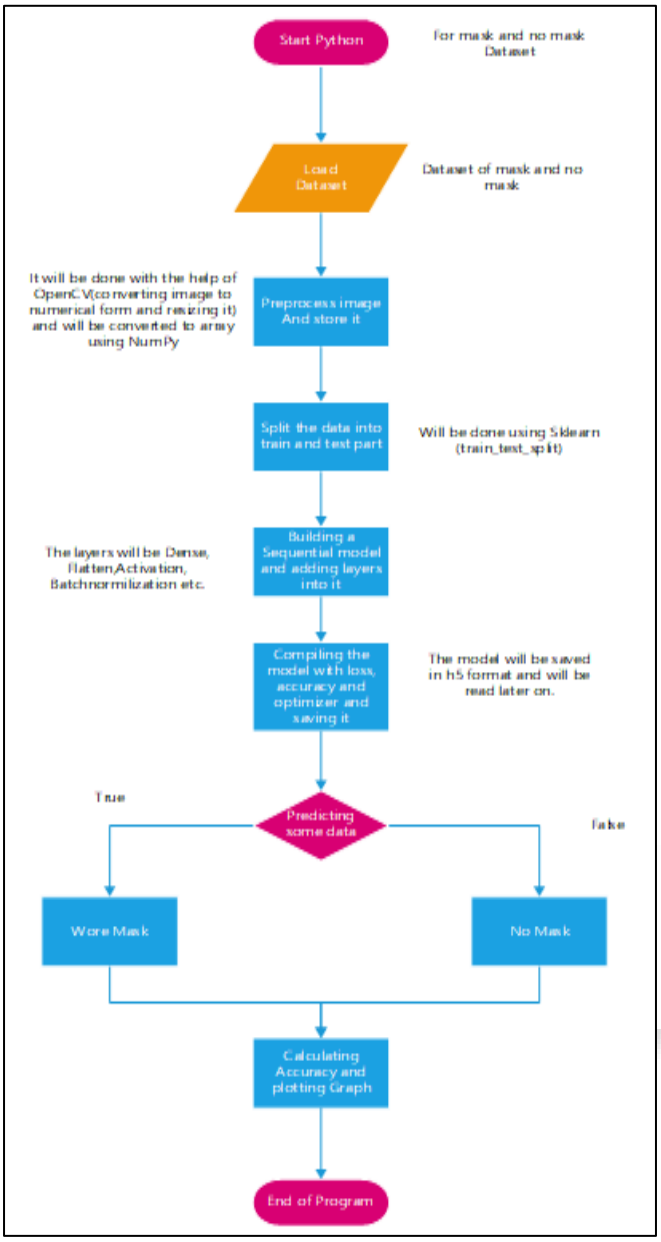
D. CSV file for Emailing

There will be a file that will store the name and email-id of the respective

name, email-address
manali, manalipathak042@gmail.com
nisha, nishasharma@gmail.com
Kandarp,kandarppandya@gmail.com
dhruv,dhruvpatel@gmail.com

IV. ARCHITECTURE OF OUR PROJECT

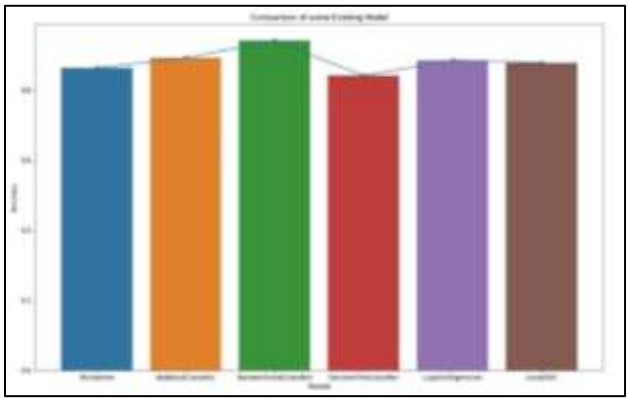
We have developed a program that will be following the given process.



V. RESULTS

We have found that the accuracy in the ML model was

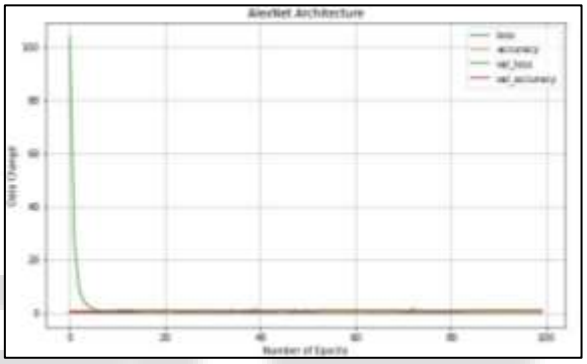
Models	Accuracy
Perceptron	0.862966
AdaBoostClassifier	0.892454
RandomForestClassifier	0.941023
DecisionTreeClassifier	0.841284
LogisticRegression	0.885516
LinearSVC	0.877710



