

Real Time Face Mask and social distancing detection using CNN

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

Alok Kumar (37110033)

Kundan Kumar Jha (37110394)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)

Accredited with —All grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **Alok Kumar (37110033)** and **Kundan Kumar Jha (37110394)** who carried out the project entitled **-Real Time Face Mask and social distancing detection using CNN** under my supervision from **December 2020 to March 2021**.

Internal Guide
DR.B. BHARTHI, M.E., Ph.D.

Head of the Department
Dr. S.Vigneshwari M.E., Ph.D.,

Head of the Department
Dr. L.Lakshmanan M.E., Ph.D.,

Submitted for Viva voce Examination held on _____

Internal Examiner

External Examiner

DECLARATION

I **Alok Kumar (37110033)** and **Kundan Kumar Jha (37110394)** hereby declare that the Project Report entitled **-Real Time Face mask and social distancing detection using CNN**” is done by me under the guidance of **DR.B. BHARTHI, M.E., Ph.D.**Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE:CHENNAI

SIGNATURE OF THECANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **DR.B. BHARTHI, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

COVID-19 brought enormous changes in the society and it enforces us to change the normal way of living our life. To avoid or better say to minimize the risk of COVID-19 effect we have to follow some basic rules i.e. to practice social distancing and wearing masks. It has been seen that those who follow these practices are found to minimize the effect of COVID-19. Many researchers in the mean while working on the vaccination for controlling it. Technologies like deep learning help out the developers to analyse the publicly available data such as the X-ray, CT scans and text data of various discussions carried out in the social media etc. Image processing and video processing are used to evaluate the monitoring of social distancing and face mask protection is followed or not. CCTV cameras connectivity in the public places, public transports, and hospitals are helpful to gather the required data for monitoring and analysis. Keras and Tensorflow tools used for image processing steps are discussed here.

This paper will focus on implementation of social distancing and real time face mask detection using CNN.

TABLE OF CONTENTS

	ABSTRACT	v
	LIST OF FIGURES	viii
CHAPTER No.	TITLE	PAGE No.
1.	INTRODUCTION	1
	1.1 INTRODUCTION	2
	1.2 SYMPTOMS OF COVID-19	3
	1.3 RESEARCH TOWARDS COVID-19	
2.	AIM AND SCOPE	6
	2.1 AIM OF PROJECT	6
	<i>2.1.1 OBJECTIVES</i>	6
	2.2 SCOPES	6
	<i>2.2.1 ADVANTAGES</i>	6
3.	SYSTEM DESIGN & METHODOLOGY	10
	3.1 EXISTING SYSTEM AND WORK	10
	3.2 SYSTEM CAPABILITIES	12
	3.3 SYSTEM FUNCTIONALITIES	16
	3.4 REQUIREMENT SPECIFICATION	18
	3.5 METHODOLOGY	18
	<i>3.5.1 SYSTEM ARCHITECTURE</i>	
	<i>3.5.2 MODULE IMPLEMENTATION</i>	
	3.6 EMERGING TECHNOLOGY	19

4.	RESULTS AND DISCUSSION	34
5.	CONCLUSION AND FUTURE WORK	43
	REFERENCES	44
	APPENDIX	
	A. PAPER ACCEPTANCE MAIL	46
	B. PLAGIARISMREPORT	47
	C. JOURNALPAPER	48
	D. SOURCECODE	59

LIST OF FIGURES

FIGURE No.	FIGURE NAME	PAGE No.
3.1	FLOW DIAGRAM	24
3.2	DATATSET WITH MASKS	28
3.3	DATASET WITHOUT MASKS	34
3.4	AUGMENTED DATA	35
3.5	FLOW DIAGRAM	36
3.6	FLOW DIAGRAM	37
4.1	MODEL ACCURACY	38
4.2	DETECTION OF FACE MASK	39
4.3	DETETCTION OF NO MASK	40
4.4	SAFE DISTANCING	41
4.5	UNSAFE DISTANCING	42
4.6	TRAIN CONFUSION MATRIX	
4.7	TEST CONFUSION MATRIX	

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

COVID-19 pandemic emerges at the end of December-2019, It created anxiety among citizens. WHO declared it as a pandemic disease because of its harmful effect for people. It took millions of lives worldwide. Almost all countries suffered from this disease and it caused a massive effect on economic stability. Many countries imposed lockdown for three to four months but instead of that also the number of cases kept increasing exponentially. But Lockdown was not a solution for this, slowly and gradually scientists and even common people analysed that people wearing a mask and maintaining social distance rarely suffered from this disease. So people started to follow this parameter.

In crowded places like school, college or university, offices where thousands of people gather at a common place, this parameter started to work as a weapon against this harmful disease and people also started following this parameter even in the supermarket, malls etc.

Gradually it emerges as an outbreak solution to this problem and the government also made it mandatory to follow these guidelines. People who were caught not following the guidelines were forced to pay the fine and severe action was taken against them.

When the people started to follow these practice, lockdowns were removed from various parts of the country as it was signalling an economic crisis. To avoid this crisis offices were resumed, public transport was again seen on the road. Government tried to bring common people's life once again to normal.

The novel COVID or COVID-19 began spreading during 2019 December at first from China, the city guest Wuhan. In china, the infection is started from the creatures and spread generally as a pandemic circumstance everywhere on the world. Coronavirus infection communicates to others however coordinate actual contact with the influenced patients and through air. The infection straightforwardly hit the lung cells through respiratory arrangement of the patients and permits it to

recreate the infection and makes an extreme irresistible problem in an exceptionally limited ability to focus time.

1.2 Symptoms of COVID -19

The most widely recognized manifestations of COVID are dry hack, fever, windedness, discomfort and migraine. The quick send of COVID produce serious muscle torments and empower the individuals with debilitate invulnerable framework got tainted by it without any problem. The outrageous phase of COVID-19 prompts demise of numerous people groups with extreme failing of lung and different organs of the body. Different examples of medicines are dealing with by the doctors everywhere on the world to find an effective method of restricting the infection being communicated to most exceedingly terrible stage.

1.3 Research towards COVID-19

There are three significant networks helped with advanced innovations can be found in more certain commitment on battling against COVID-19. The computerized endeavours go through by gathering the X-Ray data and CT-Scan information to recognize the expectation of COVID-19 with the assistance of man-made consciousness (AI). AI and AI people group have quite a gigantic extent of commitment on forecast of COVID19 out of a limited ability to focus time by executing different complex calculations. Another academic network with numerical specialists and actual officials breaking down the mind boggling dispersion models of the Corona infection to discover the different methods of infection versatility and social distancing needs.

CHAPTER 2

AIM AND SCOPE

2.1 AIM OF PROJECT

The primary aim of this proposed system is to develop a model for predicting facemasks and social distancing so that things can be done in a good surveillance. Currently, there are no efficient face mask and social distancing monitoring system which can be used in densely populated areas, schools and colleges, residential societies, markets and common places to ensure safety.

2.1.1 OBJECTIVES

The primary objective of this project is to develop a deep learning model for face mask detection and social distancing monitoring using Python, Keras, OpenCV and YOLO.

To develop the face mask detector model for detecting whether person is wearing a mask or not and monitoring the social distancing.

The objective will be achieved by training the model using CNN and in the last step, the OpenCV library is used to run an infinite loop to use our web camera to detect for facemask.

2.2 SCOPE

This model will help in building the strategies to prevent the spread of coronavirus which will be helpful for the society. Tracing and monitoring plays a key role in determining the spread and transit of coronavirus.

2.2.1 ADVANTAGES

- Get to know whether a person has worn the mask or not.
- Get to know about the social distancing between humans.

CHAPTER 3

SYSTEM DESIGN & METHODOLOGY

3.1 EXISTINGSYSTEM AND WORK

Before starting up the work, various other works were looked. Some of them were:

Presented a cumulative article that contains the different categories of COVID datasets available. These datasets are publicly available open source data. The listed sequence of data holds the X-rays images, CT scan images, in few cases with health background is narrowed then MRI images are also used. Another set of dataset used for COVID analysis is textual data based on discussions, medical suggestions made on social media etc. Clinical test results and reports are also considered in many dataset windows that are utilized for analysis of diagnostic procedures.

Evaluated a virtual social distancing model that helps out the peoples being alerted in the public places. They graphically represented four types of spacing calling intimate space, personal space, social space and public spacing. Based on distance measurement rule, the spaces are measured. The procedure belongs to scene understanding and geometrical measurement, homograph estimation, metric references and density estimation etc. the secondary analysis consists of two dimensional people detection and multiple angle people Social Distance Monitoring and Face Mask Detection Using Deep Neural Network 2 detection.

