A PROJECT REPORT
on

"FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK"

Submitted to
KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING

BY

Ajit	2006161
Ayush Raj	2006120
Happy Ranjan	2006432
Meetali Verma	2006224

UNDER THE GUIDANCE OF

Dr. Subhashree Darshana

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

May 2024

A PROJECT REPORT

on

"FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK"

Submitted to
KIIT Deemed to be University

In Partial Fulfilment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING

BY

Ajit	2006161
Ayush Raj	2006120
Meetali Verma	2006224
Happy Ranjan	2006432

UNDER THE GUIDANCE OF

Dr. Subhashree Darshana

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAE, ODISHA -751024

May 2024

KIIT Deemed to be University
School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

"FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK"

submitted by

Ajit	2006161
Ayush Raj	2006120
Meetali Verma	2006224
Happy Ranjan	2006432

is a record of bonafide work carried out by them, in the partial fulfilment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering OR Information Technology) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2023-2024, under our guidance.

Date: 01/05/24

(Dr. Subhashree Darshana)

Project Guide

Acknowledgements

We are profoundly grateful to **Subhashree Darshana** of **Affiliation** for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. ....

Ajit
Ayush Raj
Meetali Verma
Happy Ranjan

ABSTRACT

One statement overview: Utilizing its capacity to efficiently learn features from image data, the CNN model and other Python packages are used in this project to detect face masks, guaranteeing precise classification of masked and unmasked faces.to use a CNN that has been trained using publicly available images from a specified image dataset. It is therefore determined that brain organizations will detect the surfaces and shadings of sores that are obvious to a few veils and resemble human dynamics.

Problem statement:With potential applications in public safety, healthcare, and security, this project aims to construct an accurate face mask recognition system utilizing convolutional neural networks (CNNs) to automatically determine whether people are wearing masks.

Methods:Deep learning and computer vision techniques are used in face mask identification. First, face detection algorithms find faces in pictures. Deep learning methods evaluate facial features to reliably detect whether people are wearing masks. These models are often CNNs (Convolutional Neural Networks).To maximize performance, these models go through a thorough training process using annotated datasets. The models are incorporated into the real-time detection system after training. Ensuring robustness and accuracy through ongoing development, evaluation, and user testing helps to contribute to successful public health compliance measures throughout the COVID-19 pandemic.

Results:By automating the tracking of face mask compliance in public areas, it seeks to improve public health initiatives. To reliably identify people wearing masks or not, the system combines deep learning model inference, real-time image collection, and alerting methods. The technology contributes to community safety by limiting viral transmission and ensuring scalability, privacy compliance, and dependability through extensive testing and validation.

Contents

Chapter 1

Introduction

Those who get COVID-19, a respiratory disease, often experience severe cases of pneumonia. In addition to respiratory droplets, salivation beads, or nasal droplets released by an infected person when they cough, sneeze, or breathe the virus into the atmosphere, direct contact with an infected person can also result in the transmission of the sickness. The COVID-19 virus kills hundreds of individuals every day throughout the world.People ought to stay away from close quarters and wear face masks to stop viruses from spreading sickness. The goal of an effective and successful computer vision technique is to develop a real-time program that observes individuals in public, whether or not they are wearing face masks. The first step in figuring out if it has a mask on it is to locate the face.This divides the entire procedure into two parts: facial mask detection.

Primary Objectives

The primary objective of this application were:
Real-time Detection: Determine whether people are wearing face masks in different settings right away..
Promote Compliance: Informing people or authorities about non-compliance with public health norms can help to promote adherence to them.
Enhance Safety: Ascertain that face masks are used appropriately in crowded or enclosed areas to reduce the chance of virus transmission.

Chapter 2

Basic Concepts/ Literature Review

Models used in projects:

In a face mask detection app, commonly used modules include OpenCV for image processing, TensorFlow for deep learning, Flask or Django for backend development, and streamlit for front-end development.These modules enable efficient face detection, mask classification and web service deployment.

Relevent Technologies:

Computer Vision: Used for facial feature detection and real-time picture processing..
Deep Learning: Uses convolutional neural networks (CNNs) to classify face masks with accuracy.
Machine Learning Algorithms: Trained models to discern between mask-wearing and mask-free faces.
IoT Sensors: Integrated with cameras for data acquisition and analysis in various environments can be used to further enhance the project capabilities.

Existing Research:

The majority of current research in face mask identification concentrates on using deep learning methods, such as convolutional neural networks (CNNs), to achieve precise detection. Techniques for augmenting data and transfer learning are frequently used to solve problems with limited datasets. In order to improve detection performance, research also looks into the fusion of multimodal data, such as visible light and thermal imaging. Two more topics that are being actively researched for practical application are real-time deployment tactics and privacy-preserving techniques.
.

Key Takeaway:

The project's foundation is a thorough understanding of machine learning models that makes use of AI, computer vision, and contemporary web development tools.
These devices help slow the transmission of infectious diseases, especially in confined or congested areas where it is difficult to keep a physical distance. But there are issues that need to be resolved, like algorithm accuracy, privacy issues, and rollout difficulties. In the end, efficient face mask detection lowers the danger of virus transmission, promotes community health and well-being, and aids in healthcare initiatives and the safe reopening of economies.
Face mask detection systems use cutting-edge technology like deep learning, machine learning, and computer vision to automate monitoring and guarantee that mask-wearing procedures are followed in a variety of situations.

The integrated nature of this projects:

Combines software (AI algorithms) with hardware (cameras, for example). Entails processing, analyzing, and making decisions in real time with data.coordination between data science, engineering, and public health specialists is necessary. takes privacy and ethical issues into account for responsible deployment.

Chapter 3

Problem Statement / Requirement Specifications

Detailed Problem Statement

The multifaceted nature of content development frequently calls for the usage of several specialized tools.Think on these key ideas.:

Work-flow fragmentation: Disjointed procedures in data collection, pre-processing, model training, and deployment are the cause of it. Disparate toolchains, inconsistent data formats, and unstandardized protocols impede scaling and collaboration. Subpar performance, inefficiencies, and delays in model iteration are caused by this fragmentation. Simplifying work processes, putting in place standardized data pipelines, and integrating tools to enable smooth stage transitions are all necessary to address this problem. Project implementation can run more smoothly when fragmentation is reduced through the use of centralized platforms that promote automation and cooperation.

Technical barriers:The accurate recognition of masks in different lighting conditions, mask kinds, and facial orientations are among the technical obstacles to face mask detection. Complicacy is increased by the lack of annotated datasets, the need for powerful computers for deep learning models, and privacy issues with facial recognition. Robust algorithms, a wide range of training data, and ethical considerations are necessary to overcome these obstacles.

Cost and complexity: Face mask detection is expensive and complex because it requires sophisticated software, such as deep learning algorithms, and sophisticated hardware, including high-resolution cameras. These requirements call for significant resources and experience.

System Requirements:

Functional Requirements

Core Content Generation Features

Real-time Detection: Face masks should be quickly detected by the system to allow for alerts or quick intervention.
High Accuracy: It must be able to tell the difference between people who are and are not wearing masks.
Adaptability: It should be able to work with various mask kinds and in a variety of lighting situations.
User Interface: It is imperative that administrators have an easy-to-use interface to monitor compliance, modify settings, and examine reports.

Non Functional Requirements:

Performance: To ensure prompt detection, the system should have minimal latency and quick response times.
Reliability: It needs to function regularly and dependably, even in harsh environmental circumstances.
Privacy: Respect for privacy laws by preserving and anonymizing sensitive information, such as face photos.
Scalability: Capacity to manage growing workloads and adjust to demand variations without sacrificing performance.
Security: Putting strong security measures in place to guard against illegal access to the system and guarantee data integrity.

3.1 Project Planning

Timeline:

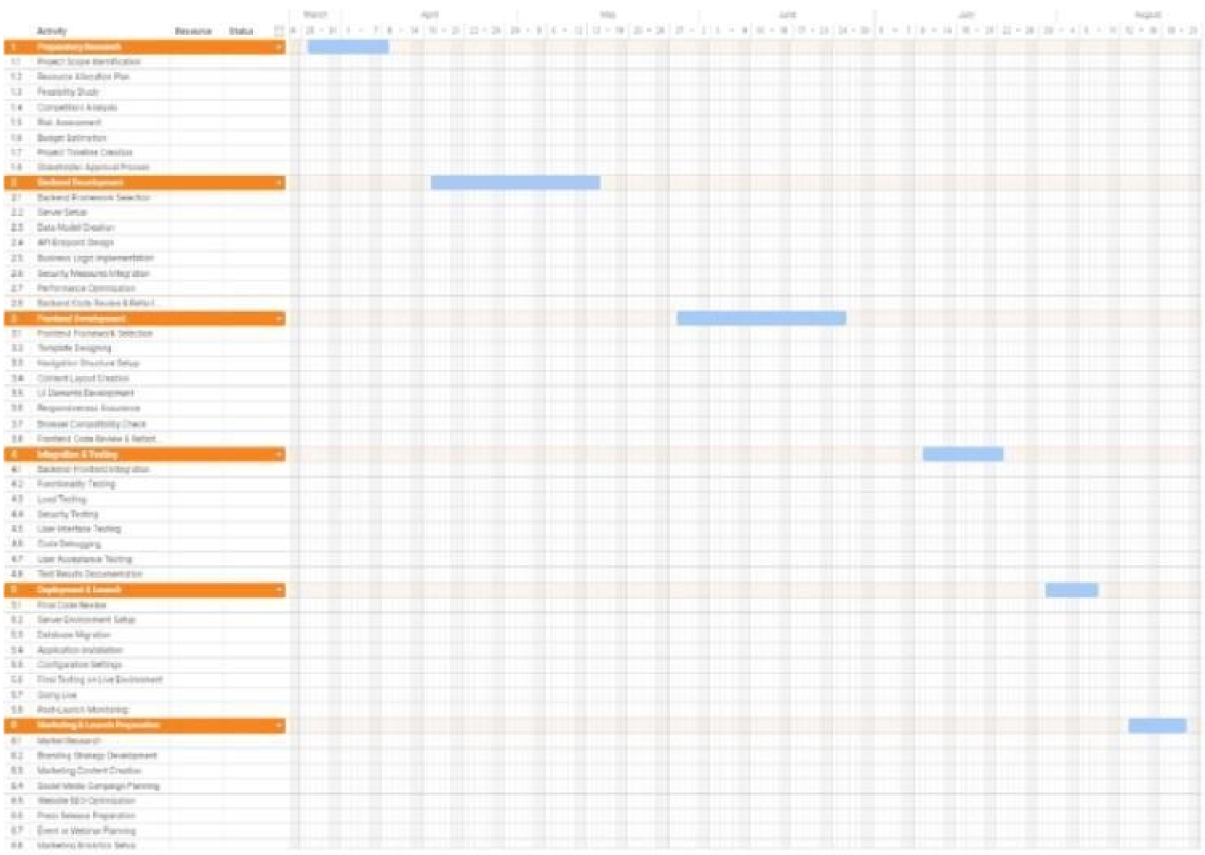


Fig. 3.1. Gantt-Chart

Resource allocation:

Team Roles and Skills:

AI/ML Engineer: For integrating customs solutions or optimizing models
UX Designer:Pay attention to the user's intuitive experience.

Hardware:
Determining and obtaining the servers, GPUs, and cameras that are required for data processing.

Software:
selecting suitable software tools and frameworks for model training, deployment, and data pre-processing.
Database management tools
Code editor
Jupyter Book

3.3.1 System Design

Design Constraints

The following are some typical limitations and possible ways to mitigate them:

Pre-processing: Utilize image processing techniques to adjust illumination, reduce noise, and improve image quality.
Mask Classification: Use deep learning algorithms to categorize faces into mask-wearing and non-masking categories.
Deployment: Install the system at the intended locations, taking into account things like power supply, network connectivity, and maintenance needs.
Data Storage and Logging: Save the processed photos and detection outcomes for use in reporting and analysis.