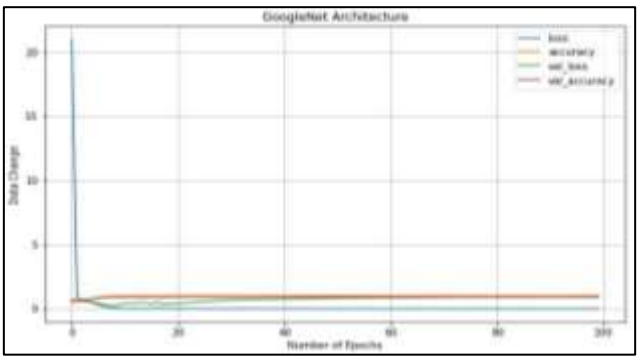
In Transfer learning models we have found that the accuracy were.

TL models	Accuracy
GoogleNet	92.9748482220295
VGG_16	56.72159583694709
AlexNet	94.275802254987
Lenet-5	94.18907198612315

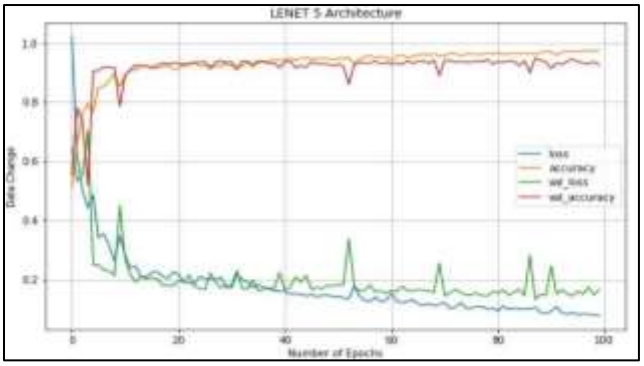
A. Alexnet Outputs



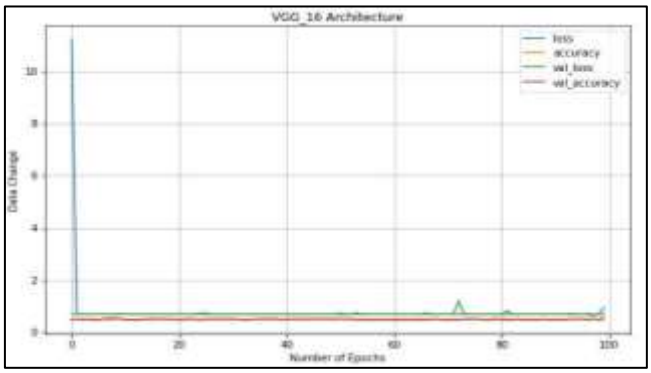
B. GoogleNet outputs



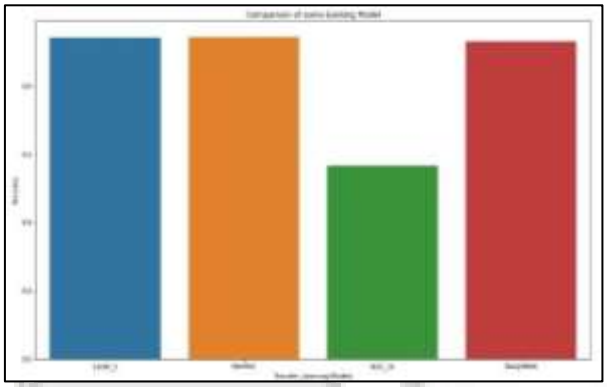
C. Lenet-5 Outputs



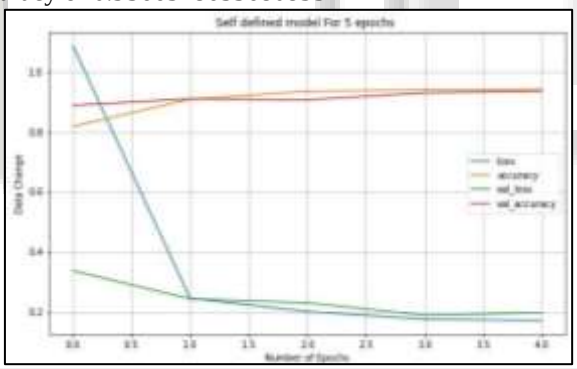
D. VGG-16 Outputs

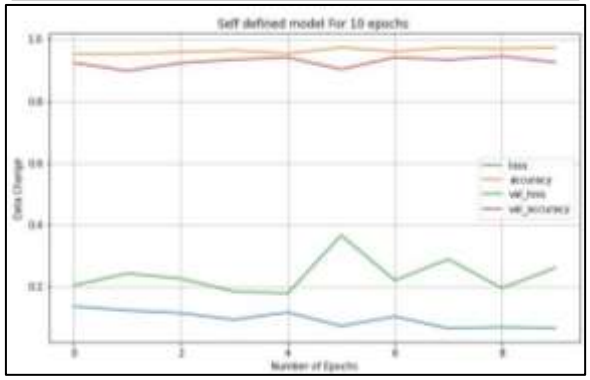


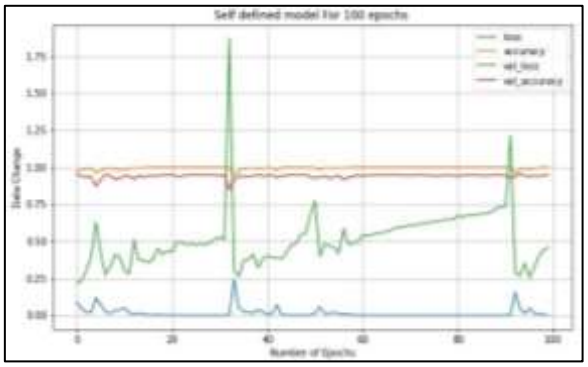
E. Final Comparison of TL models