Evaluated a human detection framework, for monitoring the social distancing and safety misuse in the pandemic situation. They have utilized the pre-trained model of recurrent CNN to identify the different models. The human detection process is done using blob segmentation. These blobs are tracked with respect to the other to measure the distance between the other persons. They faced the challenge of detecting the person's body blobs during the outdoor area because of the correlations of other objects nearby. They found this challenge need to be rectified in further research. Evaluated a deep learning approach on face detection and face classification. Clustering of different faces are done using pre-trained facial dataset. Fddb dataset is applied to train and test the proposed model. The

proposed model is altered to obtain novelty and high performing in prediction. The accuracy achieved on using convolution neural network is noted high and the future challenges are declared using real time face images and live capturing of videos.

3.2 SYSTEMCAPABILITIES

The proposed system i.e. Real-Time Face Mask and social distancing detection using CNN consist of various features which make this model efficient for public usage and to determine whether a person is wearing a mask or not and whether the public is following the social distancing norm or not.

The system is capable of predicting the faces of the person and determining the person has worn the face mask or the person is without the face mask. It can predict multiple faces at a time.

The system uses a haar cascade face feature to recognize the faces in the real world and in this way it can remove the noises and get the face of a person and then it checks for the mask. The system is capable of measuring the distance in 2D coordinate which is a stepping stone in determining the social distancing. On the detection of face mask, it shows a green rectangle on the person's face. On the detection of no mask, it shows a red rectangle on the person's face. The system measures the distance between two faces on the screen and displays it, if the distance measure is less than the threshold distance which is 25 cm in our case then the system flashes a message of "STAY AWAY".

The system model accurately determines its result and there is less chance of error.

3.3 SYSTEM FUNCTIONALITIES

The functionality of the system is to determine the person is following the Covid-19 guidelines or not. The guidelines are to wear a facemask and follow the social distancing rules. The system can perform such functionality to a large extent.

3.4 REQUIREMENTS SPECIFICATION

This proposed software runs effectively on a computing system that has the minimum requirements. Undertaking all the equipment necessities are not satisfied but rather exist in their systems administration between the customer's machines already. So the main need is to introduce appropriate equipment for the product. The requirements are split into two categories, namely:

Software Requirements

The basic software requirements to run the program are:

- i. Microsoft Windows XP, Windows 7, Windows 10
- ii. PYTHON
- iii. OpenCV2, KERAS, TENSORFLOW
- iv. JUPYTER NOTEBOOK
- v. Anaconda Navigator

Hardware Requirements

The basic hardware required to run the program are:

- i. Hard disk of 500 GB and above
- ii. System memory (RAM) of 4GB and above
- iii. I3 processor-based computer or higher.
- iv. Web camera

3.5 METHODOLOGY

For detecting face masks and social distancing, a robust model using CNN is being developed. Wearing a facemask and social distancing are mandatory steps to reduce the transit of coronavirus. After a year of outbreak, it is still not in full control, so strict follow of the rules is mandatory to control the covid transit.

The CNN model will help in detecting facemasks, whether a user has worn them or not, and in the detection of social distancing.

For identifying face masks, a deep neural classifier to classify people with masks and without a mask is created.

Before using the deep neural classifier, the Haar cascade classifier for face detection and object detection is used, and then the neural classifier for checking the mask is used.

The deep CNN method along with Haar Cascade face classifier and computer vision techniques are used in detecting social distancing which is one of the major criteria to be followed to control covid. Firstly, detecting human beings and by using the 6 feet distancing calibration technique, whether if the human beings are maintaining social distance can be found and since in this project the hardware's are limited, The face would be used as an object and 25 cm as the threshold distance for measuring social distance

CNN uses a convolutional kernel to convolve with the original images or feature maps to extract higher-level features.

3.5.1 SYSTEM ARCHITECTURE

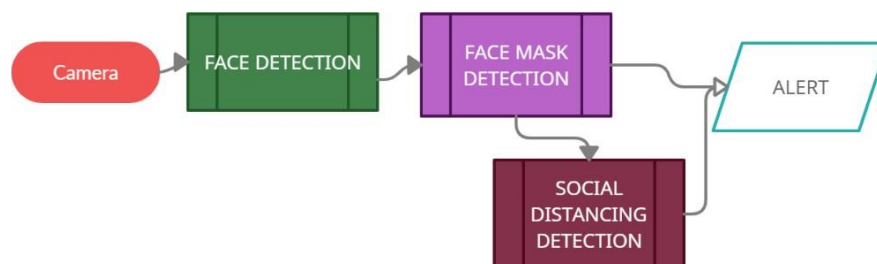


Fig:3.1: Flow diagram of facemask and social distancing detection

The system architecture is as follows: our model uses the camera with the help of an open-source computer vision library; it is a library of programming functions mainly aimed at real-time computer vision and detects faces and checks whether the person has worn the face mask or not, if there are multiple faces then it looks for the social distancing and generates an alert. The facemask detection and social distancing feature is discussed in the next sections of the paper.

3.5.2 MODULE IMPLEMENTATION

The implementation process is divided into four modules which simplify the understanding of our problem statement. These are as follows:

MODULE 1: DATA PREPROCESSING: It deals with the loading of dataset and preprocessing operations performed on the dataset.

MODULE 2: BUILDING/TRAINING THE CNN: It deals with the building of the model using CNN and training of the model with train and test data. It also determines the accuracy of the model with the train and test data.

MODULE 3: DETECTING FACE MASKS AND GENERATING ALERTS: This module deals with the real time prediction of facemasks, whether a person has a wear a face mask or not.

MODULE 4: DETECTING SOCIAL DISTANCING: This module determines the social distancing between the persons in the real time.

In depth building and discussion of the modules are done below:

MODULE 1: DATA PREPROCESSING

Before talking about the pre-processing operations, let's discuss the dataset. Dataset is an important factor in model performance. The dataset consists of about 1,376 images with 690 images containing people with face masks and 686 images containing people without face masks.

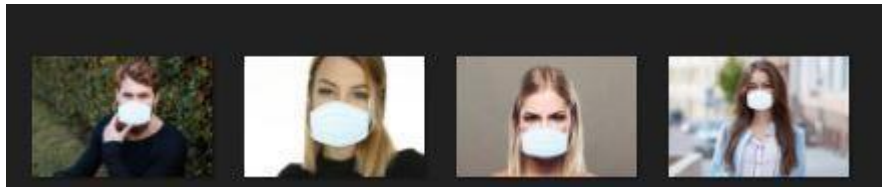


Fig:3.2: Dataset with masks



Fig:3.3: Dataset without masks

The above figure shows the dataset which is taken for training the model. The dataset was stored in two separate folders, one folder for images with a face mask and another one just images of a human face with no mask.

The pre-processing steps include data visualization which helps in visualizing the total number of images. Data Augmentation is another step of pre-processing, where the images are rotated and flipped, and are stored back in the dataset. The below figure shows augmentation of image which is done to include more images in the dataset. It is significantly done to increase the diversity of data available for training models.

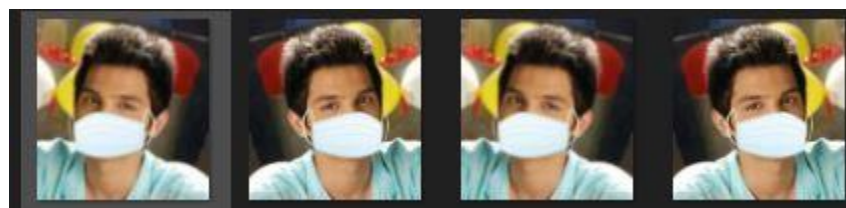


Fig:3.4: Augmented Data

In the next step, the data is splitted into **data** which will contain the images on which the CNN model will be trained and the **target** with the images on which our model will be tested.

The `split_size = 0.9`, which means that 90% of the total images will go to the training set, and the remaining 10% of the images will go to the test set.

MODULE 2: BUILDING/TRAINING THE CNN

Before the training, the CNN needs to be built which is sequential with various layers such as Conv2D, MaxPooling2D, Flatten, Dropout, and Dense.

The sequential API of the Keras library allows us to create the new layer for our model step by step. The first layer is the Conv2D with 200 filters and the filter size or the kernel size of 3x3. In the current work, the activation function used in the first layer is the 'Relu'. This Relu function stands for Rectified Linear Unit which will output the input directly if it is positive, otherwise, it will output zero. In the second layer, the maxpooling2d is used with the pool size of 2x2.

