Report On

Face Mask Detector using CNN

Submitted in partial fulfillment of the requirements of the Course project in Semester VIII of Final Year Artificial Intelligence and Data Science

by

Sakshi Mhatre (Roll No. 13)
Amrita Nair (Roll No. 16)
Prachi Kadam (Roll No. 33)

Supervisor
Dr. Tatwadarshi P. N

University of Mumbai

Vidyavardhini's College of Engineering & Technology

Department of Artificial Intelligence and Data Science

(2023-24)

Vidyavardhini's College of Engineering & Technology
Department of Artificial Intelligence and Data Science

CERTIFICATE

This is to certify that the project entitled “Face Mask Detector using CNN” is a bonafide work of" Sakshi Mhatre (Roll No. 13), Amrita Nair (Roll No. 16), Prachi Kadam (Roll No. 33)" submitted to the University of Mumbai in partial fulfillment of the requirement for the Course project in semester VIII of Final Year Artificial Intelligence and Data Science engineering.

Supervisor
Dr. Tatwadarshi P. N.

Dr. Tatwadarshi P. N.
Head of Department

Abstract

This project introduces a Face Mask Detection System using deep learning techniques to enhance public safety during contagious outbreaks. The system employs a pre-trained convolutional neural network (CNN) to accurately identify faces with or without masks. Extensive training data encompassing diverse demographics, lighting conditions, and poses ensures robust performance. The system provides real-time feedback through an intuitive interface, facilitating integration with various devices. Designed for seamless deployment in public spaces, the system assists in enforcing mask mandates. It alerts authorities or individuals when mask non-compliance is detected. The technology's versatility and effectiveness are validated through rigorous experiments. With applications in airports, hospitals, and crowded environments, the system plays a pivotal role in collective protection during pandemics and similar health crises. Its implementation represents a significant stride toward safeguarding public health and safety.

Table of Contents

Chapter 1

1.1 Problem Statement:

In the post-COVID era, ensuring public health and safety remains paramount. Adherence to mask-wearing mandates is critical in preventing potential outbreaks. However, manual monitoring of mask compliance in crowded spaces is resource-intensive and poses risks to personnel. Existing solutions often lack accuracy, speed, or seamless integration with surveillance systems. Moreover, they may not be optimized for diverse post-pandemic environments. The need for an automated system capable of swiftly and accurately identifying individuals wearing or not wearing masks is pressing. This project aims to develop a state-of-the-art Face Mask Detection System, leveraging advanced deep learning techniques. The system must achieve high accuracy and provide real-time alerts for non-compliance. It should be adaptable to various settings and easily integrate with existing hardware configurations. By automating mask compliance monitoring, this system will alleviate the burden on human resources and ensure a safer environment for all. It represents a crucial step towards safeguarding public health in the aftermath of the COVID-19 pandemic.

