*International Journal of New Media Studies (IJNMS), ISSN: 2394-4331*
*Volume 10 Issue 1, January-June, 2023, Impact Factor: 7.786*

# Automated Mask Detection in Live Video Streams

**Rubiya Rawat**

Student, School of Computing, Graphic Era Hill University, Dehradun-248002, India

**ABSTRACT**

**The ongoing global healthcare crisis caused by the pandemic has led to a critical situation worldwide. The virus primarily spreads through respiratory droplets emitted by infected individuals, posing a risk to others. Public places have a higher risk of transmission, and wearing a face mask is recommended by the World Health Organization (WHO) to reduce the chances of infection. In this project, we propose a method that utilizes TensorFlow and OpenCV to detect face masks on individuals. The approach involves drawing a bounding box around a person's face to determine whether they are wearing a mask or not. If a person's face is stored in the database, the system detects their identity and sends them an email notification reminding them to wear a mask and take necessary precautions. If the person's face is not present in the database, the system directly uses the live webcam to detect whether the person is wearing a mask or not. This system is particularly useful for businesses and organizations that aim to identify and protect individuals who are not wearing face masks. By detecting and notifying those who are not following the mask-wearing guidelines, appropriate measures can be taken to ensure everyone's safety.**

**Keywords: Computer Vision, Covid 19, Image processing, Mask, No mask, OpenCV, Pandemic, safety, Tensorflow.**

## INTRODUCTION

Face mask detection[1] is an effective approach toprevent the spread of the virus among individuals. To ensure compliance with this essential safety measure, a strategy needs to be developed. Although the spread of the COVID-19 virus has decreased, it is still present. The complete eradication of the virus can be achieved if everyone consistently follows all safety measures. This will significantly reduce the number of cases and eventually lead to the disappearance of the COVID-19 virus.The current situation of the COVID-19 pandemic necessitates the development of an effective face mask detection system[2]. This project aims to implement the system at various locations such as colleges, airports, hospitals, and offices[3] where there is a higher risk of COVID-19 transmission through close contact. Studies have shown that wearing face masks significantly reduces the risk of spreading the virus in the workplace. The project addresses the problem of object detection and classification, specifically distinguishing between two classes: Mask and Without Mask. A hybrid model combining deep learning and classical machine learning techniques will be employed to detect face masks. The system[4] will be built using a dataset and implemented using Python, OpenCV, TensorFlow, and Keras. Upon entering the premises, individuals will be required to scan their faces, ensuring that they have a mask with them. If anyone is found without a face mask, an audible alert will be generated. Given that workplaces are reopening and COVID-19 cases are still being reported, it is crucial for everyone to adhere to safety measures. This module aims to help detect non-compliance with mask-wearing regulations and promote a safer work environment. The face mask detection method consists of two main components: face detection and mask detection.

Currently, there is a lack of efficient face mask detection applications to determine whether a person is wearing a mask or not. Therefore, there is a growing demand for a reliable system to detect face masks in various settings, including transportation, densely populated areas, residential areas, and businesses, in order to ensure safety. This project utilizes machine learning models built with OpenCV and TensorFlow to accurately detect face masks on individuals[5].

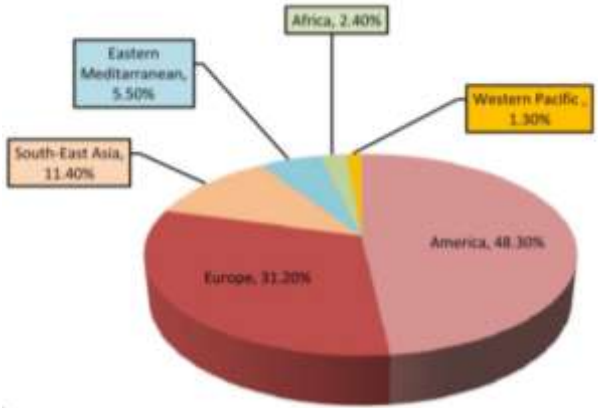


**Fig 1. This pie chart presents the cumulative number of deaths reported by the World Health Organization across various regions worldwide.**

**Problem Formulation**

We can implement a face mask detection system[6] to determine if a person is wearing a mask or not. This method detects and identifies the presence of a face mask on individuals. Our project involves building a real-time system that can detect whether a person on the webcam is wearing a mask or not[7]. We will train the face mask detector model using Keras and OpenCV. The model's architecture will be designed using Keras, and the training process is the initial stage of the project, followed by testing using OpenCV[8] and the webcam. Our dataset consists of 1376 images, with 690 images containing individuals wearing masks and 686 images of individuals without masks. We will evaluate the performance of the face mask detector model using OpenCV. This proposed model can be integrated with computer or laptop cameras, enabling it to detect individuals wearing or not wearing masks. The model is developed using a combination of deep learning and classical machine learning techniques, incorporating OpenCV, TensorFlow, and Keras.