The next layer is again a Conv2D layer with another 100 filters of the same filter size 3x3 and the activation function used is the 'Relu'. This Conv2D layer is followed by a MaxPooling2D layer with pool size 2x2.

In the next step, the Flatten() layer to flatten all the layers into a single 1D layer is used.

After the flatten layer, the Dropout(0.5) layer to prevent the model from overfitting is used.

Finally, towards the end, the Dense layer with 50 units and the activation function as 'Relu' is used.

In this current work, the last layer of the model will be another Dense layer, with only two units and the activation function used will be the 'Softmax' function. The Softmax function outputs a vector that will represent the probability distribution of each of the input units. The Softmax function will output a vector with two probability distribution values.

After building the model, the model is compiled and the loss function and optimizer function is defined. In this model, the 'Adam' optimizer is used and the 'categorical_crossentropy' as the loss function for the training purpose.

Finally, the CNN model is trained for 20 epochs with two classes, one denoting the class of images with the face masks and the other without face masks.

MODULE 3: DETECTING FACE MASKS AND GENERATING ALERTS

The data model trained is loaded. Before detecting face masks, the face detection has to be implemented. The Haar Feature-based Cascade Classifier for detecting the features of the face will be used. This Cascade Classifier is designed by OpenCV to detect the frontal face by training thousands of images.

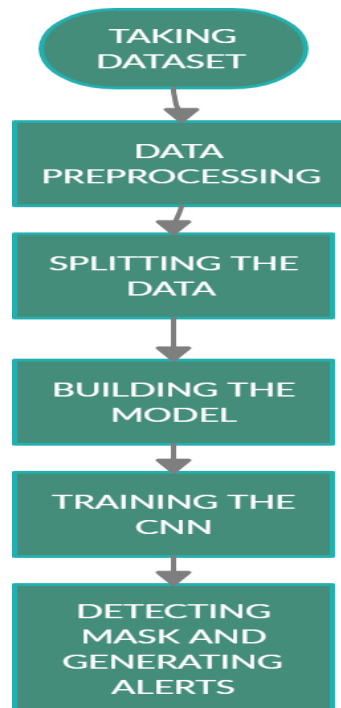


Fig:3.5: Flow diagram for face mask detection

The above figure illustrates the various processes which help in the building of models that helps in detecting face masks.

Two probabilities, '0' for Mask and '1' for without mask are labelled, and also setting the boundary rectangle color using the RGB values. 'Red' for 'without_mask' and 'Green' for 'with_mask'.

In the last step, the OpenCV library is used to run an infinite loop to use the webcam in which the face is detected using the Cascade Classifier.

The code "source=cv2.VideoCapture(0)" denotes the usage of webcam.

The model will predict the probability of each of the two classes(with_mask, without_mask). Based on which probability is higher, the label will be chosen and displayed around our faces.

MODULE 4. DETECTING SOCIAL DISTANCING

Social Distancing is one of the important factors to control covid transmission. Since the hardware is limited, the Haar Cascade Face Classifier is used to determine the distance between two persons in a frame. The distance between two faces can be scaled and measured. Depending on the preset minimum distance(which is 25 cm in our case), any distance less than the acceptable distance between any two individuals will be indicated with red lines that serve as a precautionary warning.



Fig:3.6: Flow diagram for Social Distancing

The previous figure explains the implementation of social distancing, the distance measurement is done in x1, y1, x2, y2 coordinates which are pushed in a stack and at the time of measurement these values are popped out of the stack data structure.

The feature was implemented with Python and OpenCV.

The region of interest of images focuses on the face of the individual.

Distance between every individual can be computed from a top-down view and the distances are scaled by the scaling factor estimated from camera view calibration.

The use of Euclidean distance formula is used to determine the distance between two persons.

3.6 EMERGINGTECHNOLOGY

In this world of technology, the world is reaching newer heights through innovation and technology day-by-day. Technology has changed the lives of people and made it easier. So we must take the advantage of the technology and apply it for the well-being of society. So we have decided to make use of the cutting-edge technologies to build this application which will help in controlling the spread of coronavirus and in determining the rules are being followed by the citizens or not and make them feel secured and comfortable. In our system Python, OpenCV, Tensorflow, Keras, and CNN have been used to make the model predict well and correctly. Technology can play an important role in reducing the number of cases happening in the world. With an average of 18k cases still in 2021, it's very important to control the crowd and here technology advancement can play an important role in controlling the spread.

CHAPTER 4

RESULTS AND DISCUSSION

The model was executed in an Anaconda environment by using Python Language. It uses open-source libraries like pandas, NumPy, and classification algorithms: Tensorflow, Keras, etc.

The experimental step was done in Intel i5-7200U CPU @2.50GGz with 8GB RAM.

```
In [37]: print(model.evaluate(test_data, test_target))  
5/5 [=====] - 2s 394ms/step - loss: 0.2168 - accuracy: 0.9493  
[0.2167614996433258, 0.9492753744125366]
```

Fig.4.1: Model Accuracy

The model was trained for 20 epochs. However, the model can be trained for more epochs to attain higher accuracy unless there occurs overfitting.

After the 20th epoch, the model has an accuracy of 99.5% with the training set and an accuracy of 93.9% with the test set which can be seen in the above figure. This implies that it is well trained without any over-fitting.

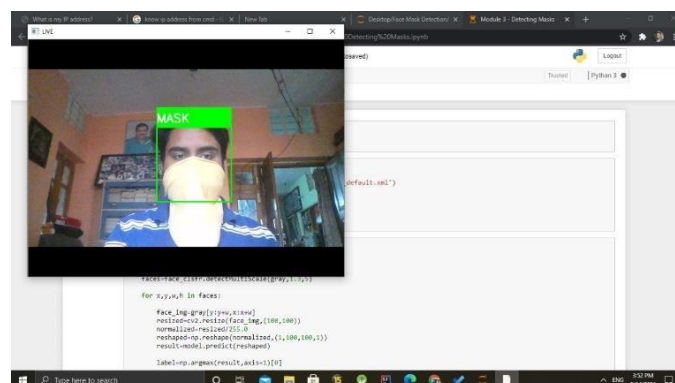


Fig.4.2: Detection of Face Mask

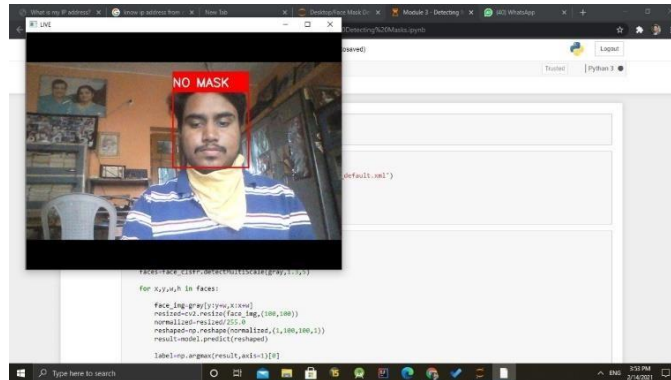
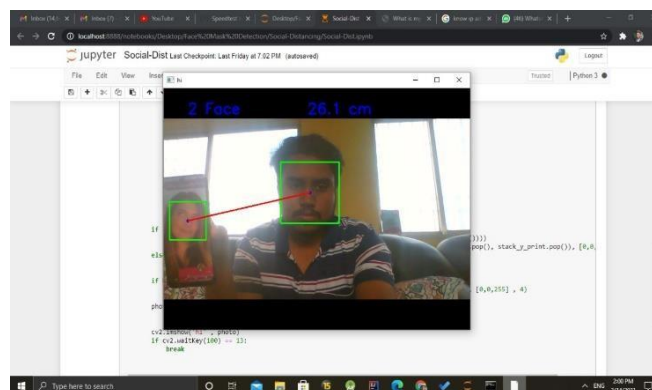


Fig:4.3: Detection of No Mask

The model was able to predict the faces with masks or not. If a mask is present, then it was highlighted by a green rectangle otherwise it was highlighted by red

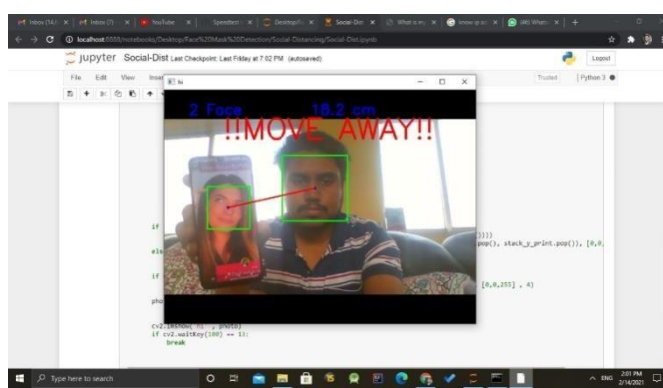


color.