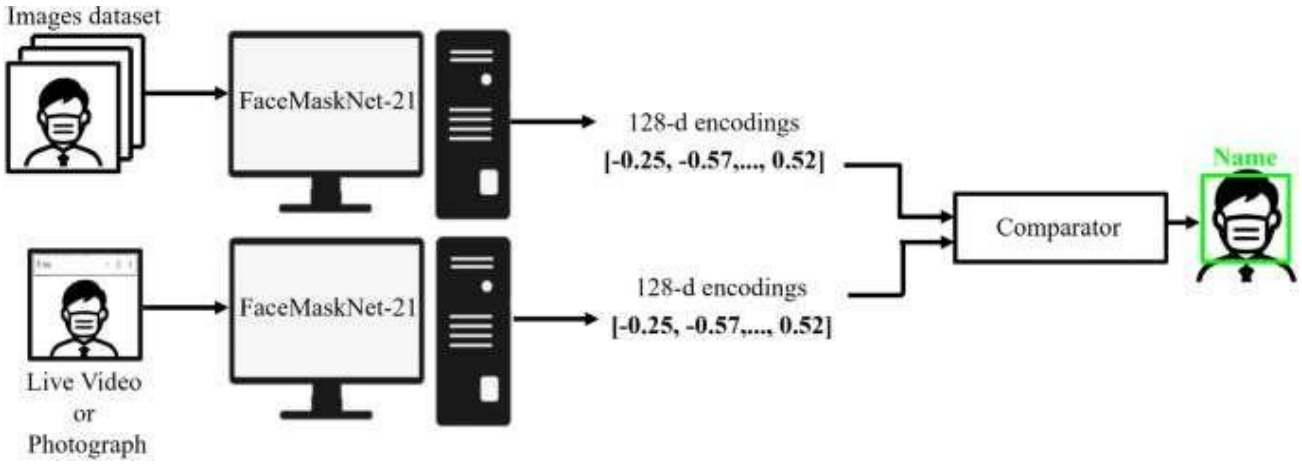
Chapter 2

2.1 Description and Working:

The Post-COVID Face Mask Compliance Monitoring System is an innovative solution designed to ensure public safety in the aftermath of the pandemic. It leverages cutting-edge deep learning technology to accurately detect and classify individuals based on their mask-wearing status in real-time. This system addresses the challenge of enforcing mask mandates in crowded public spaces, providing an automated and efficient alternative to manual monitoring.

The system employs a pre-trained convolutional neural network (CNN), a type of deep learning model, which is adept at understanding visual patterns. It processes live or recorded video feeds from surveillance cameras. As frames of the video stream are captured, the CNN analyzes each frame to identify human faces. Once a face is detected, the system focuses on the region of interest, i.e., the face.

Using the learned features, the CNN then determines whether the individual is wearing a mask or not. This classification decision is made with high accuracy, owing to the extensive training data and advanced neural network architecture. In real-time, the system updates its assessment with each new frame, ensuring swift and accurate identification.



2.2 Software & Hardware used:

Software:

- Visual Studio Code
- Python 3.11
- Windows 10 OS
- Google Colab

Hardware:

- 64 bit Operating System
- 6gb RAM
- Intel i5 processor

Chapter # 3

3.1 Code:

In [1]:
```
import tensorflow as tf
import cv2,os
import numpy as np
import keras

data_path=r'C:\Users\Amrita\Downloads\mini project\dataset'
categories=os.listdir(data_path)
labels=[i for i in range(len(categories))]

label_dict=dict(zip(categories,labels))

print(label_dict)
print(categories)
print(labels)
```

```
{'with mask': 0, 'without mask': 1}
['with mask', 'without mask']
[0, 1]
```

In [ ]:
```
img_size=100
data=[]
target=[]

for category in categories:
    folder_path=os.path.join(data_path,category)
    img_names=os.listdir(folder_path)

    for img_name in img_names:
        img_path=os.path.join(folder_path,img_name)
        img=cv2.imread(img_path)

        try:
            gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
            #Converting the image into gray scale
            resized=cv2.resize(gray,(img_size,img_size))
            #resizing the gray scale into 100x100, since we need a fixed common size for all the images in the dataset
            data.append(resized)
            target.append(label_dict[category])
            #appending the image and the label(categorized) into the list (dataset)

        except Exception as e:
            print('Exception:',e)
            #if any exception raised, the exception will be printed here. And pass to the next image
```

In [3]:
```
import numpy as np
from sklearn.preprocessing import LabelBinarizer

from keras.utils import np_utils
data=np.array(data)/255.0
data=np.reshape(data,(data.shape[0],img_size,img_size,1))

lb= LabelBinarizer()
target=lb.fit_transform(target)
target=np_utils.to_categorical(target)

target=np.array(target)
```

In [10]:
```
from keras.models import Sequential
from keras.layers import Dense,Activation,Flatten,Dropout
from keras.layers import Conv2D,MaxPooling2D

num_classes=2

model=Sequential()
model.call = tf.function(model.call)

model.add(Conv2D(64,(3,3),input_shape=(img_size,img_size,1)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The first CNN layer followed by Relu and MaxPooling layers

model.add(Conv2D(128,(3,3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
#The second convolution layer followed by Relu and MaxPooling layers

model.add(Flatten())
model.add(Dropout(0.5))

model.add(Dense(64,activation='relu'))
#Dense layer of 64 neurons
model.add(Dense(num_classes,activation='softmax'))
#The Final layer with two outputs for two categories
```

In [13]:
```
from tensorflow.keras.optimizers import Adam
from keras.callbacks import ModelCheckpoint
```