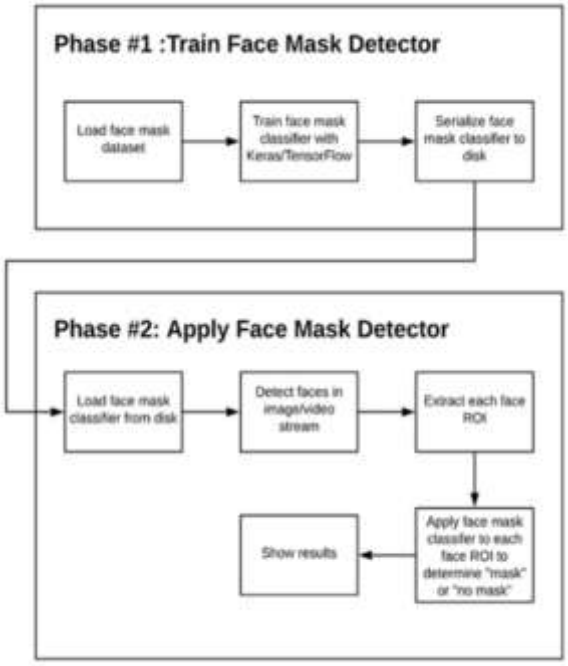


**Fig. 2: A COVID-19 face mask detector can be implemented in two phases.**

To develop a custom face mask detector, we can divide the project into two main phases, each consisting of several sub-steps. These phases and their corresponding sub-steps are illustrated in Figure 2 above.

The first phase is the training phase, where we focus on loading the face mask detection dataset from storage. We then proceed to train a model using Keras/TensorFlow[8] on this dataset. Once the training is complete, we serialize the trained face mask detector and save it to disk for future use.

The second phase is the deployment phase. After training the face mask detector, we move on to loading the detector model. We perform face detection on the input images or video frames using the loaded model. For each detected face, we classify it as either "with_mask" or "without_mask" based on the trained model's predictions.

By following these two distinct phases, we can successfully train a custom face mask detector and deploy it to detect and classify faces as either wearing or not wearing a mask.

**Implementation**

**Collection of dataset:** The dataset used in this study was obtained from the Kaggle Repository. After analyzing the dataset, it was divided into training and testing data.

**Model training for detection of face masks:** To train a model for face mask detection, facial images were extracted using a preexisting OpenCV module. Subsequently, a Keras model was trained to recognize the presence of face masks in these images.

**Detecting the person not wearnig mask**: The goal was to develop an OpenCV model that could identify individuals who were not wearing masks by cross-referencing their names with the database. The model was trained specifically for this purpose.

We can implement a face mask detection system to determine whether a person is wearing a mask or not. This method involves using face mask detection techniques to analyze images or video streams[9] and identify individuals who are wearing masks. In this project, our objective is to develop a real-time system that can detect whether a person on the webcam is wearing a mask[10]. We will train a face mask detector model using Keras and OpenCV, utilizing a network architecture. The project consists of two main parts: training the model, which involves using a dataset containing 1376 images (690 images of people wearing masks and 686 images of people without masks), and testing the model's performance using OpenCV and a webcam. The proposed model can be seamlessly integrated with computer or laptop cameras, enabling it to detect individuals who are wearing masks and those who are not. The model is built using a combination of deep learning and classical machine learning techniques, employing OpenCV, TensorFlow, and Keras.

Initially, a base model is created using Keras. This base model is then used as the foundation for

generating a head model. The generated version is trained using a labeled dataset, which is divided into two parts. One part contains 75% of the images and is used for training, while the remaining 25%[11] is used for testing the accuracy of the model. Once the model is trained, it can be utilized for detecting face masks on human faces. The input to the model is an image of a person without the background, which was used during the earlier training process. The model outputs whether a mask is present or not.