Fig:4.4: Safe Distancing

Fig:4.5: Unsafe Distancing

The above screenshots show the distance between two faces which are being



detected in the video frame.

The Euclidean distance formula helps in determining the distance between two bodies. If the two faces are within the threshold distance, then a message "Move Away" is displayed on the screen which can be further changed to sending alerts to the admin with the screenshots of the persons. If the bodies are outside of the threshold distance, then no message is shown. The distance can be seen displaying on the screen with the number of faces being detected.

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known.

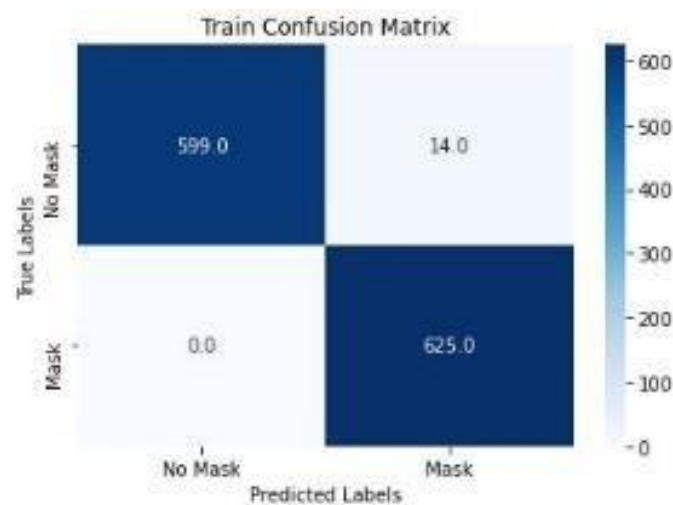


Fig:4.6: Train Confusion Matrix

After plotting the confusion matrix of training, it can be inferred that the model predicts 14 images wrong, else all the images are correctly predicted.

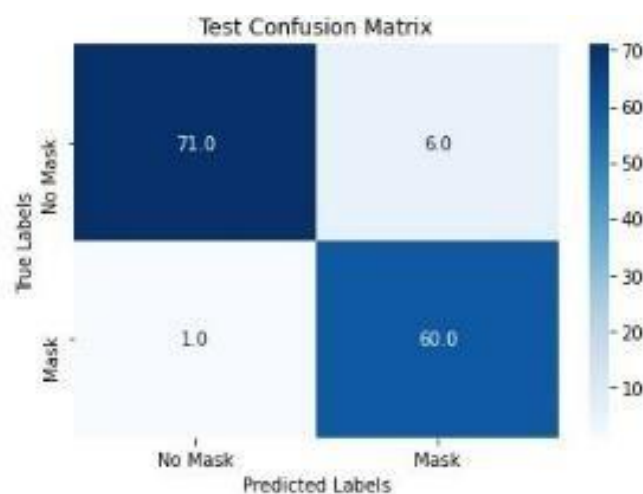


Fig:4.7: Test Confusion Matrix

After plotting the confusion matrix of testing, it can be inferred that the model predicts 7 images wrong, else all the images are correctly predicted. So the model will work with real time more precisely.

CHAPTER 5

CONCLUSION AND FUTURE WORK

CONCLUSION

Rapid transit of CORONA VIRUS enables the researchers to innovate various protection systems for the society. The detailed study is done with the help of previous literature works and results based on social distancing and technical computations done related to that. Protecting oneself from the critical virus is most important. Here image processing based social distancing framework with deep learning neural network for analysis is evaluated. The model was able to work on our objective of face mask detection and social distancing detection.

FUTURE WORK

In future, the proposed model needs to be implemented in large coverage area to improve the accuracy of the prediction model with respect to improving the training model. In future, Technologies like cloud can be integrated with this model to collect data and manipulate it accordingly. This combined model can be used in offices, universities, crowded places like railway stations, markets etc. In addition to it, backend systems need to be stronger for tracing the data.

REFERENCES

- [1] John Hopkins Coronavirus Resource Center,
<https://coronavirus.jhu.edu/map.htm>
- [2] J. K. Kelso, G. J. Milne and H. Kelly, —Simulation suggests that rapid activation of social distancing can arrest epidemic development due to a novel strain of influenza, *Il* vol. 9, no.1, Dec. 2009, pp.117-127.
- [3] Centers for Disease Control (CDC), —Implementation of Mitigation Strategies for Communities with Local COVID-19 Transmission, *Il*,
<https://www.cdc.gov/coronavirus/2019-ncov/downloads/community-mitigation-strategy.pdf>.
- [4] Paul Viola and Michael Jones, —Rapid object detection using a boosted cascade of simple features, *Il*, IEEE computer society conference on computer vision and pattern recognition, 2001.
- [5] M. Cristani, A. D. Bue, V. Murino, F. Setti and A. Vinciarelli, "The Visual Social Distancing Problem", *IEEE Access*, vol. 8, pp. 126876- 126886, 2020.
- [6] Musab Coşkun, Ayşegül Uçar, Özal Yıldırım, Yakup Demir, —Face recognition based on convolutional neural network. *International Conference on Modern Electrical and Energy Systems (MEES) Il*, 2017.
- [7] [https://github.com/prajnasb/observations/tree/master/experiments/ data/ Faces](https://github.com/prajnasb/observations/tree/master/experiments/data/Faces) dataset, Accessed: June 15, 2020.
- [8] Agnieszka Mikołajczyk, Michał Grochowski, —Data augmentation for improving deep learning in image classification problem, *Il*, 2018 International Interdisciplinary PhD Workshop (IIPhDW), 9-12, May 2018.
- [9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li FeiFei, —Imagenet: A large-scale hierarchical image database, *Il*, IEEE conference on computer vision and pattern recognition, 2009.
- [10] C. T. Nguyen et al., "A Comprehensive Survey of Enabling and Emerging Technologies for Social Distancing—Part I: Fundamentals and Enabling Technologies", *IEEE Access*, vol. 8, pp. 153479-153507, 2020.

[11] K. Simonyan and A. Zisserman, —Very deep convolutional networks for large-scale image recognition, 3rd International Conference on Learning Representations, 2015.

APPENDIX

PlagiarismReport

The screenshot displays the Adobe Acrobat Reader interface with a plagiarism report open. The report title is "Real Time Face mask and Social distancing using CNN DOC! - Google Docs2.pdf". The similarity index is 23%. The report is categorized into four sections: Internet Sources (18%), Publications (9%), and Student Papers (10%). The primary sources are listed as follows:

Rank	Source	Percentage
1	github.com Internet Source	9%
2	towardsdatascience.com Internet Source	6%
3	Yew Cheong Hou, Mohd Zafri Baharuddin, Salman Yussof, Sumayah Dzulkifly. "Social Distancing Detection with Deep Learning Model", 2020 8th International Conference on Information Technology and Multimedia (ICIMU), 2020 Publication	2%
4	Abdellah Oumina, Nouredine El Makhfi, Mustapha Hamdi. "Control The COVID-19	1%

The interface includes a sidebar with bookmarks, a top menu bar, and a right-hand panel with export and conversion options. The Windows taskbar is visible at the bottom.

REAL TIME FACE MASK AND SOCIAL DISTANCING DETECTION

Kundan Kumar Jha and Alok Kumar

Abstract- COVID-19 brought enormous changes in the society and it enforces us to change the normal way of living our life. To avoid or better say to minimize the risk of COVID-19 effect we have to follow some basic rules i.e to practice social distancing and wearing masks. It has been seen that those who follow these practices are found to minimize the effect of COVID-19.

This paper will focus on implementation of social distancing and real time face mask detection using CNN.

Keywords- COVID, social-distancing, OpenCV, CNN

I. INTRODUCTION

COVID-19 pandemic emerges during march-20[20], It created anxiety among citizens. WHO declared it as a pandemic disease because of its harmful effect for people. It took millions of lives worldwide. Almost all countries suffered from this disease and it caused a massive effect on economic stability. Many countries imposed lockdown for three to four months but instead of that also the number of cases kept increasing exponentially. But Lockdown was not a solution for this, slowly and gradually scientists and even common people analyzed that people wearing a mask and maintaining social distance rarely suffered from this disease. So people started to follow this parameter.