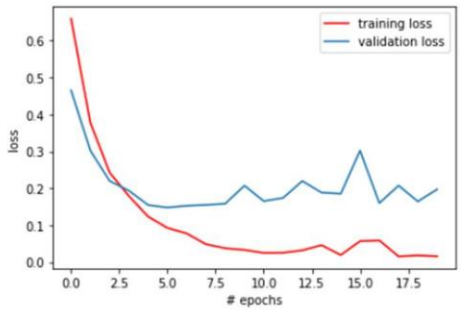
In [14]:
```
from sklearn.model_selection import train_test_split

train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1,random_state=0)
```

In [15]:
```
tf.compat.v1.disable_eager_execution()
model.compile(loss='categorical_crossentropy',optimizer=Adam(learning_rate=0.001),metrics=['accuracy'])
checkpoint = ModelCheckpoint('model-{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True,mode='auto')
history=model.fit(train_data,train_target,callbacks=[checkpoint],epochs=20,validation_split=0.25)
```

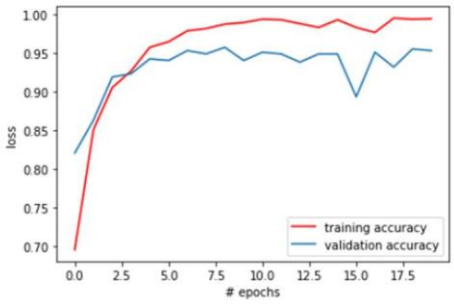
```
acy: 0.9487
Epoch 15/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0188 - accuracy: 0.9929 - val_loss: 0.1850 - val_accur
acy: 0.9487
Epoch 16/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0565 - accuracy: 0.9829 - val_loss: 0.3020 - val_accur
acy: 0.8932
Epoch 17/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0584 - accuracy: 0.9765 - val_loss: 0.1595 - val_accur
acy: 0.9509
Epoch 18/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0150 - accuracy: 0.9950 - val_loss: 0.2071 - val_accur
acy: 0.9316
Epoch 19/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0177 - accuracy: 0.9936 - val_loss: 0.1640 - val_accur
acy: 0.9551
Epoch 20/20
1403/1403 [==============================] - 18s 13ms/sample - loss: 0.0154 - accuracy: 0.9943 - val_loss: 0.1968 - val_accur
acy: 0.9530
```

In [16]:
```
from matplotlib import pyplot as plt
plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],label='validation loss')
plt.xlabel('# epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```

In [16]:
```
from matplotlib import pyplot as plt
plt.plot(history.history['loss'],'r',label='training loss')
plt.plot(history.history['val_loss'],label='validation loss')
plt.xlabel('# epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



In [18]:
```
plt.plot(history.history['accuracy'],'r',label='training accuracy')
plt.plot(history.history['val_accuracy'],label='validation accuracy')
plt.xlabel('# epochs')
plt.ylabel('loss')
plt.legend()
plt.show()
```



In [24]:
```
scores=model.evaluate(test_data,test_target,verbose=0)
print("accuracy: %.2f%%" % (scores[1]*100))
```

```
accuracy: 94.71%
```

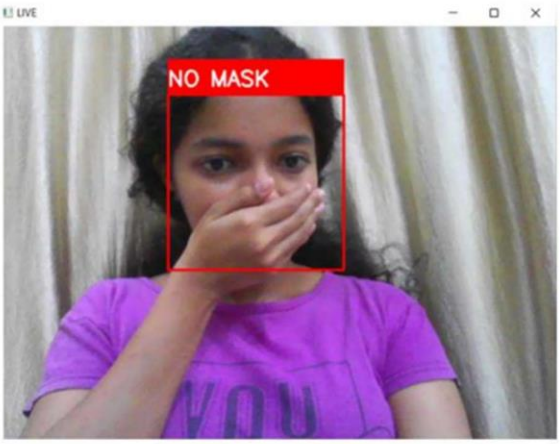
In [8]:
```
from keras.models import load_model
import cv2
import numpy as np
from pygame import mixer
mixer.init()

sound=mixer.Sound("mixkit-system-beep-buzzer-fail-2964.wav")
```