3.3.2 System Architecture

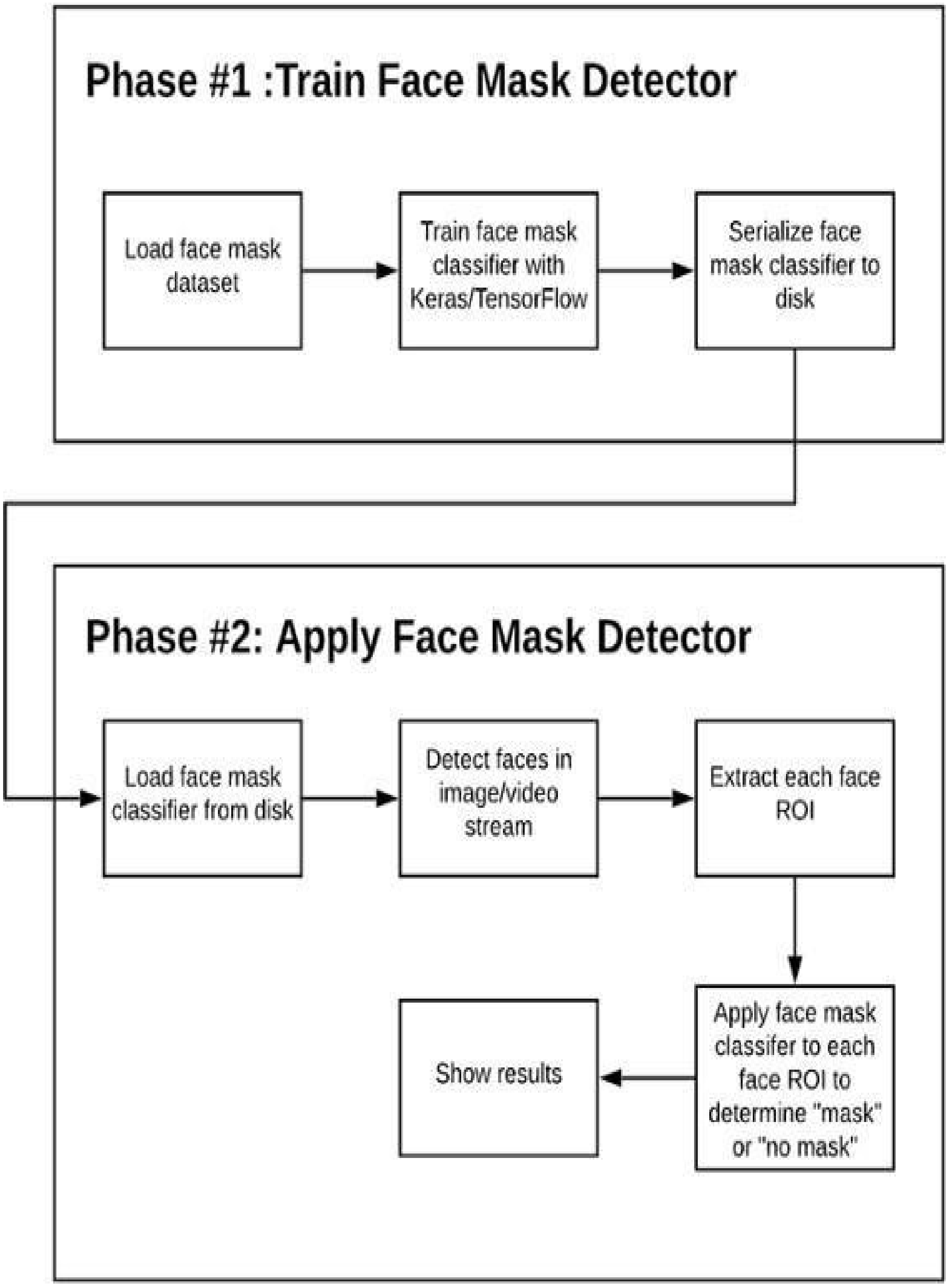

Phase #1 :Train Face Mask Detector
Load face mask dataset
Train face mask classifier with Keras/TensorFlow
Serialize face mask classifier to disk
Phase #2: Apply Face Mask Detector
Load face mask classifier from disk
Detect faces in image/video stream
Extract each face ROI
Apply face mask classifer to each face ROI to determine "mask" or "no mask"
Show results

Fig. 3.3.2. Face Mask Detector Architecture

Chapter 4

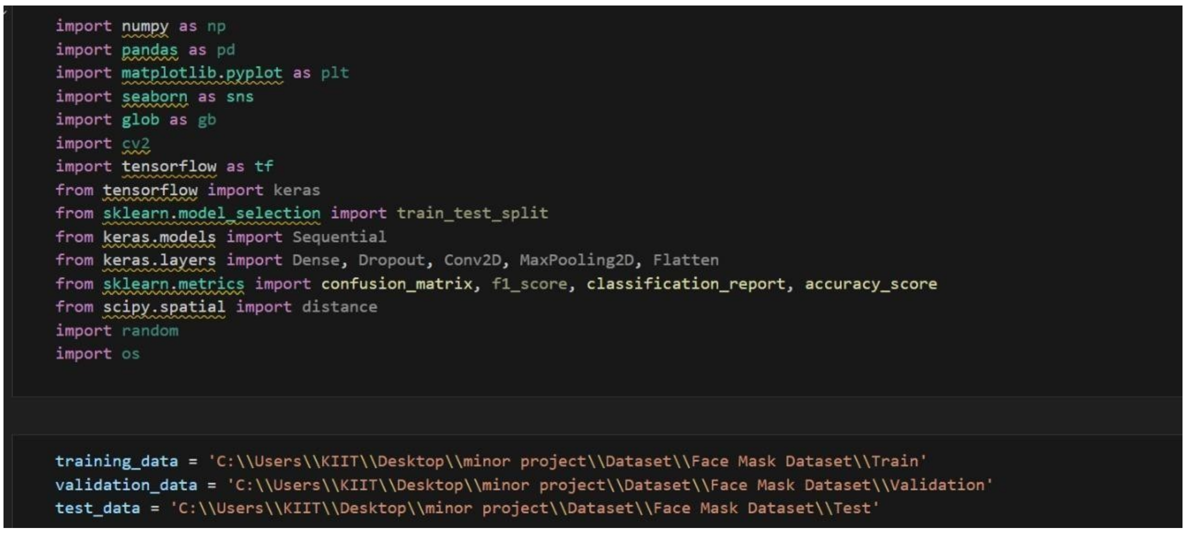
Implementation

4.1 Methodology OR Proposal

Utilizing computer vision and deep learning approaches, the suggested face mask identification methodology creates an automated system. The first step in the data collection process will be to compile a variety of face photos, both with and without masks. Pre-processing techniques will improve picture quality and level out lighting. Next, the data-set will be used to train Convolution neural networks (CNNs), a kind of deep learning model, to correctly identify masked and unmasked faces. Lastly, to ensure public health compliance, the trained model will be implemented for real-time detection in a variety of settings.

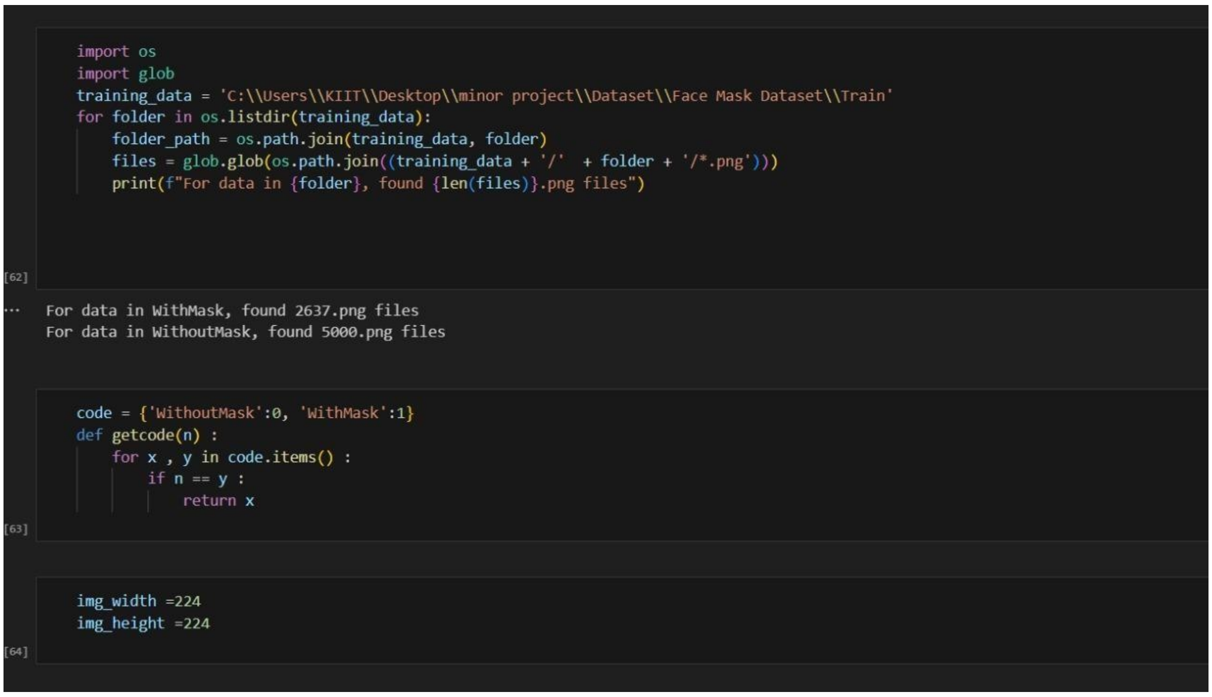
Justification:
Enforcing public health measures during the COVID-19 outbreak requires face mask detection. Automated technologies improve compliance in crowded areas where physical distance is difficult and reduce the strain of manual monitoring. These methods help to contain viral transmission overall by providing precise and real-time detection through the use of computer vision and deep learning. In addition to protecting public health, this technology enhances social cohesion by lowering the possibility of epidemics and making it easier for businesses and public areas to safely reopen.

Development Process



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import glob as gb
import cv2
import tensorflow as tf
from tensorflow import keras
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, Dropout, Conv2D, MaxPooling2D, Flatten
from sklearn.metrics import confusion_matrix, f1_score, classification_report, accuracy_score
from scipy.spatial import distance
import random
import os
```

```
training_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Train'
validation_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Validation'
test_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Test'
```

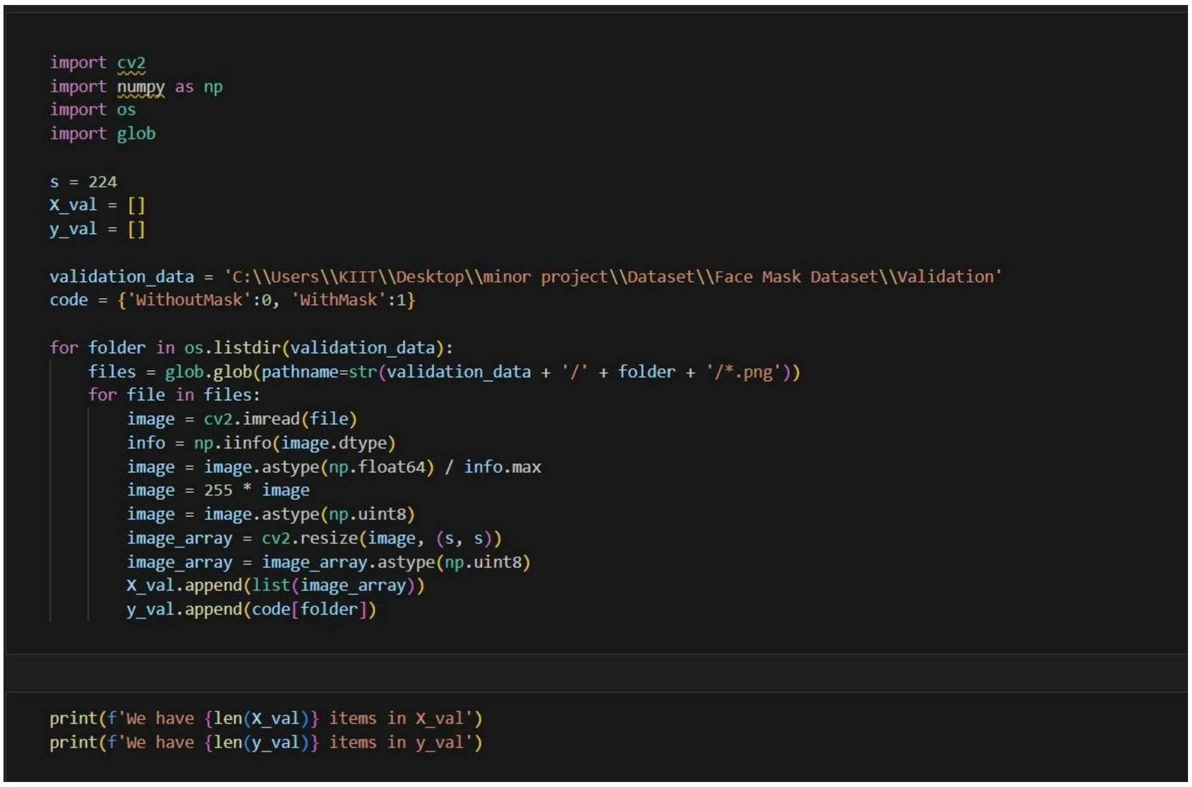