In crowded places like school, college or university, offices where thousands of people gather at a common place, this parameter started to work as a weapon against this harmful disease and people also started following this parameter even in the supermarket, malls etc.

Gradually it emerges as an outbreak solution to this problem and the government also made it mandatory to follow these guidelines. People who were caught not following the guidelines were forced to pay the fine and severe action was taken against them.

When the people started to follow this practice lockdowns were removed from various parts of the country as it was signalling an economic crisis. To avoid this crisis offices were resumed, public transport was again seen on the road. Government tried to bring common people's life once again to normal.

II. RELATED WORK

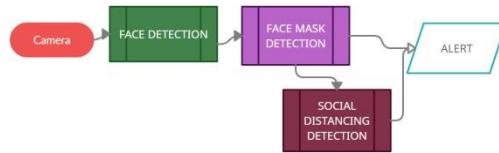
Various other technologies were developed during the pandemic to counter this disease, some of them were invented by the government sector and some of them were introduced by the private sector.

Technologies like IOT(Internet Of Things), Mobile applications were used to find a suitable solution which is feasible and easy to implement. Some of them are ArogyaSetu app, temperature sensor, real-time monitoring of number of cases etc. Out of these ArogyaSetu apps were enforced by the government and made it mandatory for the citizens to have it while travelling or in a crowded place.

Technologies like OpenCV, Deep learning, tensorflow were used by the developers to detect face masks but feasibility were questionable.

III. PROPOSED SYSTEM AND METHODOLOGY

In this paper, for detecting face masks and social distancing, we are developing a robust model using CNN. Wearing a facemask and social distancing are mandatory steps to reduce the transit of coronavirus. After the camera with the help of an open-source computer vision library; it is a library of programming a year of outbreak, it is still not functions mainly aimed at real-time computer vision and detects faces and checks whether the person has in full control, so strict follow worn the face mask or not, if there are multiple faces then it looks for the social distancing and generates of the rules are mandatory to an alert. The facemask detection and social distancing feature is discussed in the next sections of the control the covid transit. paper..



The system architecture is as follows: our model uses an open-source computer vision library; it is a library of programming functions mainly aimed at real-time computer vision and detects faces and checks whether the person has in full control, so strict follow worn the face mask or not, if there are multiple faces then it looks for the social distancing and generates of the rules are mandatory to an alert. The facemask detection and social distancing feature is discussed in the next sections of the paper..

The CNN model will help in detecting facemasks, whether a user has worn them or not, and in the detection of social distancing.

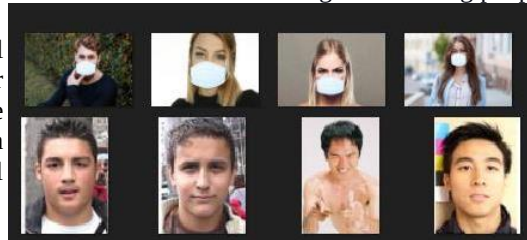
A. MODULE IMPLEMENTATION

MODULE 1: DATA PREPROCESSING

For identifying face masks, we create a deep neural classifier to classify people with masks and without a mask.

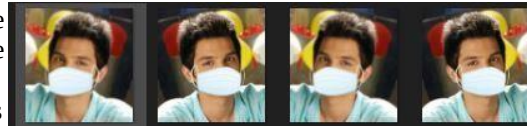
Before talking about the preprocessing operations, let's discuss the dataset. Dataset is an important factor in model performance. The dataset consists of about 1,376 images with 690 images containing people with face masks and 686 images containing people without face masks [14].

Before using the deep neural classifier, we use the Haar cascade classifier for face detection and object detection [11], and then we use the neural classifier for checking the mask.



The deep CNN method along with Haar Cascade face Data Augmentation is another step of preprocessing, where we rotate and flip each of the images in our classifier and computer vision dataset. We augment that to include more images in our dataset. It is significantly done to increase the techniques are used in detecting diversity of data available for training models [15].

social distancing which is one of the major criteria to be followed to control covid. Firstly, detecting human beings and by using the 6 feet distancing calibration technique we can find whether the human beings are maintaining social distance [12] and since we are limited to hardware, we are using our face as an object and 25 cm as the threshold distance for measuring social distance



In the next step, we split the data into **data** which will contain the images on which the CNN model will be trained and the **target** with the images on which our model will be tested.

The split_size = 0.9, which means that 90% of the total images will go to the training set, and the remaining 10% of the images will go to the test set.

MODULE 2: BUILDING/TRAINING THE CNN

Before the training, we need to build the CNN which is sequential [16] with various layers such as Conv2D, MaxPooling2D, Flatten, Dropout, and Dense.

CNN uses a convolutional The sequential API of the Keras library allows us to create the new layer for our model step by step. The kernel to convolve with the first layer is the Conv2D with 200 filters and the filter size or the kernel size of 3x3. The activation function used in the first layer is the 'Relu'. This Relu function stands for Rectified Linear Unit which will to extract higher-level features output the input directly if it is positive, otherwise, it will output zero. In the second layer, the maxpooling2d is used with the pool size of 2x2.

The next layer is again a Conv2D layer with another 100 filters of the same filter size 3x3 and the activation function used is the 'Relu'. This Conv2D layer is followed by a MaxPooling2D layer with pool size 2x2.

A. SYSTEM ARCHITECTURE

In the next step, we use the Flatten() layer to flatten all the layers into a single 1D layer.

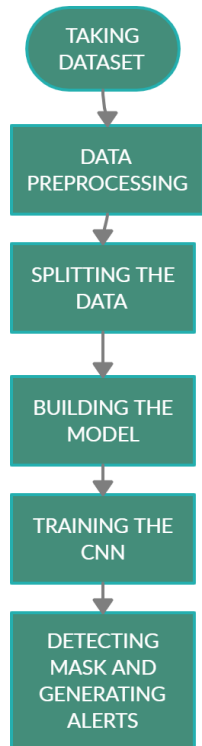
After the flatten layer, we use the Dropout(0.5) layer to prevent the model from overfitting.

Finally, towards the end, we use the Dense layer with 50 units and the activation function as 'Relu'.

The last layer of our model will be another Dense layer, with only two units and the activation function used will be the 'Softmax' function. The softmax function outputs a vector that will represent the probability distribution of each of the input units. The softmax function will output a vector with two probability distribution values.

After building the model, we compile the model and define the loss function and optimizer function. In this model, we use the 'Adam' optimizer and the 'categorical_crossentropy' as the loss function for the training purpose.

Finally, the CNN model is trained for 20 epochs with two classes, on denoting the class of images with the face masks and the other without face masks.



We label two probabilities, '0' for Mask and '1' for without mask, and also setting the boundary rectangle color using the RGB values. 'Red' for 'without_mask' and 'Green' for 'with_mask'.

In our last step, we use the OpenCV library to run an infinite loop to use our web camera in which we detect the face using the Cascade Classifier.

The code "source=cv2.VideoCapture(0)" denotes the usage of webcam.

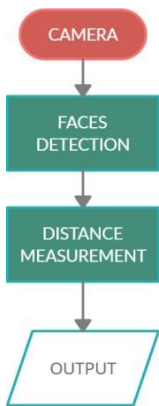
The model will predict the probability of each of the two classes(with_mask, without_mask). Based on which probability is higher, the label will be chosen and displayed around our faces.

MODULE 3: DETECTING FACE MASKS AND GENERATING ALERTS

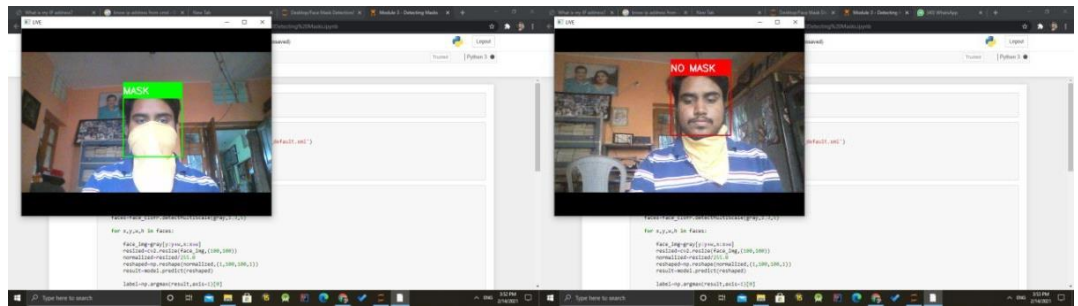
Social Distancing is one of the important factors to control covid transmission [17].