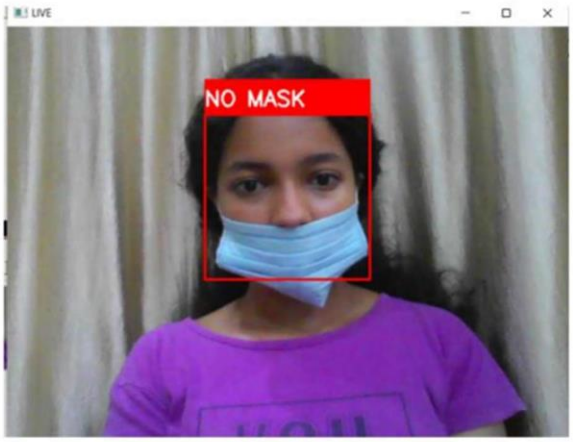
In [9]:
```
model = load_model('model-006.model')

face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

source=cv2.VideoCapture(0)

labels_dict={0:'MASK',1:'NO MASK'}
color_dict={0:(0,255,0),1:(0,0,255)}
```

In [3]:

```
while(True):

    ret,img=source.read()
    grayscale_img=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces=face_clsfr.detectMultiScale(grayscale_img,1.3,5)

    for x,y,w,h in faces:

        face_img=grayscale_img[y:y+w,x:x+w]
        resized_img=cv2.resize(face_img,(100,100))
        normalized_img=resized_img/255.0
        reshaped_img=np.reshape(normalized_img,(1,100,100,1))
        result=model.predict(reshaped_img)

        label=np.argmax(result,axis=1)[0]
        cv2.rectangle(img,(x,y),(x+w,y+h),color_dict[label],2)
        cv2.rectangle(img,(x,y-40),(x+w,y),color_dict[label],-1)
        cv2.putText(img,labels_dict[label],(x, y-10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)


        if(labels_dict[label] =='NO MASK'):
            sound.play()




    cv2.imshow('LIVE',img)
    key=cv2.waitKey(1)

    if(key==27):
        break

cv2.destroyAllWindows()
```

3.2 Results:





3.3 CONCLUSION AND FUTURE SCOPE:

The Post-COVID Face Mask Compliance Monitoring System represents a critical advancement in public safety measures, particularly in the wake of the COVID-19 pandemic. By harnessing deep learning technology, it provides an efficient and accurate means of enforcing mask mandates in crowded spaces. The system's ability to swiftly identify individuals wearing or not wearing masks reduces the burden on human resources and minimizes potential health risks. Through extensive training and advanced neural network architecture, it achieves high levels of accuracy in real-time assessments.

Future Scope:

The project lays a strong foundation for further enhancements and applications in the field of public health and safety. Future avenues for development include:

1. **Multi-Modal Sensing:** Integration of additional sensors like thermal cameras for temperature screening, enhancing the system's ability to identify potentially symptomatic individuals.
2. **Mask Quality Assessment:** Incorporating features to evaluate the quality and effectiveness of masks worn by individuals.
3. **Social Distancing Monitoring:** Expanding the system's capabilities to include monitoring and alerting for social distancing violations.
4. **Data Privacy and Security:** Strengthening measures to ensure the privacy and security of data collected by the system, adhering to relevant regulations and standards.
5. **Geographical Expansion:** Adapting the system to various geographic regions and diverse demographic profiles, considering cultural differences in mask-wearing norms.
6. **Real-Time Analytics and Reporting:** Incorporating features for generating reports and analytics on mask compliance trends, aiding in policymaking and enforcement strategies.

Chapter 4

REFERENCES

[1] Toshan Meenpal, Ashutosh Balakrishnan, Amit Verma," Facial Mask Detection using Semantic Segmentation", 2019 4th International Conference on Computing, pp. 1-6, DOI:10.1109/CCCS.2019.8888092

[2] Jiang, X.; Gao, T.; Zhu, Z.; Zhao, Y. "Real-Time Face Mask Detection Method Based on YOLOv3", Electronics 2021, pp. 1-17, https://doi.org/10.3390/electronics10070837

[3] Mingjie Jiang, Xinqi Fan, Hong Yan," Retina face mask: A face mask detector" June 9, 2020, pp. 1-9, arXiv:2005.03950v2 [cs.CV]

[4] Safa Teboulbi, Seifeddine Messaoud, Mohamed Ali Hajjaji," Real-Time Implementation of AIBased Face Mask Detection and Social Distancing Measuring System for COVID-19 Prevention",27 September 2017, Volume 2021, Article ID 8340779, pp. 1-21,10.1155

[5] G. Jignesh Chowdary, Narinder Singh Punn, Sanjay Kumar Sonbhadra, Sonali Agarwal," Face Mask Detection using Transfer Learning of InceptionV3",20 October 2020, pp. 1-11, arXiv :2009.08369v2 [cs.CV]