Additionally, another model is trained using images of individuals. These training images are associated with the person's name and email address, serving as labels for the model. OpenCV is employed to accomplish this task. When an input image is provided to the OpenCV model, it detects the person's face and prompts the user to input the person's name and email address, which are then stored in a database. The output of the primary model serves as the input to this secondary model. The face detected by the OpenCV model is compared with the faces in the database. If a match is found, a bounding box is drawn around the person's face, displaying their name, and an email and SMS are sent to notify them that they are not wearing a mask. Otherwise, if the person is wearing a mask, the bounding box will display the word "mask," and if not, it will display "No mask." OpenCV is utilized to perform these actions. When the CV model receives an input image, it identifies the person's face and prompts the user to provide their name and email address, which will be stored in the database. The output of the initial model serves as the input for this model. The detected face is then compared with the individuals present in the database. If there is a match, a bounding box is drawn around the person's face, displaying their name, and an email and SMS are sent to inform them that they are not wearing a mask. Conversely, if the person is wearing a mask[12], the word "mask" will be displayed under the bounding box. If the person is not wearing a mask, the phrase "No mask" will be displayed.

**THE PROPOSED METHOD**

The suggested approach includes a cascade classifier and a pre-trained CNN that incorporates two 2D convolution layers linked to dense neuron layers.

**DATA PROCESSING**

Data preprocessing encompasses transforming data from a given format into a more user-friendly, desired, and meaningful format. It can take various forms such as tables, images, videos, graphs, etc. This organized data aligns with an information model or structure and captures the relationships between different entities. The suggested approach focuses on handling image[13] and video data by utilizing Numpy and OpenCV.

A) Data visualization involves converting abstract data into meaningful representations through the use of visual elements to communicate knowledge and discover insights. It is a valuable tool for studying patterns within a dataset. The total number of images in the dataset is visualized for both categories, namely 'with mask' and 'without mask'. To achieve this, the statement "categories = os.listdir(datapath)" organizes the list of directories in the specified data path. The variable 'categories' now contains: ['with mask', 'without mask']. To determine the number of labels, we differentiate these categories by assigning labels as [0, 1]. Subsequently, each category is mapped to its corresponding label using the label dict=dict(zip(categories, labels)) operation, which pairs the items from each iterator in sequence. The resulting mapped variable 'label dict' appears as: {'with mask': 0, 'without mask': 1}.

B) Converting an RGB image to a grayscale image is a common practice in modern descriptor-based image recognition systems. The specific method used for this conversion is often not emphasized because it has little impact on the performance of robust descriptors. Including unnecessary color information can increase the size of the training data needed, so working with grayscale images is preferred. Grayscale simplifies the algorithm and reduces computational requirements, making it suitable for extracting descriptors instead of working directly with color images. Figure: Conversion of an RGB image to a grayscale image of size 100 x 100. The cv2.cvtColor (input image, flag) function is used to change the color space, where the flag specifies the type of conversion. In this case, the flag cv2.COLOR_BGR2GRAY is used for grayscale conversion. Deep CNNs require a fixed-size input image, so it is necessary to resize all the images in the dataset to a common size of 100 x 100 using cv2.resize().

C) Reshaping images is necessary because the input for image classification is a three-dimensional tensor, where each channel represents a specific color pixel. However, images in a dataset may have different sizes and their corresponding feature tensors may vary as well. This misalignment creates challenges during data collection and model implementation, as most convolutional neural networks (CNNs) require standardized input sizes. To address this issue, the input images are reshaped before being fed into the network. The first step is to normalize the pixel values, ensuring they fall within the range of 0 to 1. Then, the images are converted into four-dimensional arrays using the function np.reshape(data, (data.shape[0], imgsize, imgsize, 1)), where the '1' represents grayscale images. Since the final layer of the neural network has two outputs (with mask[14] and without mask) and

requires categorical labels, the data is further converted into categorical representation.

## TRAINING MODEL

**A) Model construction using CNN architecture:** Convolutional Neural Networks (CNNs) have gained prominence in various computer vision tasks. The current approach employs a Sequential CNN model. The model begins with a Convolution layer followed by Rectified Linear Unit (ReLU) and MaxPooling layers. The Convolution layer is trained using 200 filters, and the kernel size is set to 3 x 3, specifying the dimensions of the 2D convolution window. To ensure the model understands the input shape, the first layer of the model requires information about the input shape. Subsequent layers perform shape calculations automatically. In this case, the input shape is determined as data. shape[1:], extracting the dimensions from index 1 of the data array. The default padding is "valid," which trims the spatial dimensions while retaining a non-zero padded input volume.