```
import os
import glob
training_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Train'
for folder in os.listdir(training_data):
    folder_path = os.path.join(training_data, folder)
    files = glob.glob(os.path.join((training_data + '/'  + folder + '/*.png')))
    print(f"For data in {folder}, found {len(files)}.png files")
```
[62]

```
For data in WithMask, found 2637.png files
For data in WithoutMask, found 5000.png files
```

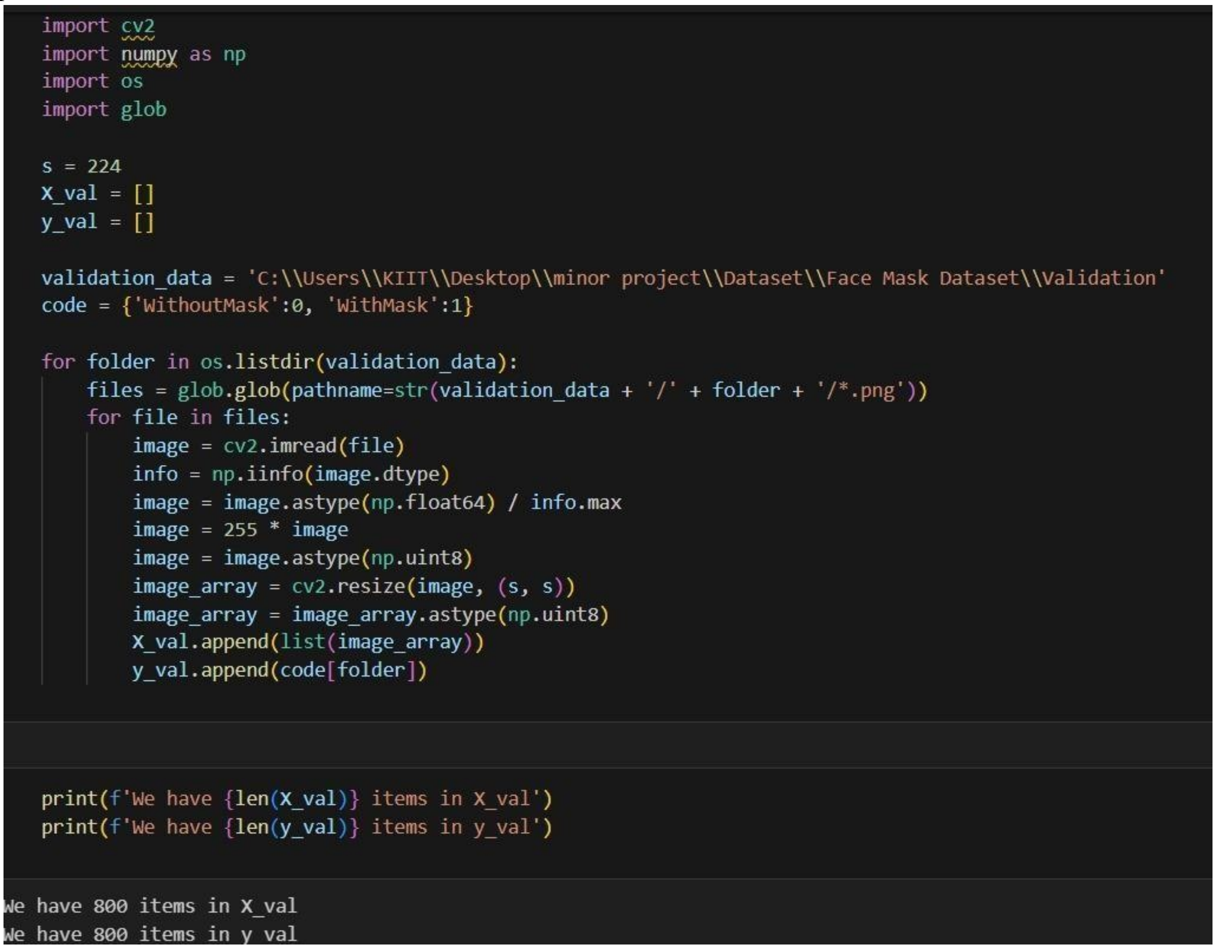
```
code = {'WithoutMask':0, 'WithMask':1}
def getcode(n) :
    for x , y in code.items() :
        if n == y :
            return x
```
[63]

```
img_width =224
img_height =224
```
[64]



```
import cv2
import numpy as np
import os
import glob

s = 224
X_val = []
y_val = []

validation_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Validation'
code = {'WithoutMask':0, 'WithMask':1}

for folder in os.listdir(validation_data):
    files = glob.glob(pathname=str(validation_data + '/' + folder + '/*.png'))
    for file in files:
        image = cv2.imread(file)
        info = np.iinfo(image.dtype)
        image = image.astype(np.float64) / info.max
        image = 255 * image
        image = image.astype(np.uint8)
        image_array = cv2.resize(image, (s, s))
        image_array = image_array.astype(np.uint8)
        X_val.append(list(image_array))
        y_val.append(code[folder])
```

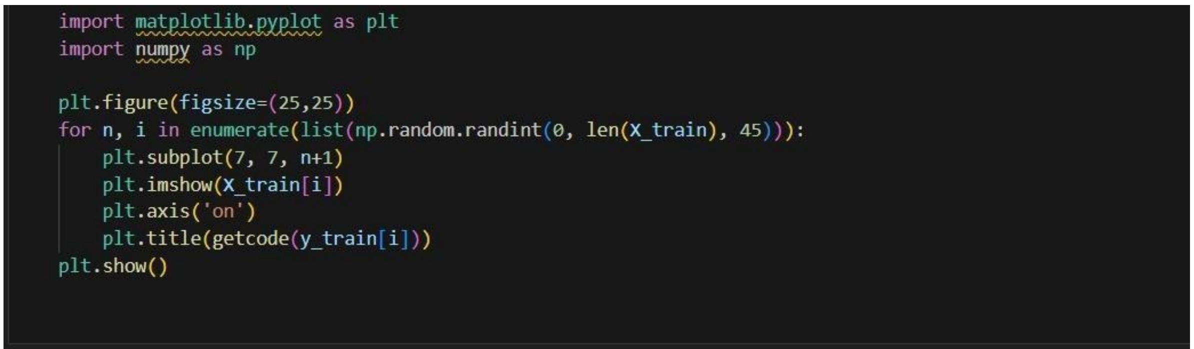
```
print(f'We have {len(X_val)} items in X_val')
print(f'We have {len(y_val)} items in y_val')
```



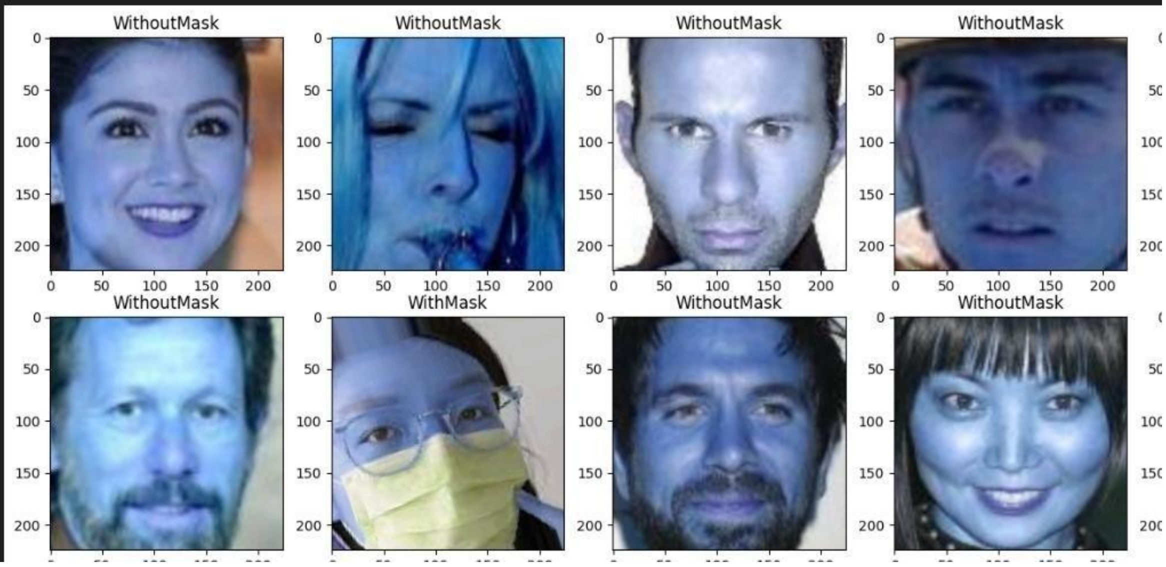
```
import cv2
import numpy as np
import os
import glob

s = 224
X_val = []
y_val = []

validation_data = 'C:\\Users\\KIIT\\Desktop\\minor project\\Dataset\\Face Mask Dataset\\Validation'
code = {'WithoutMask':0, 'WithMask':1}

for folder in os.listdir(validation_data):
    files = glob.glob(pathname=str(validation_data + '/' + folder + '/*.png'))
    for file in files:
        image = cv2.imread(file)
        info = np.iinfo(image.dtype)
        image = image.astype(np.float64) / info.max
        image = 255 * image
        image = image.astype(np.uint8)
        image_array = cv2.resize(image, (s, s))
        image_array = image_array.astype(np.uint8)
        X_val.append(list(image_array))
        y_val.append(code[folder])
```

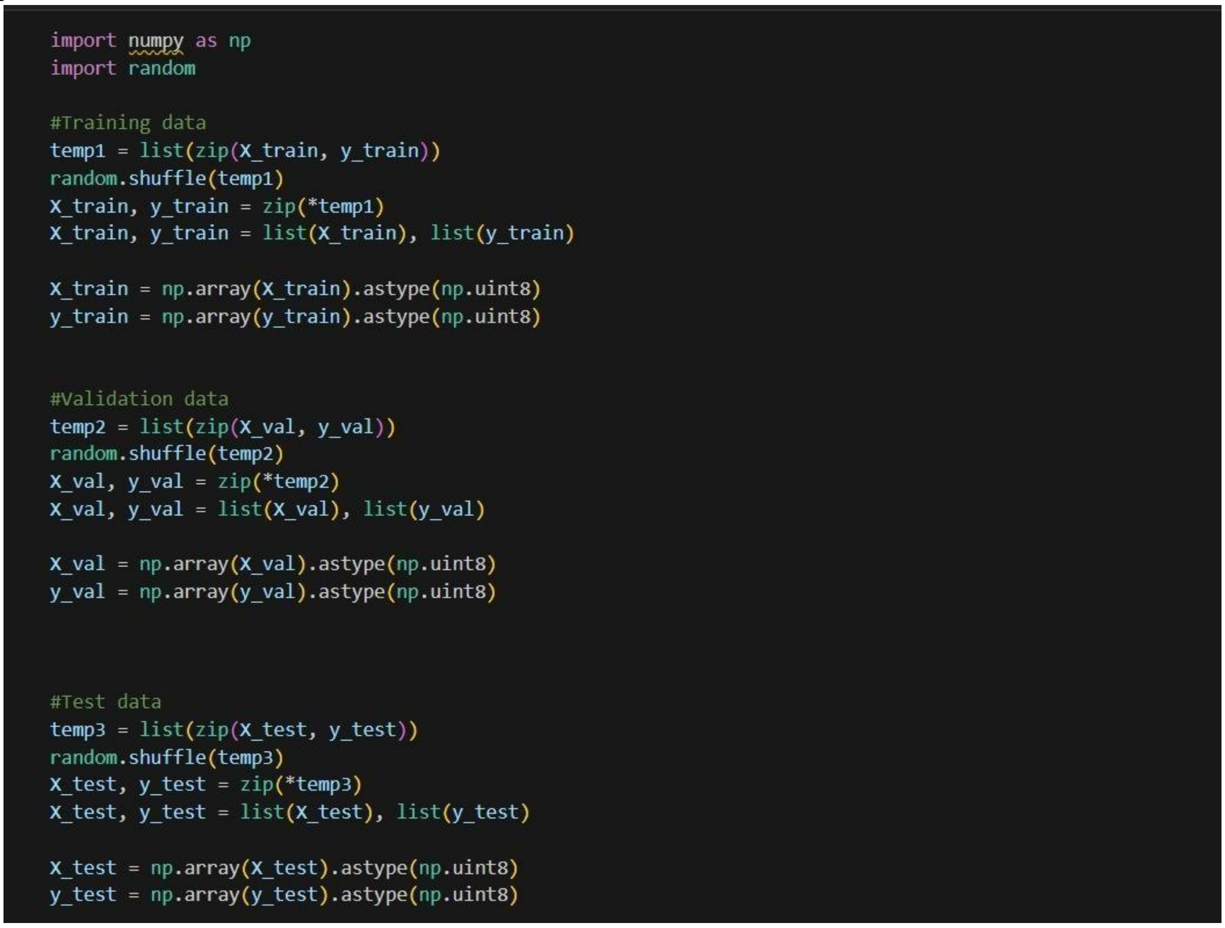
```
print(f'We have {len(X_val)} items in X_val')
print(f'We have {len(y_val)} items in y_val')
```