We have also designed our model which was having accuracy of 0.9566348655680833





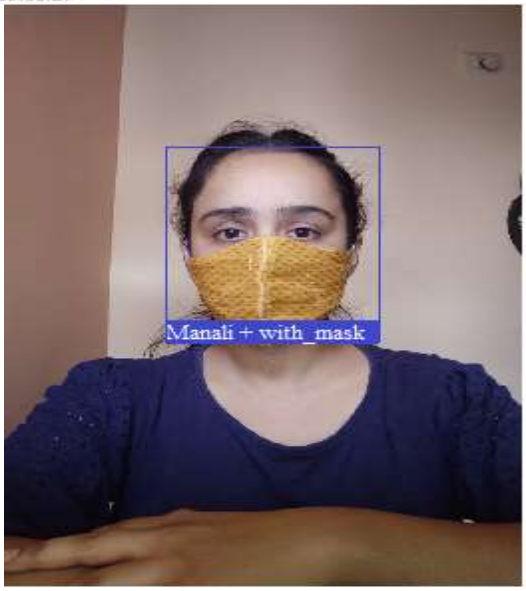


In our designed model, the accuracy increases as we increase the number of epochs. Initially the loss was high which gradually decreased as the number of epochs increased also the validation of data is done along with it. In which validation loss is decreasing and validation accuracy is increasing.

Our project output are

As a precautionary measure it is essential to wear mask for you and for world.

But right now you are fined with 100 Rupees and your reference number is :- 9920461895 on 7/April/2021 at :- 20:53:27