The activation parameter for the Conv2D class is set as "relu," representing an approximately linear function that offers advantages for optimization [15] with gradient-descent methods compared to other activation functions. Max Pooling is employed to reduce the spatial dimensions of the output volume. The pool size is set to 3 x 3, and the resulting output shape is calculated as (input shape - pool size + 1)/strides, where the strides have a default value of (1,1). As depicted in Figure 4, the second Convolution layer incorporates 100 filters with a kernel size of 3 x 3. It is followed by ReLU and MaxPooling layers.

To input the data into the CNN, the long vector of input is passed through a Flatten layer, which transforms the matrix of features into a vector suitable for a fully connected neural network classifier. To address overfitting, a Dropout layer is added to the model, randomly setting 50% of the inputs to zero. A Dense layer with 64 neurons and a ReLU activation function is then included. The final layer, a Dense layer, has two outputs for the two categories and utilizes the Softmax activation function.

Prior to commencing the learning process, the model needs to be configured using the compile method. In this case, the "adam" optimizer is employed, and categorical cross-entropy (also known as multiclass log loss) is chosen as the loss function to be minimized during training. Given that this is a classification problem, the metrics are set to "accuracy."

**B) Splitting the data and training the CNN model:**
Once the blueprint for data analysis is established, the model needs to undergo training using a specific dataset and subsequently be evaluated using a separate dataset. A well-defined model and an optimized train-test split play a crucial role in achieving accurate predictions. In this scenario, the test size is set to 0.1, meaning that 90%[16] of the dataset is utilized for training, while the remaining 10% is reserved for testing purposes. ModelCheckpoint is employed to monitor the validation loss during training.
Next, the images from both the training set and the test set are fitted to the Sequential model. In this process, 20% of the training data is designated as validation data, aiding in the assessment of model performance. The model is trained for 20 epochs (iterations), striking a balance[17] between achieving high accuracy and mitigating the risk of overfitting.

## METHODOLOGY

**System design:**
To implement this project, we require the Python programming language, along with Deep Learning, Machine Learning, Computer Vision, and various Python libraries. The architecture of our system incorporates MobileNet as the backbone, which is suitable for both high and low computation scenarios. The proposed system utilizes an Algorithm.

**Implementation:**
Our implementation consists of four main modules:

**Dataset Collection:**
We gather a significant number of datasets comprising images of individuals wearing and not wearing face masks. The accuracy of our system depends on the quantity and quality of the collected images.

**Dataset Extraction:**
Using the MobileNet V2, we extract the features from the datasets containing images of people with and without masks. This extraction process helps in capturing the essential characteristics necessary for accurate detection.

**Model Training:**
We train our model using OpenCV and Keras, which are Python libraries known for their effectiveness in training models. This step involves fine-tuning the model to recognize and classify face mask[18] usage accurately.

**Face Mask Detection:**
In this module, we perform pre-processing on the input images and implement real-time face mask detection using live video feeds. If a person is detected wearing a mask, they are permitted to proceed. However, if a person is not wearing a mask, an alert is triggered, urging them to wear a mask to prevent the transmission of viruses.

**Benefits**

1) The task of manually monitoring whether people are wearing masks or not is challenging for officers. Our technique alleviates this burden by using a webcam to detect people's faces and prevent virus transmission.
2) This system offers fast and highly accurate results, ensuring efficient and reliable detection of face masks.
3) The versatility of our system enables its implementation in various settings such as ATMs, banks, and other public spaces.
4) By utilizing our technique, we can effectively safeguard individuals, ensuring their safety and well-being.

## CONCLUSION

The system proposed in this work addresses a significant challenge faced worldwide during the ongoing COVID-19[19] pandemic. The research successfully integrates a face mask detection model with a person identification model, allowing for email notifications to be sent to registered individuals on our platform who are not wearing masks. Additionally, this research work accomplishes the detection of multiple individuals in a single video frame, whether they are wearing masks or not. This innovative technology serves as an additional tool to ensure people's safety by detecting multiple individuals simultaneously and ensuring compliance with government-issued guidelines[20] during these challenging times.