```
We have 800 items in X_val
We have 800 items in y_val
```



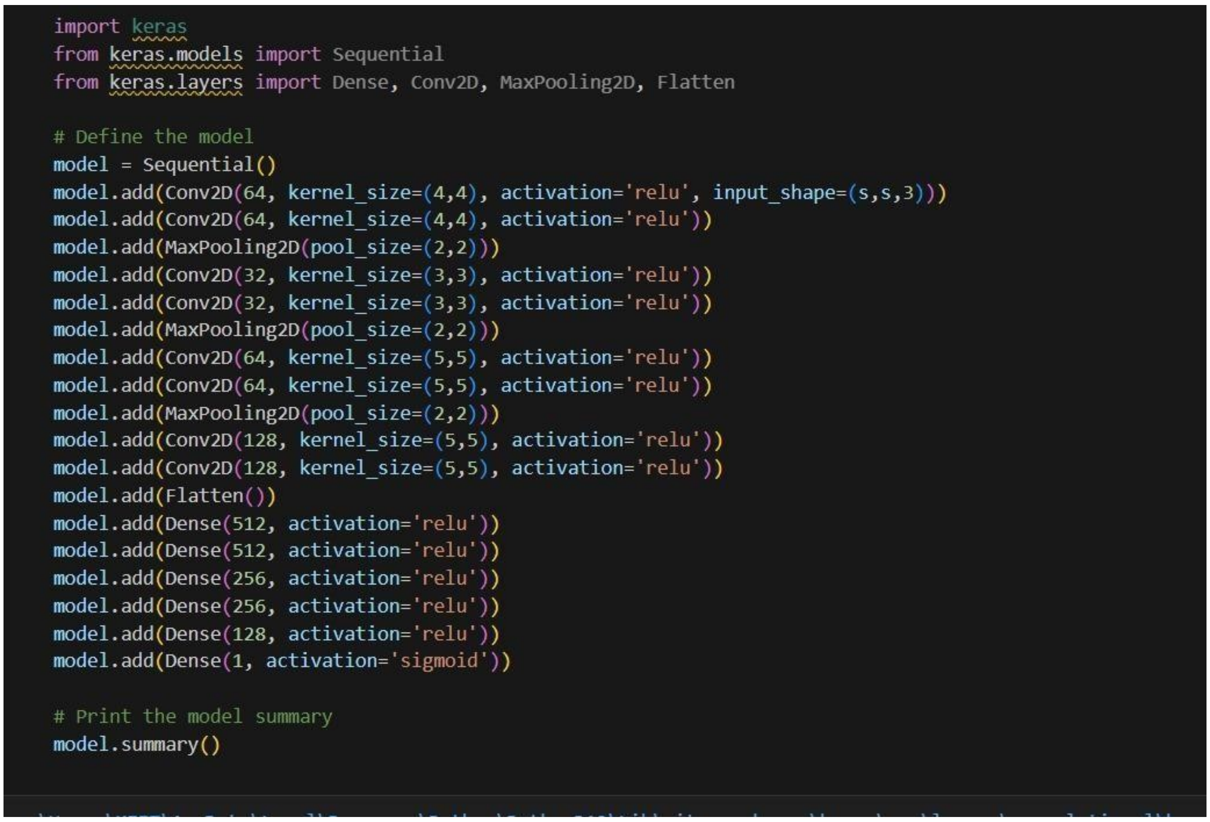
```
import matplotlib.pyplot as plt
import numpy as np

plt.figure(figsize=(25,25))
for n, i in enumerate(list(np.random.randint(0, len(X_train), 45))):
    plt.subplot(7, 7, n+1)
    plt.imshow(X_train[i])
    plt.axis('on')
    plt.title(getcode(y_train[i]))
plt.show()
```



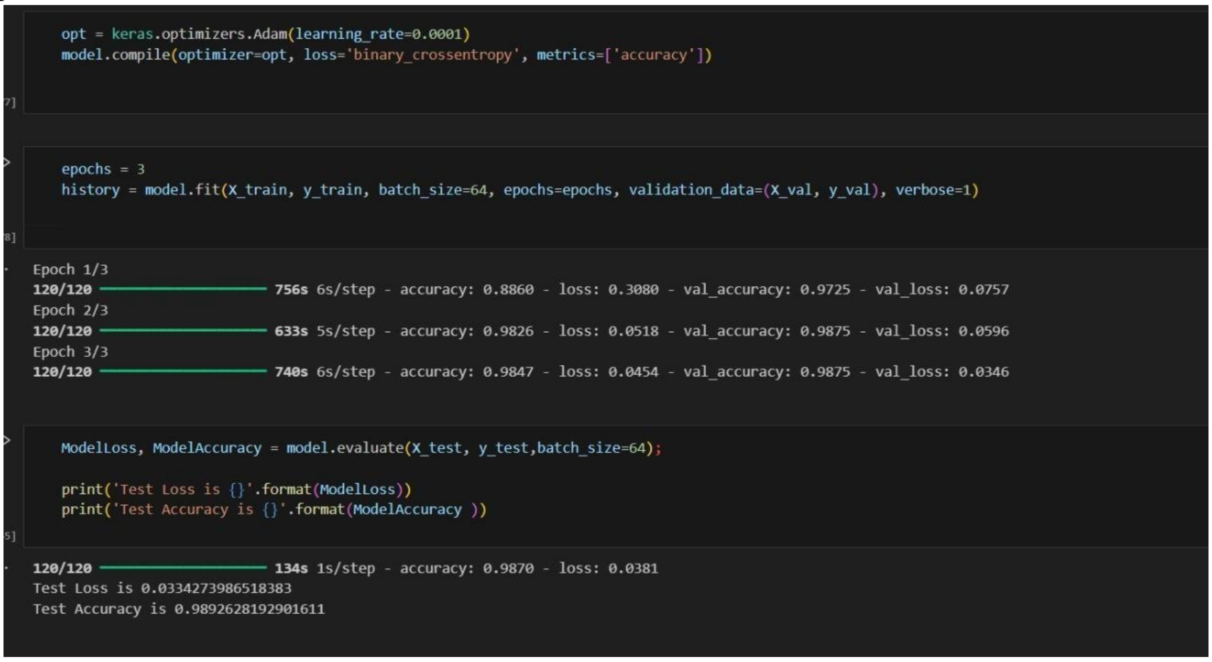


```python
import numpy as np
import random

#Training data
temp1 = list(zip(X_train, y_train))
random.shuffle(temp1)
X_train, y_train = zip(*temp1)
X_train, y_train = list(X_train), list(y_train)

X_train = np.array(X_train).astype(np.uint8)
y_train = np.array(y_train).astype(np.uint8)


#Validation data
temp2 = list(zip(X_val, y_val))
random.shuffle(temp2)
X_val, y_val = zip(*temp2)
X_val, y_val = list(X_val), list(y_val)

X_val = np.array(X_val).astype(np.uint8)
y_val = np.array(y_val).astype(np.uint8)



#Test data
temp3 = list(zip(X_test, y_test))
random.shuffle(temp3)
X_test, y_test = zip(*temp3)
X_test, y_test = list(X_test), list(y_test)

X_test = np.array(X_test).astype(np.uint8)
y_test = np.array(y_test).astype(np.uint8)
```

Keras API Interaction and add different layers of CNN



```python
import keras
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten

# Define the model
model = Sequential()
model.add(Conv2D(64, kernel_size=(4,4), activation='relu', input_shape=(s,s,3)))
model.add(Conv2D(64, kernel_size=(4,4), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu'))
model.add(Conv2D(32, kernel_size=(3,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(64, kernel_size=(5,5), activation='relu'))
model.add(Conv2D(64, kernel_size=(5,5), activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Conv2D(128, kernel_size=(5,5), activation='relu'))
model.add(Conv2D(128, kernel_size=(5,5), activation='relu'))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dense(512, activation='relu'))
model.add(Dense(256, activation='relu'))
model.add(Dense(256, activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid'))

# Print the model summary
model.summary()
```



```
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(optimizer=opt, loss='binary_crossentropy', metrics=['accuracy'])
```

```
epochs = 3
history = model.fit(X_train, y_train, batch_size=64, epochs=epochs, validation_data=(X_val, y_val), verbose=1)
```

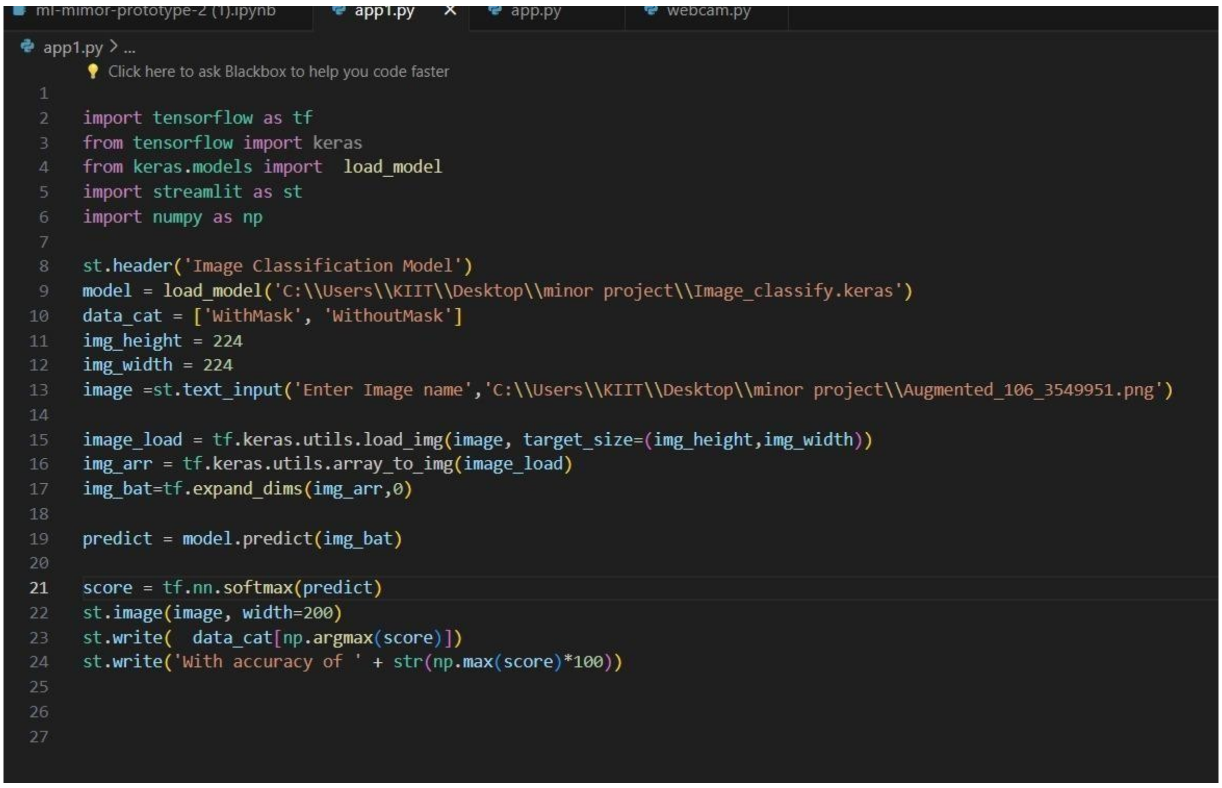
```
Epoch 1/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 756s 6s/step - accuracy: 0.8860 - loss: 0.3080 - val_accuracy: 0.9725 - val_loss: 0.0757
Epoch 2/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 633s 5s/step - accuracy: 0.9826 - loss: 0.0518 - val_accuracy: 0.9875 - val_loss: 0.0596
Epoch 3/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 740s 6s/step - accuracy: 0.9847 - loss: 0.0454 - val_accuracy: 0.9875 - val_loss: 0.0346
```