The data model trained is Since the hardware is limited, we are using the Haar Cascade face Classifier to determine the distance between two persons in a frame. The distance between two faces can be scaled and measured. Depending on the preset minimum distance(which is 25 cm in our case), any distance less than the acceptable face detection. We will be using distance between any two individuals will be indicated with red lines that serve as a precautionary warning.

Classifier for detecting the features of the face. This Cascade Classifier is designed by OpenCV to detect the frontal face by training thousands of images.



93.9% with the test set. This implies that it is well trained without any over-fitting.



The model was able to predict the faces with masks or not. If a mask is present then it was highlighted by a green rectangle otherwise it was highlighted by red color.

It was implemented with Python and OpenCV.

The region of interest of images focuses on the face of the individual.

Distance between every individual can be computed from a top-down view and the distances are scaled by the scaling factor estimated from camera view calibration.

The use of Euclidean distance formula is used to determine the distance between two persons.

IV. RESULT AND DISCUSSIONS

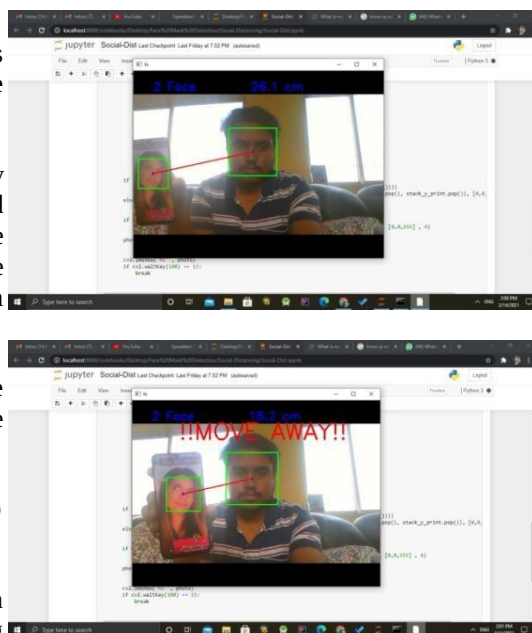
The model was executed in an Anaconda environment by using Python Language. It uses open-source libraries like pandas, NumPy, and classification algorithms: Tensorflow, Keras, etc.

The experimental step was done in Intel i5-7200U CPU @2.50GGz with 8GB RAM.

Epoch 20/20
31/31 [=====] - 48s 2s/step - loss: 0.0164 -

The model was trained for 20 epochs. However, we can train for more epochs to attain higher accuracy unless there occurs overfitting.

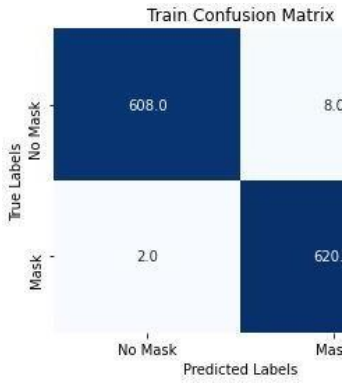
We can see that after the 20th epoch, our model has an accuracy of 99.5% with the training set and an accuracy of



The screenshots show the distance between two faces which are being detected in the video frame.

The Euclidean distance formula helps in determining the distance between two bodies. If the two faces are within the threshold distance then a message "Move Away" is displayed on the screen which can be further changed to sending alerts to the admin with the screenshots of the persons. If the bodies are outside of threshold distance, then no message is shown. the distance can be seen displaying on the screen with the no. of faces being detected.

A confusion matrix is a table that is often used to describe the performance of a classification model on a set of test data for which the true values are known.



stronger for tracing the data.

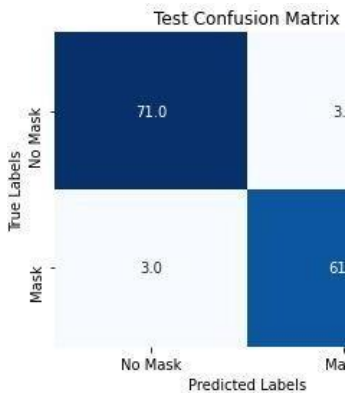
REFERENCES

- [11] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition, 2001
- [12] M. Cristani, A. D. Bue, V. Murino, F. Setti and A. Vinciarelli, "The Visual Social Distancing Problem," in IEEE Access, vol. 8, pp. 126876- 126886,2020,doi:10.1109/ACCESS.2020.3008370.
- [13] Musab Coşkun, AyşegülUçar, ÖzalYıldırım, Yakup Demir. Face recognition based on convolutional neural network. International Conference on Modern Electrical and Energy Systems (MEES), 2017.

After plotting the confusion matrix of training, it can be inferred that the model predicts 10 images wrong, else all the images are correctly predicted.

[14]<https://github.com/prajnasb/observations/tree/master/experiments/ data/ Faces dataset>, Accessed: June 15, 2020

[15] Agnieszka Mikołajczyk; MichałGrochowski: Data augmentation for improving deep learning in image classification problem. 2018 International Interdisciplinary PhD Workshop (IIPhDW), 9-12 May 2018.



[16] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li FeiFei. Imagenet: A large-scale hierarchical image database. IEEE conference on computer vision and pattern recognition, 2009

[17] C. T. Nguyen et al., "A Comprehensive Survey of Enabling and Emerging Technologies for Social Distancing—Part I: Fundamentals and Enabling Technologies," in IEEE Access, vol. 8, pp. 153479-153507, 2020, doi: 10.1109/ACCESS.2020.3018140.

[18] K. Simonyan and A. Zisserman : Very deep convolutional networks for large-scale image recognition, 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, 2015.

After plotting the confusion matrix of testing, it can be inferred that the model predicts 6 images wrong, else all the images are correctly predicted. So the model will work with real time more precisely [18].

V. CONCLUSIONS AND FUTURE WORKS

Considering its effects various other researches are possible. Considering its feasibility it is still a questionable action. Technologies like cloud can be integrated with this model to collect data and manipulate it accordingly. This combined model can be used in offices, universities, crowded places like railway stations, markets etc.

In addition to it, backend systems need to be more

B. SourceCode

```
{
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 19,
      "metadata": {},
      "outputs": [
        {
          "name": "stdout",
          "output_type": "stream",
          "text": [
            "{ 'with mask': 0, 'without mask': 1}\n",
            "[ 'with mask', 'without mask']\n",
            "[0, 1]\n"
          ]
        }
      ],
      "source": [
        "import cv2,os\n",
        "from matplotlib import pyplot as plt \n",
        "import numpy as np \n",
        "\n",
        "data_path='dataset'\n",
        "categories=os.listdir(data_path)\n",
        "labels=[i for i in range(len(categories))]\n",
        "\n",
        "label_dict=dict(zip(categories,labels))\n",
        "\n",
        "print(label_dict)\n",
        "print(categories)\n",
      ]
    }
  ]
}
```

```

    "print(labels)"
]
},
{
    "cell_type": "code",
    "execution_count": 20,
    "metadata": {},
    "outputs": [],
    "source": [
        "img_size=100\n",
        "data=[]\n",
        "target=[]\n",
        "\n",
        "\n",
        "for category in categories:\n",
        folder_path=os.path.join(data_path,category)\n", "
            img_names=os.listdir(folder_path)\n",
        "    \n",
        "    for img_name in img_names:\n",
        img_path=os.path.join(folder_path,img_name)\n", "
            img=cv2.imread(img_path)\n",
        "\n",
        "        try:\n",
        gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
        "\n",
        "            #Coverting the image into gray scale\n",
        "            resized=cv2.resize(gray,(img_size,img_size))\n",
        "            #resizing the gray scale into 100x100, since we need a fixed
common size for all the images in the dataset\n",
        "            data.append(resized)\n",
        "            target.append(label_dict[category])\n",
        "            #appending the image and the label(categorized) into the list
(dataset)\n",
        "\n",
        "    except Exception as e:\n",
        "        print('Exception:',e)\n",
        "\n"
    ]
}

```

```

]
},
{
  "cell_type": "code",
  "execution_count": 24,
  "metadata": {},
  "outputs": [],
  "source": [
    "import numpy as np\n",
    "\n",
    "data=np.array(data)/255.0\n",
    "data=np.reshape(data,(data.shape[0],img_size,img_size,1))\n",
    "target=np.array(target)\n",
    "\n",
    "from keras.utils import np_utils\n",
    "\n",
    "new_target=np_utils.to_categorical(target)"
  ]
},
{
  "cell_type": "code",
  "execution_count": 4,
  "metadata": {},
  "outputs": [],
  "source": [
    "np.save('data',data)\n",
    "np.save('target',new_target)"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
}