## REFERENCES

[1] Wuttichai Vijitkunsawat, Peerasak Chantngram on" Machine Learning Algorithms for Face Mask Detection" doi: 10.1109/InCIT5058.2020.9310963
[2] Susanto Febri ,Alwan Putra , Riska Analia ,Ika Karlina Laila Nur Suciningtyas" Face Mask Detection For Preventing the Spread of COVID-19"doi: 10.1109/ESCI50559.2021.9396783
[3] Wei Bu∗ †, Jiangjian Xiao†, Chuanhong Zhou∗ , Minmin Yang‡, Chengbin Peng† DOI:10.1109/TSP52935.2021.9522677
[4] Sahana Srinivasan, Rujula Singh R ,Ruchita R Biradar, Revathi, "Face Mask Detection on Surveillance video datasets" DOI: 10.1109/ICCIS.2017.8274819.
[5] Ravi Kishore Kodali and Rekha Dhanekula" FACE MASKDETECTIONUSING DEEP LEARNING" doi: 10.1109/ESCI50559.2021.9396783.
[6] Baluprithviraj. K.N, Bharathi.K.R, Chendhuran.S" Artificial Intelligence based Smart Door with Face Mask Detection"
[7] Gayatri Deore, Ramakrishna Bodhula, Dr. Vishwas Udpikar, Prof. VidyaMore" Masked Face Detection Approach in Video Analytics" Doi:10.1109/CASP.2016.7746164
[8] Ian Goodfellow, Yoshua Bengio and Aaron Courville, "Deep Learning" MIT 38 Press, 2017.
[9] Andrew G. Howard, Menglong Zhu, Bo Chen and Dmitry Kalenichenko," MobileNets: Efficient Convolution Neural Networks for MobileVision Application", Computer Vision and Pattern Recognition, 2017.
[10] World Health Organization (WHO) "WHO Corona virus Disease (COVID-19) Dashboard" https://covid19.who.int/, [Online; accessed 13 Jan 2021].
[11] Kaihan Lin, Huimin Zhao, JujianLv (&) Jin Zhan, Xiaoyong Liu, Rongjun Chen, Canyao Li, and Zhihui Huang, "Face Detection and Segmentation with Generalized Intersection over Union Based on Mask R- CNN", Advances in Brain Inspired Cognitive System J. Ren et al. (Eds.): BICS 2019, LNAI 11691, pp. 106–116, 2020.
[12] N. Ud Din, K. Javed, S. Bae and J. Yi,"A Novel GAN- Based Network for Unmasking of Masked Face", in IEEE Access, vol. 8, pp. 44276-44287, 2020, doi:10.1109/ACCESS.2020.2977386.
[13] Wijara IGPS, Widiartha I, Arjarwani SE, "Pornographic Image Recognition Based on Skin Probability and Eigen porn of Skin ROIs Image", Telecommunication Compute Electron Control, 2015.
[14] Sneha Sen, Harish Patidar, "Face Mask Detection System for COVID_19 Pandemic Precautions using Deep Learning Method", Journal of Emerging Technologies and Innoative Research (JETIR), vol. 7, issue 10, pp.16-21.
[15] Khaoula Karimi, "Secure Smart Door Lock System based on Arduino and Smartphone App", JARDCS, Vol.12 issue 0, pp.407-414, 2020
[16] J. Howard et al., "An evidence review of face masks against COVID19,''Proc. Nat. Acad. Sci. USA, vol. 118, no. 4, 2021. [Online].Available:https://www.pnas.org/content/118/4/e2014564118
[17] Manasee Mishra, Piyusha Majumdar; Social Distancing During COVID-19: Will it Change the Indian Society? (2020).
[18] Marco Cristan, Alessio Del Bue, Vittorio Murino, Fraccesco Setti And Alessandro Vinciarelli, The Visual Social Distancing Problem, 2020.
[19] C. Zhao and B. Chen, "Real-time pedestrian detection based on improved YOLO model," in 2019 11th International Conference on Intelligent Human- Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 2019, pp. 25-28, doi:10.1109/IHMSC.2019.10101, 40.
[20] Coronavirus Disease (COVID-19): Vaccines. Accessed: Jun. 2, 2021. [Online]. Available: https://www.who.int/news-room/q-a-detail/coronavirus-disease-(covid-19)-vaccines