```
ModelLoss, ModelAccuracy = model.evaluate(X_test, y_test,batch_size=64);

print('Test Loss is {}'.format(ModelLoss))
print('Test Accuracy is {}'.format(ModelAccuracy ))
```

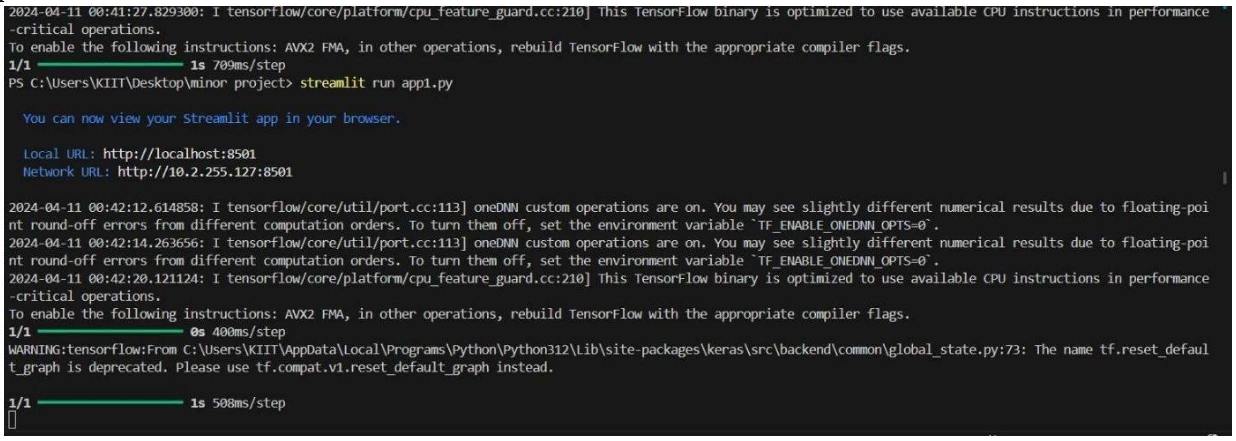
```
120/120 ━━━━━━━━━━━━━━━━━━━━ 134s 1s/step - accuracy: 0.9870 - loss: 0.0381
Test Loss is 0.0334273986518383
Test Accuracy is 0.9892628192901611
```

UI Design

In order to customize the system to particular surroundings and operating requirements, provide choices for modifying the detection sensitivity, allowing input for a desired length, setting alarm thresholds, and controlling user permissions.Establish a notification system that will promptly notify administrators or law enforcement when people are seen without masks. The system should provide information such as the location, time, and video feed to enable prompt action..


ml-mimor-prototype-2 (1).ipynb
app1.py
app.py
webcam.py
app1.py > ...
Click here to ask Blackbox to help you code faster

```python

import tensorflow as tf
from tensorflow import keras
from keras.models import  load_model
import streamlit as st
import numpy as np

st.header('Image Classification Model')
model = load_model('C:\\Users\\KIIT\\Desktop\\minor project\\Image_classify.keras')
data_cat = ['WithMask', 'WithoutMask']
img_height = 224
img_width = 224
image =st.text_input('Enter Image name','C:\\Users\\KIIT\\Desktop\\minor project\\Augmented_106_3549951.png')

image_load = tf.keras.utils.load_img(image, target_size=(img_height,img_width))
img_arr = tf.keras.utils.array_to_img(image_load)
img_bat=tf.expand_dims(img_arr,0)

predict = model.predict(img_bat)

score = tf.nn.softmax(predict)
st.image(image, width=200)
st.write(  data_cat[np.argmax(score)])
st.write('With accuracy of ' + str(np.max(score)*100))
```



```
2024-04-11 00:41:27.829300: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance
-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 709ms/step
PS C:\Users\KIIT\Desktop\minor project> streamlit run app1.py

  You can now view your Streamlit app in your browser.

  Local URL: http://localhost:8501
  Network URL: http://10.2.255.127:8501

2024-04-11 00:42:12.614858: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly different numerical results due to floating-poi
nt round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2024-04-11 00:42:14.263656: I tensorflow/core/util/port.cc:113] oneDNN custom operations are on. You may see slightly different numerical results due to floating-poi
nt round-off errors from different computation orders. To turn them off, set the environment variable `TF_ENABLE_ONEDNN_OPTS=0`.
2024-04-11 00:42:20.121124: I tensorflow/core/platform/cpu_feature_guard.cc:210] This TensorFlow binary is optimized to use available CPU instructions in performance
-critical operations.
To enable the following instructions: AVX2 FMA, in other operations, rebuild TensorFlow with the appropriate compiler flags.
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 400ms/step
WARNING:tensorflow:From C:\Users\KIIT\AppData\Local\Programs\Python\Python312\Lib\site-packages\keras\src\backend\common\global_state.py:73: The name tf.reset_defaul
t_graph is deprecated. Please use tf.compat.v1.reset_default_graph instead.

1/1 ━━━━━━━━━━━━━━━━━━━━ 1s 508ms/step
```

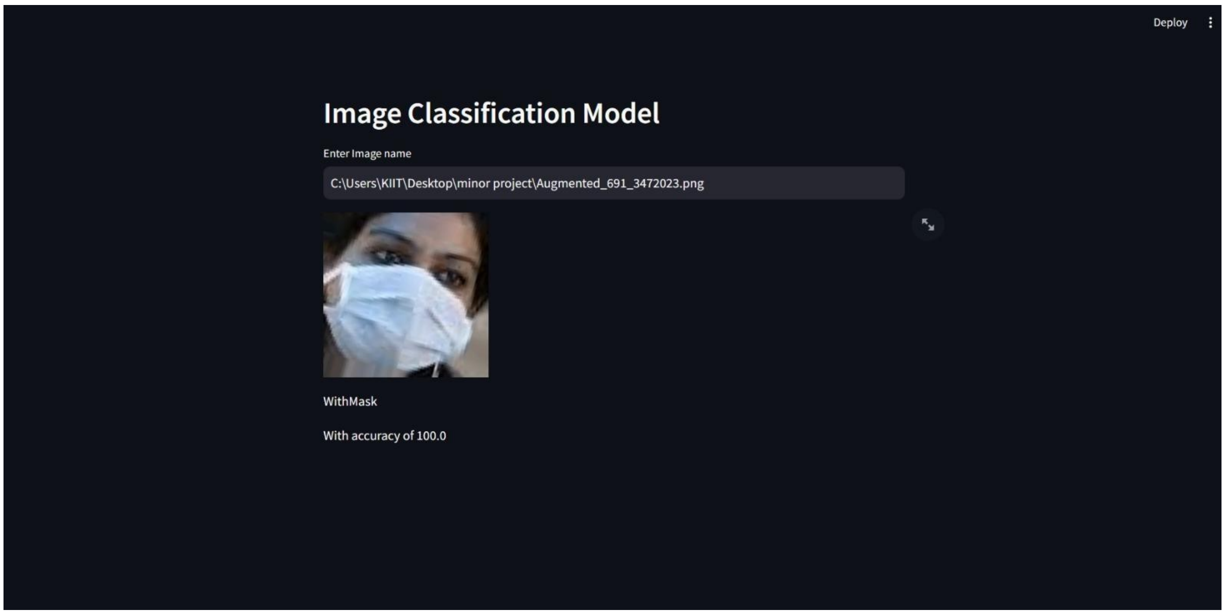


Image Classification Model

Enter Image name

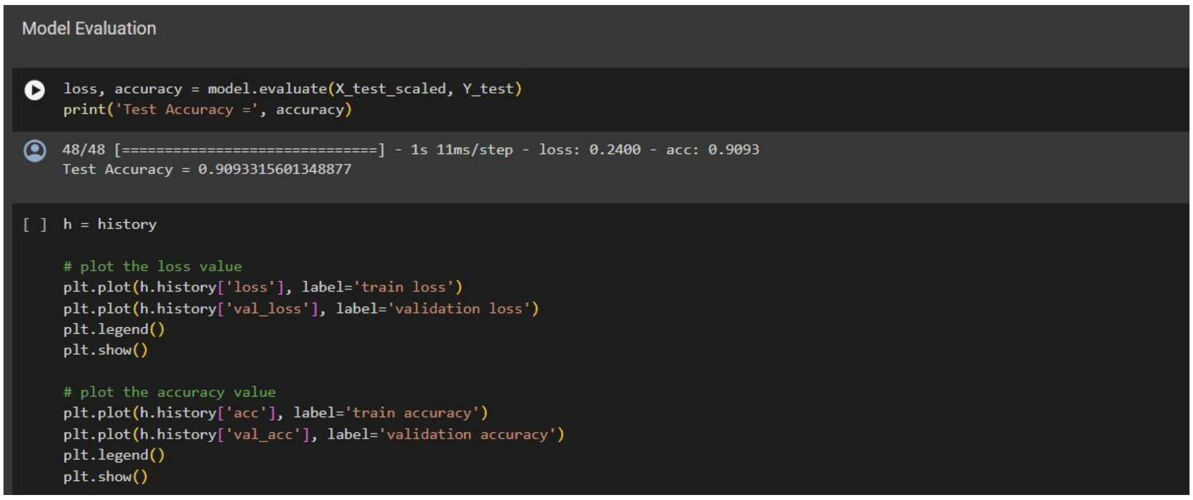
C:\Users\KIIT\Desktop\minor project\Augmented_691_3472023.png

WithMask

With accuracy of 100.0

4.2 Testing OR Verification Plan

4.2.1 Model Evaluation



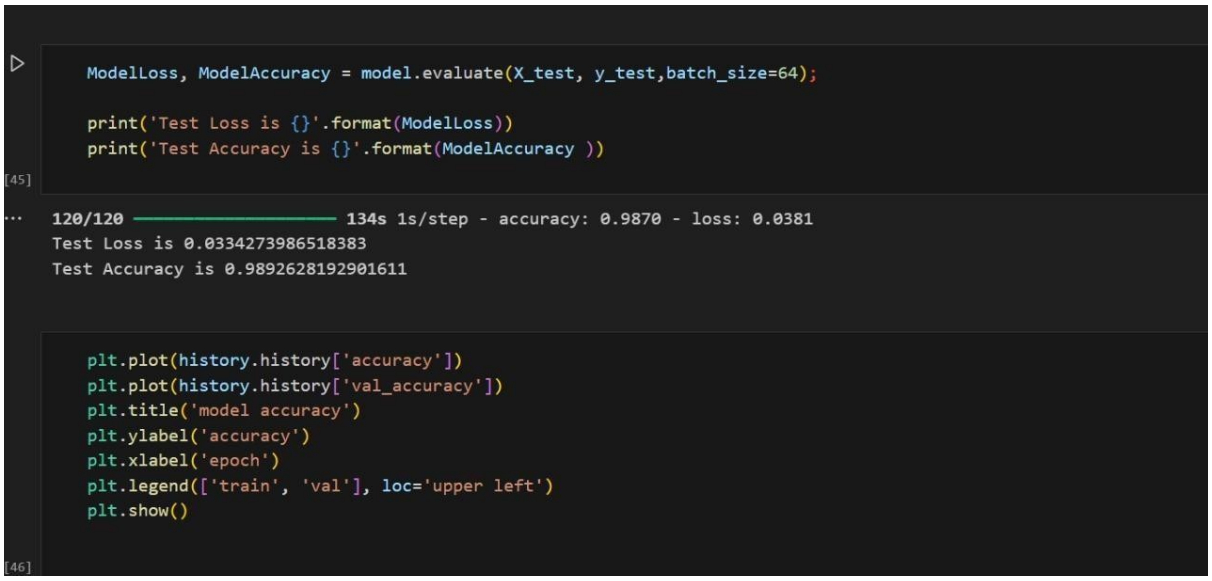
Model Evaluation

```
loss, accuracy = model.evaluate(X_test_scaled, Y_test)
print('Test Accuracy =', accuracy)
```

```
48/48 [==============================] - 1s 11ms/step - loss: 0.2400 - acc: 0.9093
Test Accuracy = 0.9093315601348877
```

```
h = history

# plot the loss value
plt.plot(h.history['loss'], label='train loss')
plt.plot(h.history['val_loss'], label='validation loss')
plt.legend()
plt.show()