```

```

    }
  ],
  "metadata": {
    "kernelspec": {
      "display_name": "Python 3",
      "language": "python",
      "name": "python3"
    },
    "language_info": {
      "codemirror_mode": {
        "name": "ipython",
        "version": 3
      },
      "file_extension": ".py",
      "mimetype": "text/x-python",
      "name": "python",
      "nbconvert_exporter": "python",
      "pygments_lexer": "ipython3",
      "version": "3.8.5"
    }
  },
  "nbformat": 4,
  "nbformat_minor": 2
}
{
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 1,
      "metadata": {},
      "outputs": [],
      "source": [
        "import numpy as np\n",
        "\n",
        "data=np.load('data.npy')\n",

```

```

"target=np.load('target.npy')\n",
"\n"
]
},
{
"cell_type": "code",
"execution_count": 2,
"metadata": {},
"outputs": [],
"source": [
"from tensorflow.keras.models import Sequential\n",
"from tensorflow.keras.layers import
Dense,Activation,Flatten,Dropout\n",
"from tensorflow.keras.layers import Conv2D,MaxPooling2D\n",
"from tensorflow.keras.callbacks import ModelCheckpoint\n",
"\n",
"model=Sequential()\n",
"\n",
"model.add(Conv2D(200,(3,3),input_shape=data.shape[1:]))\n",
"model.add(Activation('relu'))\n",
"model.add(MaxPooling2D(pool_size=(2,2)))\n",
"#The first CNN layer followed by Relu and MaxPooling layers\n",
"\n",
"model.add(Conv2D(100,(3,3)))\n",
"model.add(Activation('relu'))\n",
"model.add(MaxPooling2D(pool_size=(2,2)))\n",
"#The second convolution layer followed by Relu and MaxPooling
layers\n",
"\n",
"model.add(Flatten())\n",
"model.add(Dropout(0.5))\n",
"#Flatten layer to stack the output convolutions from second
convolution layer\n",
"model.add(Dense(50,activation='relu'))\n",
"#Dense layer of 64 neurons\n",
"model.add(Dense(2,activation='softmax'))\n",

```

```

"#The Final layer with two outputs for two categories\n",
"\n",

"model.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])"
]
},
{
"cell_type": "code",
"execution_count": 3,
"metadata": {},
"outputs": [],
"source": [
"from sklearn.model_selection import train_test_split\n",
"\n",

"train_data,test_data,train_target,test_target=train_test_split(data,target,test_size=0.1)"
]
},
{
"cell_type": "code",
"execution_count": 4,
"metadata": {
"scrolled": true
},
"outputs": [
{
"name": "stdout",
"output_type": "stream",
"text": [
"Epoch 1/20\n",
"31/31 [=====] - 49s 1s/step - loss: 0.7507 - accuracy: 0.5311 - val_loss: 0.6649 - val_accuracy: 0.5121\n",
"INFO:tensorflow:Assets written to: model-001.model\\assets\n",
"Epoch 2/20\n",

```

```

"31/31 [=====] - 44s 1s/step -
loss: 0.6135 - accuracy: 0.6368 - val_loss: 0.4722 - val_accuracy:
0.7702\n",

"INFO:tensorflow:Assets written to: model-002.model\assets\n",

"Epoch 3/20\n",

"31/31 [=====] - 42s 1s/step -
loss: 0.4166 - accuracy: 0.8023 - val_loss: 0.3449 - val_accuracy:
0.8589\n",

"INFO:tensorflow:Assets written to: model-003.model\assets\n",

"Epoch 4/20\n",

"31/31 [=====] - 43s 1s/step -
loss: 0.2627 - accuracy: 0.8856 - val_loss: 0.2441 - val_accuracy:
0.9274\n",

"INFO:tensorflow:Assets written to: model-004.model\assets\n",

"Epoch 5/20\n",

"31/31 [=====] - 43s 1s/step -
loss: 0.2114 - accuracy: 0.9073 - val_loss: 0.2284 - val_accuracy:
0.9153\n",

"INFO:tensorflow:Assets written to: model-005.model\assets\n",

"Epoch 6/20\n",

"31/31 [=====] - 44s 1s/step -
loss: 0.1475 - accuracy: 0.9421 - val_loss: 0.2456 - val_accuracy:
0.9153\n",

"Epoch 7/20\n",

"31/31 [=====] - 50s 2s/step -
loss: 0.1131 - accuracy: 0.9659 - val_loss: 0.1721 - val_accuracy:
0.9435\n",

"INFO:tensorflow:Assets written to: model-007.model\assets\n",

"Epoch 8/20\n",

"31/31 [=====] - 48s 2s/step -
loss: 0.0557 - accuracy: 0.9861 - val_loss: 0.2046 - val_accuracy:
0.9355\n",

"Epoch 9/20\n",

"31/31 [=====] - 45s 1s/step -
loss: 0.0427 - accuracy: 0.9891 - val_loss: 0.2472 - val_accuracy:
0.9234\n",

"Epoch 10/20\n",

"31/31 [=====] - 43s 1s/step -
loss: 0.0636 - accuracy: 0.9747 - val_loss: 0.1678 - val_accuracy:
0.9476\n",

"INFO:tensorflow:Assets written to: model-010.model\assets\n",

```

```

    "Epoch 11/20\n",
    "31/31 [=====] - 44s 1s/step -
loss: 0.0428 - accuracy: 0.9882 - val_loss: 0.2371 - val_accuracy:
0.9395\n",
    "Epoch 12/20\n",
    "31/31 [=====] - 43s 1s/step -
loss: 0.0308 - accuracy: 0.9881 - val_loss: 0.1896 - val_accuracy:
0.9597\n",
    "Epoch 13/20\n",
    "31/31 [=====] - 43s 1s/step -
loss: 0.0205 - accuracy: 0.9928 - val_loss: 0.2192 - val_accuracy:
0.9516\n",
    "Epoch 14/20\n",
    "31/31 [=====] - 46s 1s/step -
loss: 0.0340 - accuracy: 0.9880 - val_loss: 0.3172 - val_accuracy:
0.9315\n",
    "Epoch 15/20\n",
    "31/31 [=====] - 47s 2s/step -
loss: 0.0274 - accuracy: 0.9897 - val_loss: 0.2612 - val_accuracy:
0.9315\n",
    "Epoch 16/20\n",
    "31/31 [=====] - 46s 2s/step -
loss: 0.0283 - accuracy: 0.9929 - val_loss: 0.2240 - val_accuracy:
0.9395\n",
    "Epoch 17/20\n",
    "31/31 [=====] - 43s 1s/step -
loss: 0.0265 - accuracy: 0.9917 - val_loss: 0.2202 - val_accuracy:
0.9516\n",
    "Epoch 18/20\n",
    "31/31 [=====] - 42s 1s/step -
loss: 0.0616 - accuracy: 0.9732 - val_loss: 0.2534 - val_accuracy:
0.9234\n",
    "Epoch 19/20\n",
    "31/31 [=====] - 44s 1s/step -
loss: 0.0318 - accuracy: 0.9903 - val_loss: 0.2332 - val_accuracy:
0.9395\n",
    "Epoch 20/20\n",
    "31/31 [=====] - 43s 1s/step -
loss: 0.0236 - accuracy: 0.9899 - val_loss: 0.2434 - val_accuracy:
0.9435\n"
]
}

```

```

],
"source": [
    "checkpoint = ModelCheckpoint('model-
{epoch:03d}.model',monitor='val_loss',verbose=0,save_best_only=True
,mode='auto')\n",

    "history=model.fit(train_data,train_target,epochs=20,callbacks=[checkp
oint],validation_split=0.2)"

]
},
{
    "cell_type": "code",
    "execution_count": 5,
    "metadata": {},
    "outputs": [
        {
            "data": {
                "image/png":
                "text/plain": [
                    "<Figure size 432x288 with 1 Axes>"
                ]
            },
            "metadata": {
                "needs_background": "light"
            },
            "output_type": "display_data"
        }
    ],
    "source": [
        "plt.plot(history.history['accuracy'],'r',label='training accuracy')\n",
        "plt.plot(history.history['val_accuracy'],label='validation accuracy')\n",
        "plt.xlabel('# epochs')\n",
        "plt.ylabel('loss')\n",
        "plt.legend()\n",
        "plt.show()"
    ]
},