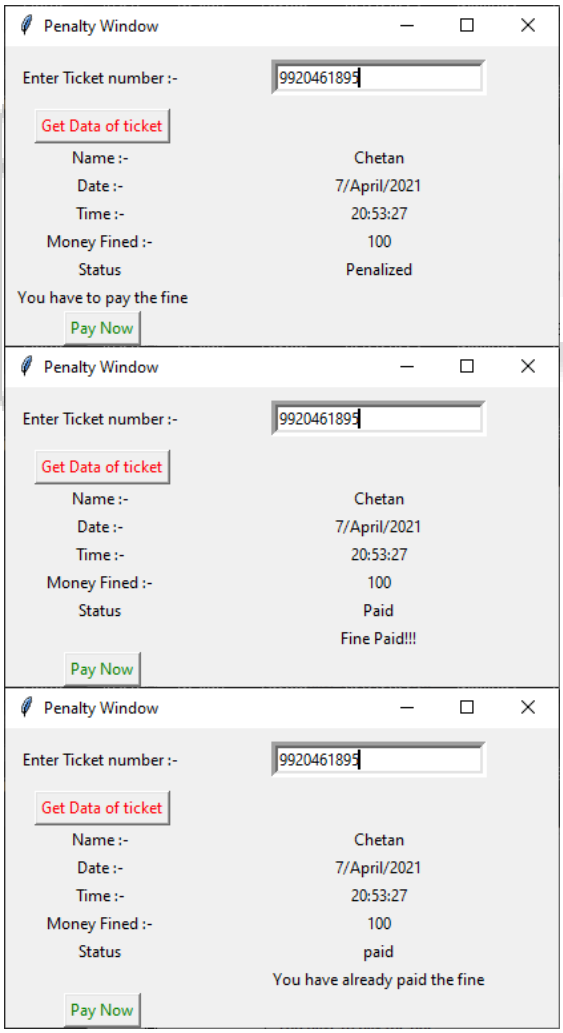


VI. DATABASE

This is our database

	Ticket_Number	Name	Date	Time	Fine	Status
	Filter	Filter	Filter	Filter	Filter	Filter
1	1397861260	Chetan	7/April/2021	20:53:56	100	Penalized
2	1831153726	Unknown	7/April/2021	20:52:45	100	Penalized
3	1948054281	Chetan	7/April/2021	20:52:38	100	Penalized
4	2012548559	Chetan	7/April/2021	20:54:23	100	Penalized
5	2182244112	Unknown	7/April/2021	20:55:42	100	Penalized
6	2245672663	Unknown	7/April/2021	20:55:23	100	Penalized
7	2426429480	Chetan	7/April/2021	20:54:8	100	Penalized
8	2657195539	Chetan	7/April/2021	20:52:58	100	Penalized
9	2712778160	Chetan	7/April/2021	20:53:20	100	Penalized
10	3208741637	Unknown	7/April/2021	20:54:54	100	Penalized
11	3310135283	Chetan	7/April/2021	20:53:41	100	Penalized
12	5192626715	Chetan	7/April/2021	20:53:5	100	Penalized
13	5891857385	Chetan	7/April/2021	20:52:52	100	Penalized
14	6504649447	Chetan	7/April/2021	20:55:0	100	Penalized
15	7858792991	Chetan	7/April/2021	20:55:29	100	Penalized
16	8386419621	Unknown	7/April/2021	20:55:15	100	Penalized
17	8462042391	Chetan	7/April/2021	20:53:48	100	Penalized
18	8465308755	Chetan	7/April/2021	20:55:8	100	Penalized

This is our window where the penalty will be paid and will be updated.



VII. CONCLUSION

We are using our model as the accuracy of our model is much higher than other algorithms which is about 95%.

VIII. FUTURE WORK

We will try to integrate in the client-side project and will be developing a website related to it.

REFERENCES

[1] Face Detection and Recognition using OpenCV and Python Department of Computer Engineering, Dr. Babasaheb Ambedkar Technological University Raigad, India
Link:https://www.irjet.net/archives/V7/i10/IRJET-V7I10219.pdf

[2] Face Detection and Recognition using Open CV Based on Fisher Faces Algorithm J.Manikandan, S. Lakshmi Prathyusha, P. Sai Kuma, Y. Jaya Chandra, M. Umaditya Hanuman
Link:https://www.ijrte.org/wp-content/uploads/papers/v8i5/E5753018520.pdf

[3] Real Time Face Detection and Tracking Using OpenCV Prof.P Y Kumbhar1 , Mohammad Attaullah2, Shubham Dhere3, Shivkumar Hipparagi4
Link:https://ijrest.net/downloads/volume-4/issue-4/pid-ijrest-44201715.pdf

[4] Real-Time Facemask Recognition with Alarm System using Deep Learning by Sammy V.Militante; Nanette V. Dionisio
Link:https://ieeexplore.ieee.org/document/9232610/figures#figures

[5] Face Detection and Segmentation Based on Improved Mask R-CNN by Kaihan Lin, Huimin Zhao, Jujian Lv, Canyao Li, Xiaoyong Liu, Rongjun Chen, and Ruoyan Zhao
Link:https://www.hindawi.com/journals/ddns/2020/9242917/

[6] Efficient Masked Face Recognition Method during the COVID-19 Pandemic by Walid Hariri
Link:https://www.researchsquare.com/article/rs-39289/v1

[7] COVID-19 FACEMASK DETECTION WITH DEEP LEARNING AND COMPUTER VISION by Vinitha,Velantina
Link:https://www.irjet.net/archives/V7/i8/IRJET-V7I8530.pdf

[8] Face Recognition: Issues, Methods and Alternative Applications by Waldemar Wójcik, Konrad Gromaszek and Muhtar Junisbekov
Link:https://www.intechopen.com/books/face-recognition-semisupervised-classification-

[9] A Study on CNN Transfer Learning for Image Classification. Mahbub Hussain, Jordan J. Bird, and Diego R. Faria, School of Engineering and Applied Science, Aston University, Birmingham, B4 7ET, UK. {hussam42, birdj1, d.faria}@aston.ac.uk.

[10] Review on Literature Survey of Human Recognition with Face Mask Volume 10, Issue 01 (January 2021) Dr. Vandana S. Bhat, Arpita Durga Shambavi, Komal Mainalli, K M Manushree, Shraddha V Lakamapur
Link:https://www.ijert.org/review-on-literature-survey-of-human-recognition-with-face-mask