# plot the accuracy value
plt.plot(h.history['acc'], label='train accuracy')
plt.plot(h.history['val_acc'], label='validation accuracy')
plt.legend()
plt.show()
```

4.3 Result Analysis OR Screenshots



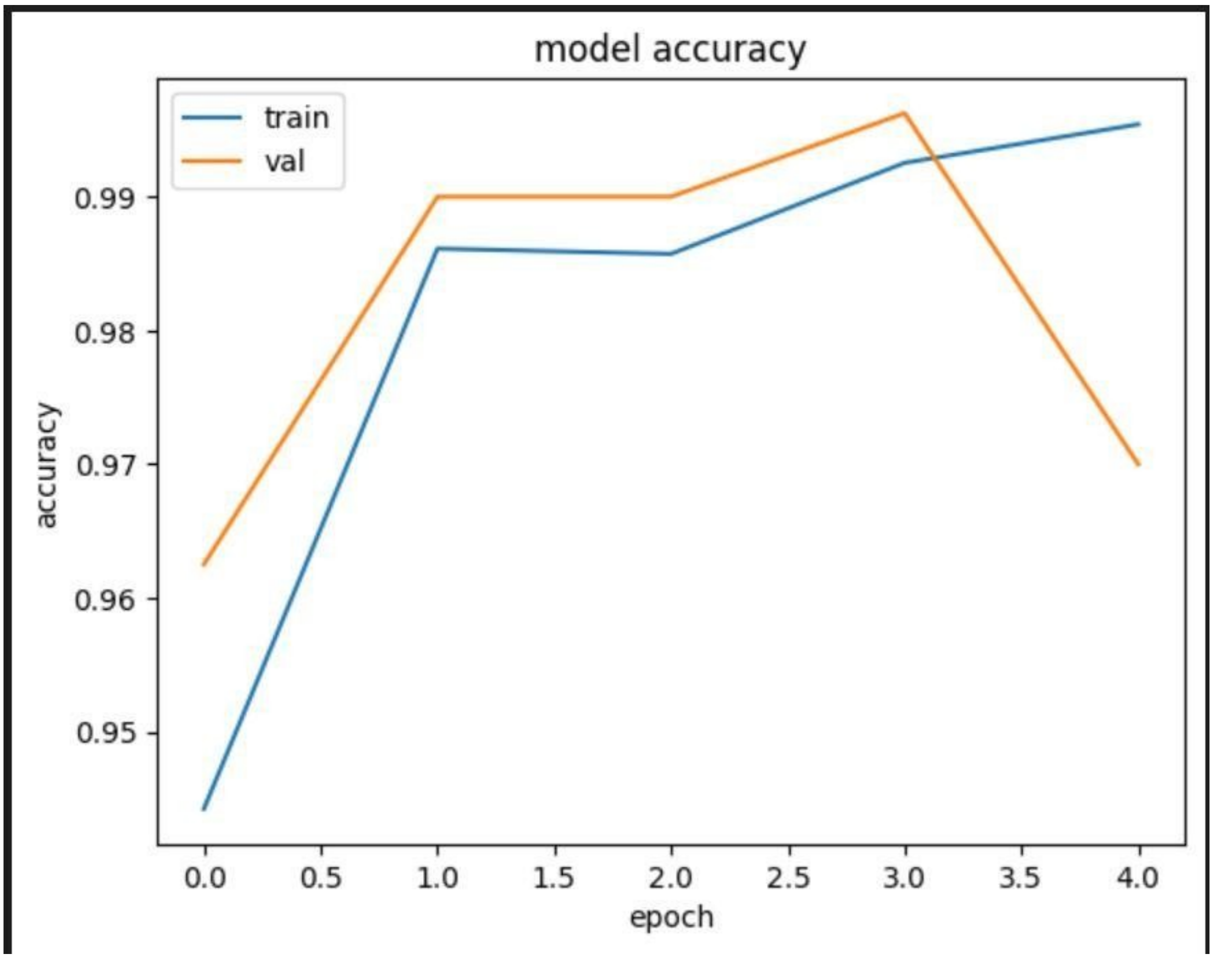
```
ModelLoss, ModelAccuracy = model.evaluate(X_test, y_test,batch_size=64);

print('Test Loss is {}'.format(ModelLoss))
print('Test Accuracy is {}'.format(ModelAccuracy ))
```

[45]

```
120/120 ━━━━━━━━━━━━━━━━━━━━ 134s 1s/step - accuracy: 0.9870 - loss: 0.0381
Test Loss is 0.0334273986518383
Test Accuracy is 0.9892628192901611
```

```
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'val'], loc='upper left')
plt.show()
```

[46]



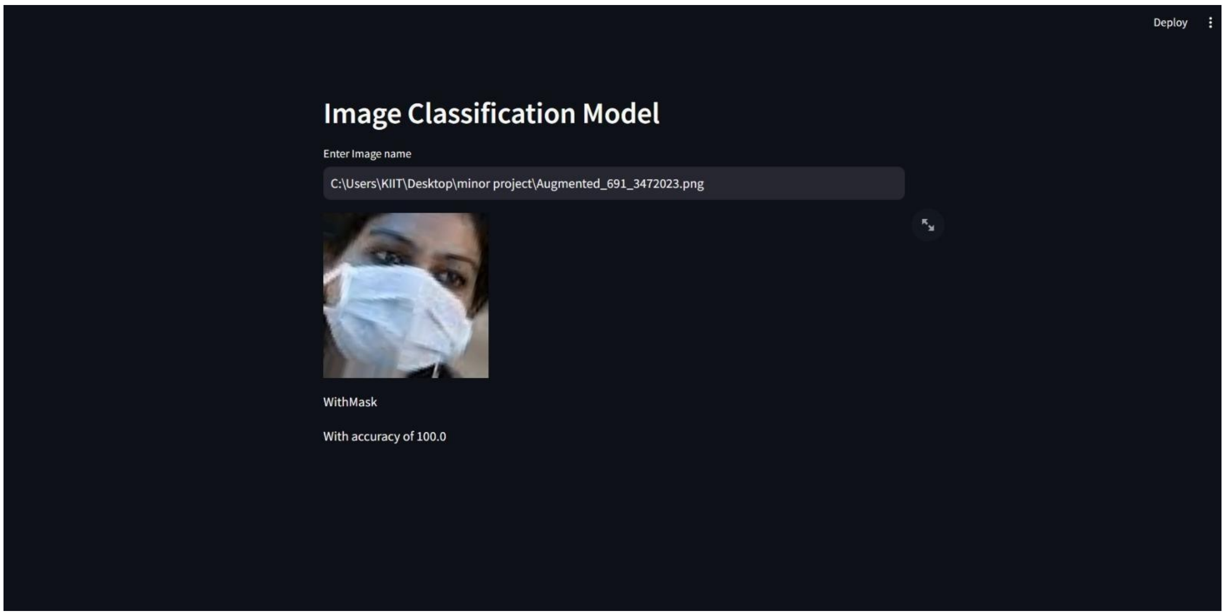


Image Classification Model

Enter Image name

C:\Users\KIIT\Desktop\minor project\Augmented_691_3472023.png

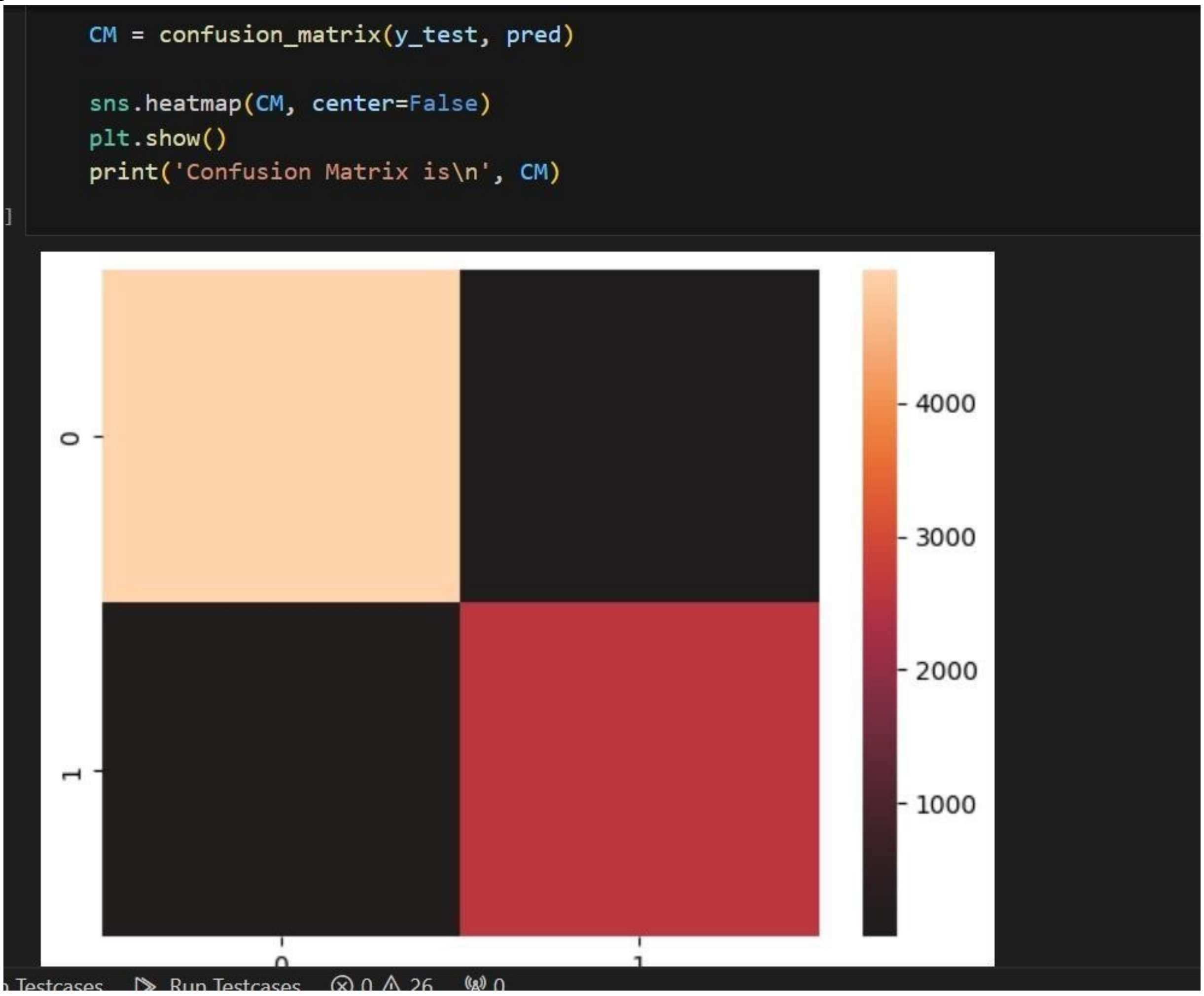
WithMask

With accuracy of 100.0

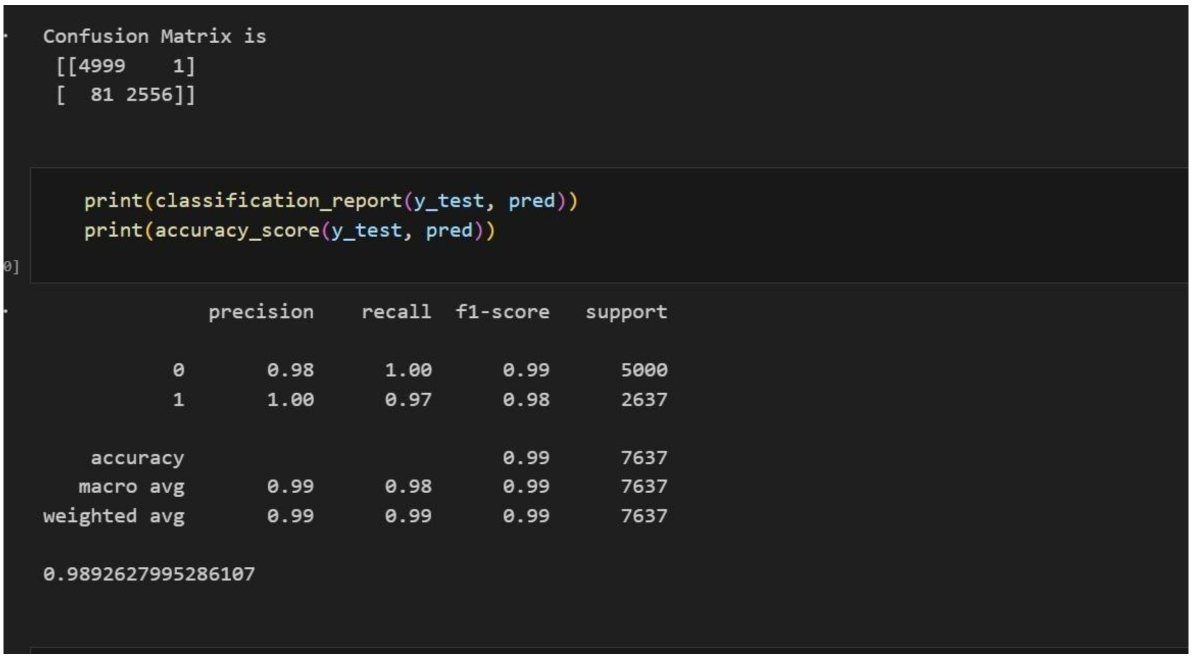
4.4 Quality Assurance

Code Quality:

Code Reviews: Frequent team or peer code reviews with an emphasis on standard compliance, brevity, and possible bug prevention.
Lintig and Static Analysis: Include tools to identify any mistakes and style problems early in the development process..
Test Coverage: Over time, try to raise the proportion of code that is tested by unit and integration tests.
Performance measure: Additionally, we use the confusion matrix to evaluate the model's performance using the recall, precision, f1-score, and support.



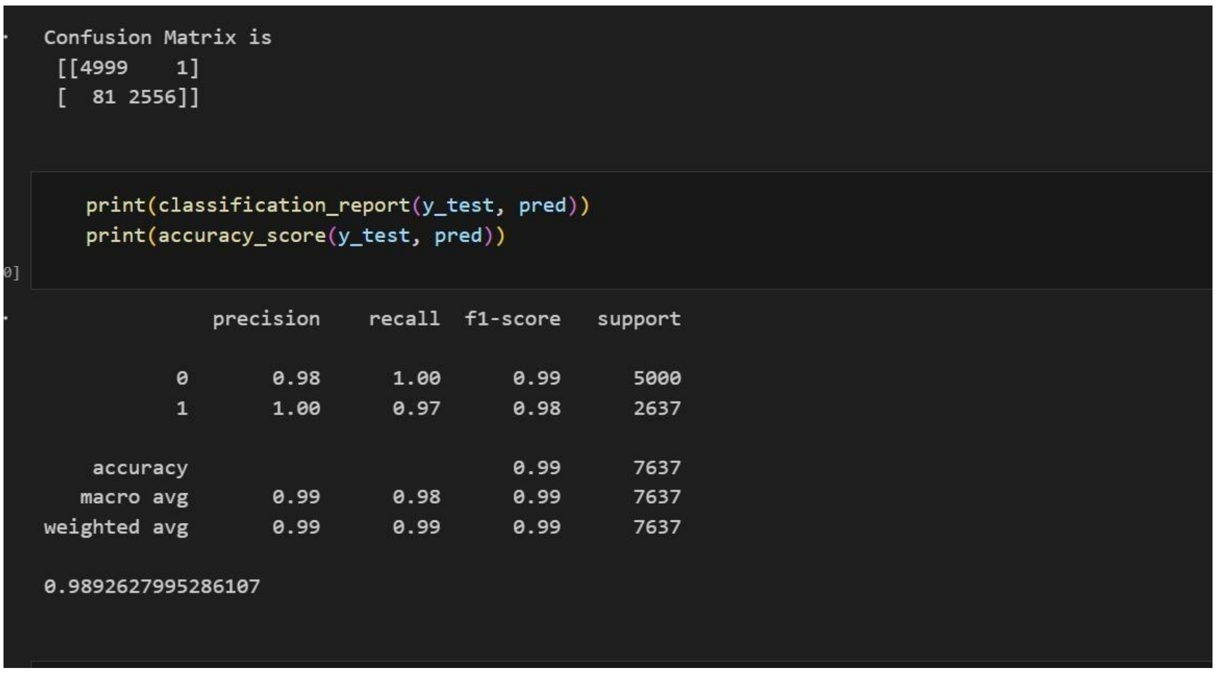
```python
CM = confusion_matrix(y_test, pred)