```

```

{
  "cell_type": "code",
  "execution_count": 7,
  "metadata": {},
  "outputs": [
    {
      "name": "stdout",
      "output_type": "stream",
      "text": [
        "5/5 [=====] - 1s 284ms/step -
loss: 0.2824 - accuracy: 0.9348\n",
        "[0.2824377119541168, 0.9347826242446899]\n"
      ]
    }
  ],
  "source": [
    "print(model.evaluate(test_data,test_target))"
  ]
},
{
  "cell_type": "code",
  "execution_count": 8,
  "metadata": {},
  "outputs": [],
  "source": [
    "from sklearn.metrics import classification_report,confusion_matrix"
  ]
},
{
  "cell_type": "code",
  "execution_count": 9,
  "metadata": {},
  "outputs": [],
  "source": [
    "import seaborn as sn"
  ]
}

```

```

},
{
  "cell_type": "code",
  "execution_count": 10,
  "metadata": {},
  "outputs": [],
  "source": [
    "predictions_one_hot = model.predict(train_data)\n",
    "#cm1 =\n",
    "confusion_matrix(train_target,model.predict(train_data).round())\n",
    "cm1 = confusion_matrix(train_target.argmax(axis=1),\n",
    "predictions_one_hot.argmax(axis=1))\n",
    "predictions_one_hot1 = model.predict(test_data)\n",
    "#cm2 =\n",
    "confusion_matrix(test_target,model.predict(test_data).round())\n",
    "cm2 = confusion_matrix(test_target.argmax(axis=1),\n",
    "predictions_one_hot1.argmax(axis=1))\n",
    "#cm3 =\n",
    "confusion_matrix(test_target,model.predict(test_data).round())"
  ]
},
{
  "cell_type": "code",
  "execution_count": 11,
  "metadata": {},
  "outputs": [
    {
      "data": {
        "text/plain": [
          "Text(33.0, 0.5, 'True Labels')"
        ]
      },
    },
    "execution_count": 11,
    "metadata": {},
    "output_type": "execute_result"
  ],
},
{

```

```

"data": {
  "image/png":
  "text/plain": [
    "<Figure size 432x288 with 2 Axes>"
  ]
},
"metadata": {
  "needs_background": "light"
},
"output_type": "display_data"
},
"source": [
  "sn.heatmap(cm1,annot=True,fmt = '.1f',xticklabels=['No\nMask','Mask'],yticklabels=['No Mask','Mask'],cmap = 'Blues')\n",
  "plt.title('Train Confusion Matrix')\n",
  "plt.xlabel('Predicted Labels')\n",
  "plt.ylabel('True Labels')\n"
],
{
  "cell_type": "code",
  "execution_count": 12,
  "metadata": {},
  "outputs": [
    {
      "data": {
        "text/plain": [
          "Text(33.0, 0.5, 'True Labels')\n"
        ]
      },
      "execution_count": 12,
      "metadata": {},
      "output_type": "execute_result"
    },
    {

```

```

"data": {
  "image/png":
  "text/plain": [
    "<Figure size 432x288 with 2 Axes>"
  ]
},
"metadata": {
  "needs_background": "light"
},
"output_type": "display_data"
},
"source": [
  "sn.heatmap(cm2,annot=True,fmt = '.1f',xticklabels=['No",
  "Mask','Mask'],yticklabels=['No Mask','Mask'],cmap = 'Blues')\n",
  "plt.title('Test Confusion Matrix')\n",
  "plt.xlabel('Predicted Labels')\n",
  "plt.ylabel('True Labels')\n",
  "]\n",
  "},\n",
  "{\n",
  "  \"cell_type\": \"code\", \n",
  "  \"execution_count\": 13,\n",
  "  \"metadata\": {},\n",
  "  \"outputs\": [\n",
  "    {\n",
  "      \"name\": \"stdout\", \n",
  "      \"output_type\": \"stream\", \n",
  "      \"text\": [\n",
  "        \"          precision    recall  f1-score   support\\n\", \n",
  "        \"\\n\", \n",
  "        \"      0      0.98      0.99      0.99     625\\n\", \n",
  "        \"      1      0.99      0.98      0.99     613\\n\", \n",
  "        \"\\n\", \n",
  "        \"  micro avg      0.99      0.99      0.99    1238\\n\", \n",
  "        \"  macro avg      0.99      0.99      0.99    1238\\n\", \n",

```

```

        "weighted avg      0.99    0.99    0.99    1238\n",
        " samples avg      0.99    0.99    0.99    1238\n",
        "\n"
    ]
}
],
"source": [

"print(classification_report(train_target,model.predict(train_data).round(
)))"
]
},
{
    "cell_type": "code",
    "execution_count": 14,
    "metadata": {},
    "outputs": [
        {
            "name": "stdout",
            "output_type": "stream",
            "text": [
                "          precision    recall  f1-score   support\n",
                "\n",
                "         0      0.95      0.91      0.93         65\n",
                "         1      0.92      0.96      0.94         73\n",
                "\n",
                " micro avg      0.93      0.93      0.93        138\n",
                " macro avg      0.94      0.93      0.93        138\n",
                "weighted avg      0.94      0.93      0.93        138\n",
                " samples avg      0.93      0.93      0.93        138\n",
                "\n"
            ]
        }
    ],
    "source": [

"print(classification_report(test_target,model.predict(test_data).round())

```

```

)"
  ]
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
},
{
  "cell_type": "code",
  "execution_count": null,
  "metadata": {},
  "outputs": [],
  "source": []
}
],
"metadata": {
  "kernelspec": {
    "display_name": "Python 3",
    "language": "python",
    "name": "python3"
  },
  "language_info": {
    "codemirror_mode": {
      "name": "ipython",
      "version": 3
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython3",
    "version": "3.8.5"
  }
}

```

```

    }
},
"nbformat": 4,
"nbformat_minor": 2
}
{
  "cells": [
    {
      "cell_type": "code",
      "execution_count": 4,
      "metadata": {},
      "outputs": [],
      "source": [
        "from tensorflow.keras.models import load_model\n",
        "import cv2\n",
        "import numpy as np\n",
        "from scipy.spatial import distance as dist"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 5,
      "metadata": {},
      "outputs": [],
      "source": [
        "model = load_model('model-019.model')\n",
        "\n",
        "face_clsfr=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')\n",
        "\n",
        "source=cv2.VideoCapture(0)\n",
        "\n",
        "labels_dict={0:'MASK',1:'NO MASK'}\n",
        "color_dict={0:(0,255,0),1:(0,0,255)}"
      ]
    }
  ],

```

```

{
  "cell_type": "code",
  "execution_count": 6,
  "metadata": {},
  "outputs": [
    {
      "ename": "KeyboardInterrupt",
      "evalue": "",
      "output_type": "error",
      "traceback": [
        "\u001b[1;31m-----\u001b[0m",
      ],
      "source": [
        "while(True):\n",
        "\n",
        "    ret,img=source.read()\n",
        "    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)\n",
        "    faces=face_clsfr.detectMultiScale(gray,1.3,5) \n",
        "\n",
        "    for x,y,w,h in faces:\n",
        "        \n",
        "        face_img=gray[y:y+w,x:x+w]\n",
        "        resized=cv2.resize(face_img,(100,100))\n",
        "        normalized=resized/255.0\n",
        "        reshaped=np.reshape(normalized,(1,100,100,1))\n",
        "        result=model.predict(reshaped)\n",
        "\n",
        "        label=np.argmax(result,axis=1)[0]\n",
        "        \n",
        "        cv2.rectangle(img,(x,y),(x+w,y+h),color_dict[label],2)\n",
        "        cv2.rectangle(img,(x,y-40),(x+w,y),color_dict[label],-1)\n",
        "        cv2.putText(img, labels_dict[label], (x, y-10),cv2.FONT_HERSHEY_SIMPLEX,0.8,(255,255,255),2)\n",
        "    \n",

```

```

cv2.imshow('LIVE',img)\n", "
    key=cv2.waitKey(1)\n",
    "\n",
    "    if(key==27):\n",
    "        break\n",
    "    \n",
    "cv2.destroyAllWindows()\n",
    "source.release()\n",
    "\n",
    "    "
]
},
{
    "cell_type": "code",
    "execution_count": null,
    "metadata": {},
    "outputs": [],
    "source": []
}
],
"metadata": {
    "kernelspec": {
        "display_name": "Python 3",
        "language": "python",
        "name": "python3"
    },
    "language_info": {
        "codemirror_mode": {
            "name": "ipython",
            "version": 3
        },
        "file_extension": ".py",
        "mimetype": "text/x-python",
        "name": "python",
        "nbconvert_exporter": "python",

```

```
"pygments_lexer": "ipython3",  
"version": "3.8.5"  
}  
,  
"nbformat": 4,  
"nbformat_minor": 2  
}
```