sns.heatmap(CM, center=False)
plt.show()
print('Confusion Matrix is\n', CM)
```



```
Confusion Matrix is
 [[4999    1]
 [  81 2556]]
```

```python
print(classification_report(y_test, pred))
print(accuracy_score(y_test, pred))
```

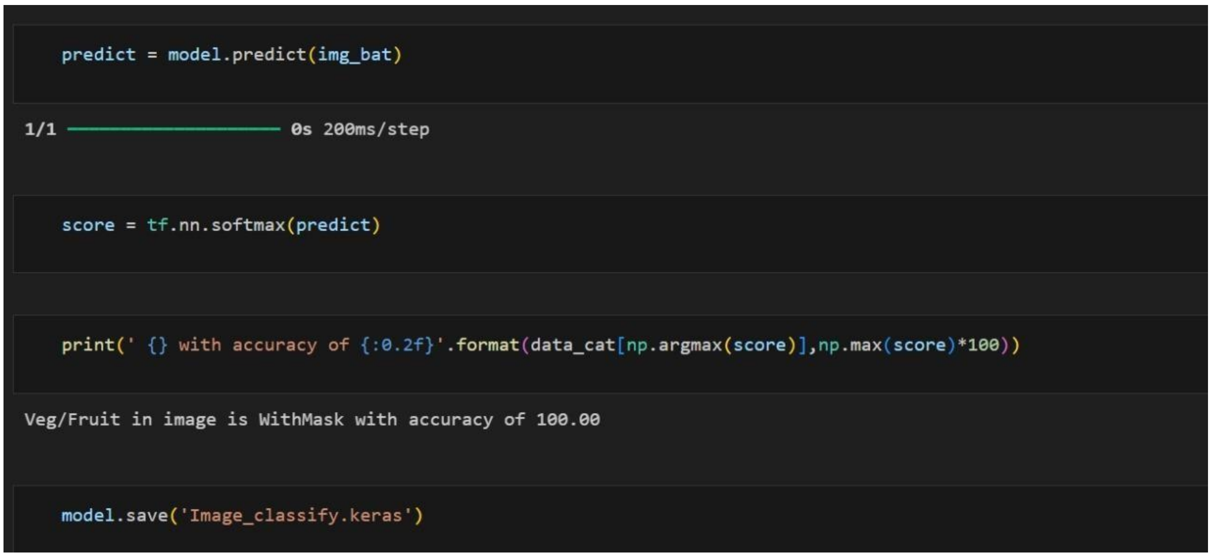
```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      5000
           1       1.00      0.97      0.98      2637

    accuracy                           0.99      7637
   macro avg       0.99      0.98      0.99      7637
weighted avg       0.99      0.99      0.99      7637

0.9892627995286107
```



```
Confusion Matrix is
 [[4999    1]
 [  81 2556]]
```

```
print(classification_report(y_test, pred))
print(accuracy_score(y_test, pred))
```

```
              precision    recall  f1-score   support

           0       0.98      1.00      0.99      5000
           1       1.00      0.97      0.98      2637

    accuracy                           0.99      7637
   macro avg       0.99      0.98      0.99      7637
weighted avg       0.99      0.99      0.99      7637

0.9892627995286107
```



```
predict = model.predict(img_bat)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 200ms/step
```

```
score = tf.nn.softmax(predict)
```

```
print(' {} with accuracy of {:0.2f}'.format(data_cat[np.argmax(score)],np.max(score)*100))
```

```
Veg/Fruit in image is WithMask with accuracy of 100.00
```

```
model.save('Image_classify.keras')
```

Chapter 5

Standards Adopted

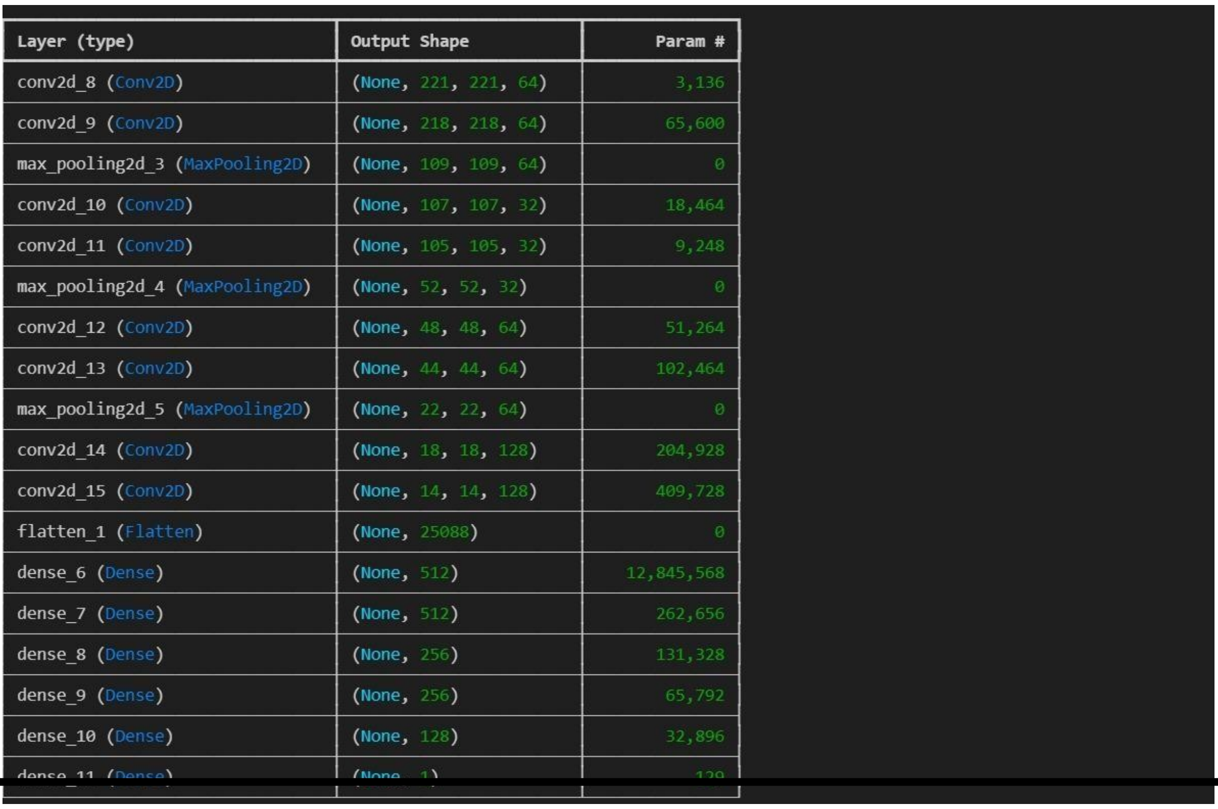
5.1 Design Standards

Accuracy and Reliability: Making sure algorithms consistently identify face masks in a variety of scenarios in order to protect public safety.
System Architecture: It entails using cameras to take pictures in real-time, processing them using deep learning models to detect masks, and then globally disseminating the results to a user interface for monitoring and alerting.
User Interface Accessibility API Integration: Creating intuitive user interfaces to promote smooth communication and enable effective detection system management and monitoring.

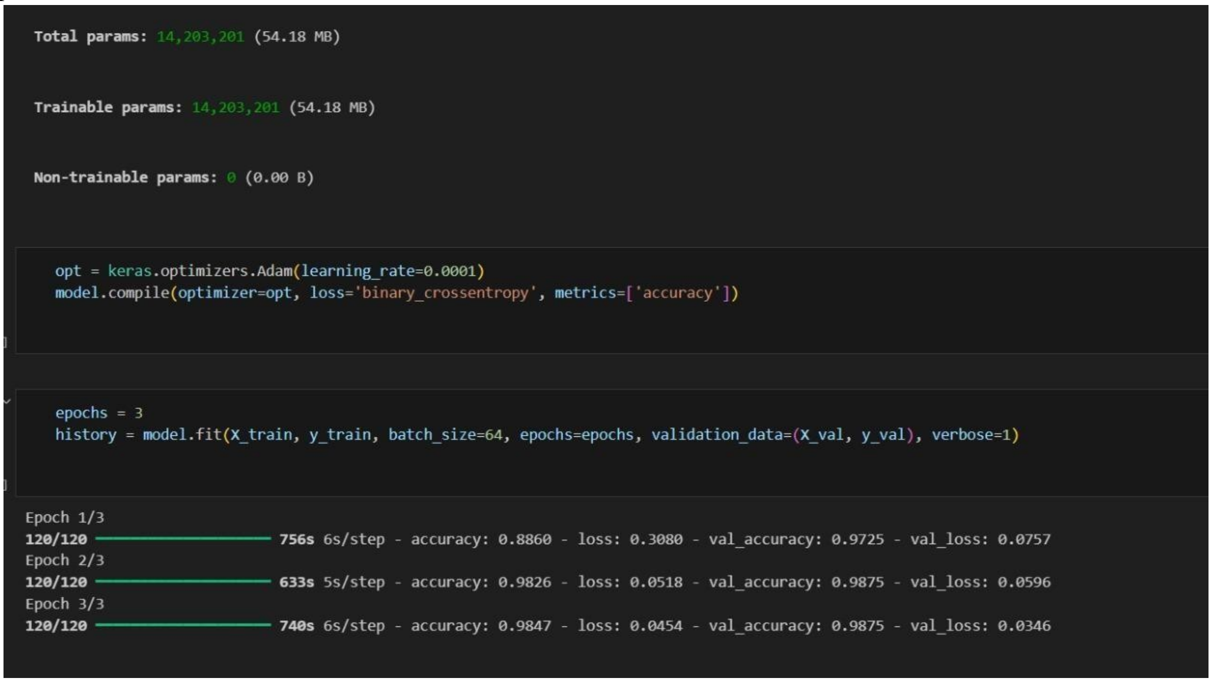
5.2 Coding Standards

Standard coding practices have been followed throughout the project and use of Proper comments and logically naming the variables have been done.

Using CNN : Employing many CNN layers to learn pertinent features from photos, allowing them to successfully extract discriminating features from face images, which helps with mask recognition.



```
Layer (type)                    Output Shape              Param #
conv2d_8 (Conv2D)               (None, 221, 221, 64)        3,136
conv2d_9 (Conv2D)               (None, 218, 218, 64)       65,600
max_pooling2d_3 (MaxPooling2D)  (None, 109, 109, 64)            0
conv2d_10 (Conv2D)              (None, 107, 107, 32)       18,464
conv2d_11 (Conv2D)              (None, 105, 105, 32)        9,248
max_pooling2d_4 (MaxPooling2D)  (None, 52, 52, 32)              0
conv2d_12 (Conv2D)              (None, 48, 48, 64)         51,264
conv2d_13 (Conv2D)              (None, 44, 44, 64)        102,464
max_pooling2d_5 (MaxPooling2D)  (None, 22, 22, 64)              0
conv2d_14 (Conv2D)              (None, 18, 18, 128)       204,928
conv2d_15 (Conv2D)              (None, 14, 14, 128)       409,728
flatten_1 (Flatten)             (None, 25088)                   0
dense_6 (Dense)                 (None, 512)            12,845,568
dense_7 (Dense)                 (None, 512)               262,656
dense_8 (Dense)                 (None, 256)               131,328
dense_9 (Dense)                 (None, 256)                65,792
dense_10 (Dense)                (None, 128)                32,896
dense_11 (Dense)                (None, 1)                     129
```



```
Total params: 14,203,201 (54.18 MB)

Trainable params: 14,203,201 (54.18 MB)

Non-trainable params: 0 (0.00 B)
```

```
opt = keras.optimizers.Adam(learning_rate=0.0001)
model.compile(optimizer=opt, loss='binary_crossentropy', metrics=['accuracy'])
```

```
epochs = 3
history = model.fit(X_train, y_train, batch_size=64, epochs=epochs, validation_data=(X_val, y_val), verbose=1)
```

```
Epoch 1/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 756s 6s/step - accuracy: 0.8860 - loss: 0.3080 - val_accuracy: 0.9725 - val_loss: 0.0757
Epoch 2/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 633s 5s/step - accuracy: 0.9826 - loss: 0.0518 - val_accuracy: 0.9875 - val_loss: 0.0596
Epoch 3/3
120/120 ━━━━━━━━━━━━━━━━━━━━ 740s 6s/step - accuracy: 0.9847 - loss: 0.0454 - val_accuracy: 0.9875 - val_loss: 0.0346
```

Python libraries used : Various Python libraries, including sklearn, matplot, numpy, and json, are used.

Keras API interaction : interacting with the Keras API to build the model using the proper optimizers, loss functions, and assessment metrics. After that, the model is trained using labeled data (pictures of faces with and without masks), and its effectiveness is assessed using validation data from interactions with other AI sources.

5.3 Testing Standards

Unit Testing: In order to ensure that specific components—like image pre-processing algorithms and model inference techniques—produce the intended results and increase the overall system's dependability and robustness, test cases must be created for each component.
Integration Testing: To guarantee smooth operation and precise detection throughout the system, it entails verifying the compatibility and interactions of numerous system components, including picture capture, deep learning model inference, and warning systems.
User Acceptance Testing: incorporates user input into testing to verify scenarios.

Chapter 6

Conclusion and Future Scope

6.1 Conclusion

This paper describes a face mask detection system that can automatically determine whether a person is wearing a mask in both static images and videos. The project's scope can be expanded to include real-time footage as well. This technique is an excellent means of preventing the spread of the COVID-19 pandemic. Using CNN, OpenCV, and Keras, the proposed system can identify whether or not a face mask is present. The model produces fast and accurate results.
The accuracy of the trained model is approximately 98%.

The proposed model performs better in terms of accuracy and processing time than DenseNet-121, MobileNet-V2, VGG-19, and Inception-V3, according to experiments comparing it to various pre-models. This methodology is a fantastic choice for a real-time monitoring system because of its accuracy and computing efficiency.

References

- https://doi.org/10.1007/978-3-319-46454-1_8
- https://doi.org/10.1109/JSEN.2021.3061178
- https://doi.org/10.1109/CVPR.2016.90
- https://ieeexplore.ieee.org/document/10085276
- https://data-flair.training/blogs/face-mask-detection-with-python/
- https://www.sciencedirect.com/science/article/pii/S15320464210075

INDIVIDUAL CONTRIBUTION REPORT:

FACE MASK DETECTION USING CONVOLUTIONAL NEURAL NETWORK

Ajit
2006161

Abstract: Our project, Face Mask Detection, automatically recognizes persons wearing masks in real time using computer vision algorithms and deep learning. Through the analysis of live video streams, our technique accurately distinguishes between masked and unmasked faces, supporting public health efforts during the COVID-19 pandemic. Using convolutional neural networks and image processing techniques, we achieve high detection speed and precision, which facilitates mask rule compliance in a range of contexts. This technology offers a scalable way to monitor and enforce laws requiring the wearing of masks in public spaces.

Individual contribution and findings:
Role in project group: As a member of the project group, my primary resposibility is to performance check of model with confusion matrics and adding different layers of CNN to the model for optimization of images.

Performance check: **True Positive (TP)**: The model accurately predicts the presence of a mask on a person.
True Negative (TN): It is accurately predicted by the model that the subject is not donning a mask.
False Positive (FP): When someone isn't wearing a mask, the model wrongly assumes they are.
False Negative (FN): When someone is wearing a mask, the model mistakenly assumes that they are not.
The efficacy of the face mask identification system can be assessed by computing multiple performance metrics, including accuracy, precision, recall, and F1-score, using this matrix.

Individual contribution to project report preparation: In project report, I significantly contributed to chapter 5 and part of confusion matrices:
Ethical considerations: It include preventing prejudice and discrimination and guaranteeing objectivity and justice in performance evaluation.
Cross-validation: Assuring the model's performance is resilient across several data subsets.
Accuracy of Training and Testing: Assessing overfitting or underfitting by contrasting results on datasets used for training and testing.
Sensitivity analysis: It evaluates the impact of modifications to data or model parameters on performance indicators.

Individual contribution for project presentation and demonstration: My particular contribution to the project presentation and CNN face mask detection demonstration was the architecture design and implementation of the convolutional neural network. I developed the model to effectively distinguish between masked and unmasked faces using features that I took out of the pictures. In order to optimize performance, I also changed the hyperparameters. During the demonstration, I showed how accurate and effective the model is in real-time detection, highlighting its potential for practical application in a range of scenarios to improve public health and safety.

Full Signature of Supervisor: ……………………………

Full signature of the student: ……………………………

INDIVIDUAL CONTRIBUTION REPORT:

FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK

AYUSH RAJ
2006120

Abstract: Our project, Face Mask Detection, automatically recognizes persons wearing masks in real time using computer vision algorithms and deep learning. Through the analysis of live video streams, our technique accurately distinguishes between masked and unmasked faces, supporting public health efforts during the COVID-19 pandemic. Using convolutional neural networks and image processing techniques, we achieve high detection speed and precision, which facilitates mask rule compliance in a range of contexts. This technology offers a scalable way to monitor and enforce laws requiring the wearing of masks in public spaces.

Individual contribution and findings: I Designed thew UI/UX part of the project. This part of the code is a Streamlit application that uses a pre-trained image classification model to predict whether an input image contains a person wearing a mask or not.The UI/UX part also provides options for adjusting detection sensitivity, input for desired duration, configuring alert thresholds, and managing user permissions to tailor the system to specific environments and operational requirements. It also Implements a notification system to alert administrators or authorities instantly when individuals are detected without masks, providing details such as location, timestamp, and camera feed for quick response.

Individual contribution to project report preparation: I Prepared the UI Design part of the project report. This part of the project report is located in Chapter 4 titled implementation and portion 4.1 Methodology OR Proposal.

Individual contribution for project presentation and demonstration: I prepared the UI Design part of the project presentation and and demonstrated the UI/UX part of the project.

Full Signature of Supervisor:
…………………………….

Full signature of the student:
…………………………….

INDIVIDUAL CONTRIBUTION REPORT:

FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK

Meetali Verma
2006224

Abstract: Our project, Face Mask Detection, automatically recognizes persons wearing masks in real time using computer vision algorithms and deep learning. Through the analysis of live video streams, our technique accurately distinguishes between masked and unmasked faces, supporting public health efforts during the COVID-19 pandemic. Using convolutional neural networks and image processing techniques, we achieve high detection speed and precision, which facilitates mask rule compliance in a range of contexts. This technology offers a scalable way to monitor and enforce laws requiring the wearing of masks in public spaces.

Individual contribution and findings: Individual Contribution and Findings: In the project focused on face mask detection, my primary contribution involved designing and implementing various CNN architectures using Keras interaction API. I experimented with different configurations, including layer depths and activation functions, to optimize model performance. Utilizing Keras' interaction API facilitated seamless integration of layers, enhancing model accuracy. Extensive experimentation and hyperparameter tuning improved convergence and robustness across datasets. Deeper architectures with multiple layers proved effective in capturing intricate features. Fine-tuning parameters like learning rate enhanced training efficiency. The project highlighted the effectiveness of CNN-based approaches for face mask detection.

Individual contribution to project report preparation: For the project report, I documented the research methodology and implementation details comprehensively. I summarized insights gained from experimentation, emphasizing the significance of architectural choices on model performance. Collaborating with team members, I structured the report coherently and highlighted potential applications and future directions.

Individual contribution for project presentation and demonstration: In the presentation, I effectively communicated the technical aspects of CNN models for face mask detection. Through visually engaging slides and live demonstrations, I showcased the models' functionality and performance in real-time scenarios. During Q&A sessions, I addressed queries related to implementation and potential extensions, engaging the audience actively.

Full Signature of Supervisor: …………………………….

Full signature of the student: …………………………….

INDIVIDUAL CONTRIBUTION REPORT:

FACE MASK DETECTION SYSTEM USING CONVOLUTIONAL NEURAL NETWORK

Happy Ranjan
2006432

Abstract: Our project, Face Mask Detection, automatically recognizes persons wearing masks in real time using computer vision algorithms and deep learning. Through the analysis of live video streams, our technique accurately distinguishes between masked and unmasked faces, supporting public health efforts during the COVID-19 pandemic. Using convolutional neural networks and image processing techniques, we achieve high detection speed and precision, which facilitates mask rule compliance in a range of contexts. This technology offers a scalable way to monitor and enforce laws requiring the wearing of masks in public spaces.

Individual contribution and findings: In our project focusedon face mask detection I developed the main Development process , importing the implemented libraries , dependencies.In my work I joined the data set to the implemented model. After this I access the data set through os.join.path and glob.glob.then access the sub classes, initialized the train,validation and test data and create labels for training our model. While creating and initializing the data the resize the data . After this showing the mixed random data using enumerate.list.np.random.randint with axis on . At last create 3 different zip files and then convert this data into 8 byte integer.

Individual contribution to project report preparation: For the project report, I documented the project planing and the implementation of the planing layout respectively. I documented the Detailed Problem Statement,Workflow fragmentation,Real-time Detection and User interface. I define and documented the development coding implementation and also the testing the code with documentation.Collaborating with team members, I structured the report coherently and highlighted potential technical aspect .

Individual contribution for project presentation and demonstration: In the presentation, I productively communicated the technical terminologies about developing the model's base development in our face mask detection project. Through visually engaging slides and live presentation I showcase the functionality and the impotence of the main stream technology used in this project . .During Q&A sessions, I addressed queries related to methods,parameters and terminology for engaging the audience actively.

Full Signature of Supervisor: ……………………….

Full signature of the student: ……………………………