Detección de máscara

A PROJECT REPORT

Submitted By

Patel Deep Rajubhai [19BEIT30010]

Prajapati Dhvanil Nileshkumar [19BEIT30016]

Kothari Jainam Gautamchand [19BEIT30027]

Bunha Kunj Chimanbhai [19BEIT30038]

In fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

Information Technology

LDRP Institute of Technology and Research, Gandhinagar

Kadi Sarva Vishwavidyalaya

February, 2022

LDRP INSTITUTE OF TECHNOLOGY AND RESEARCH GANDHINAGAR

IT Department

CERTIFICATE

This is to certify that the Project Work entitled **"Detección de máscara"** has been carried out by **Patel Deep Rajubhai (19BEIT30010)** under my guidance in fulfilment of the degree of Bachelor of Engineering in Information Technology Semester-6 of Kadi Sarva Vishwavidyalaya University during the academic year 2021-22.

Prof. Himani Trivedi	Dr. Mehul Barot
Internal Guide	**Head of the Department**
LDRP-ITR	**LDRP-ITR**

LDRP INSTITUTE OF TECHNOLOGY AND RESEARCH GANDHINAGAR

IT Department

CERTIFICATE

This is to certify that the Project Work entitled **"Detección de máscara"** has been carried out by **Prajapati Dhvanil Nileshkumar (19BEIT30016)** under my guidance in fulfilment of the degree of Bachelor of Engineering in Information Technology Semester-6 of Kadi Sarva Vishwavidyalaya University during the academic year 2021-22.

Prof. Himani Trivedi	Dr. Mehul Barot
Internal Guide	**Head of the Department**
LDRP-ITR	**LDRP-ITR**

LDRP INSTITUTE OF TECHNOLOGY AND RESEARCH GANDHINAGAR

IT Department

CERTIFICATE

This is to certify that the Project Work entitled **"Detección de máscara"** has been carried out by **Kothari Jainam Gautamchand (19BEIT30027)** under my guidance in fulfilment of the degree of Bachelor of Engineering in Information Technology Semester-6 of Kadi Sarva Vishwavidyalaya University during the academic year 2021-22.

Prof. Himani Trivedi	Dr. Mehul Barot
Internal Guide	**Head of the Department**
LDRP-ITR	**LDRP-ITR**

LDRP INSTITUTE OF TECHNOLOGY AND RESEARCH GANDHINAGAR

IT Department

CERTIFICATE

This is to certify that the Project Work entitled **"Detección de máscara"** has been carried out by **Bunha Kunj Chimanbhai (19BEIT30038)** under my guidance in fulfilment of the degree of Bachelor of Engineering in Information Technology Semester-6 of Kadi Sarva Vishwavidyalaya University during the academic year 2021-22.

Prof. Himani Trivedi	Dr. Mehul Barot
Internal Guide	**Head of the Department**
LDRP-ITR	**LDRP-ITR**

ACKNOWLEDGEMENT

With great gratification, we would like to present this report on out topic “Detección de Mascara”. We are thankful to all that have helped us a lot for the successful completion of our project and providing us the courage for completing the work.

We are helpful to our Head of the Department Dr. Mehul P. Barot, our internal faculty Prof. Himani Trivedi for providing encouragement, constant support and guidance throughout our work giving us their valuable time.

At last, we would like to thank our parents and friends who have directly or indirectly helped us in making the project work successfully.

Patel Deep R. **[19BEIT30010]**

Prajapati Dhvanil N. **[19BEIT30016]**

Kothari Jainam G. **[19BEIT30027]**

Bunha Kunj C. **[19BEIT30038]**

Abstract

Since the inception of Machine Learning and Image Processing, the Face Detection has been a great revolution. But since the rise in Covid-19 in the world, it has seen a great deal of changes over the past few years. The major focus is to detect if the human is wearing the face mask properly or not. Several different face mask detection models have been developed using various approaches and algorithms.

“Detección de Mascara” is the Spanish translation of Detection of Mask. “Detección de Mascara” is the Deep Learning model which uses the python libraries TensorFlow, Keras, OpenCV and NumPy to detect Face Masks. This model uses the resources very efficiently, so it can be used for maintaining the medical hygiene. This model uses the Single Shot Detector (SSD) as the face detector and MobileNetV2 architecture as the framework for the classifier. MobileNetV2 is very lightweight and can be used in embedded devices like Raspberry pi, NVIDIA, etc. to perform real-time face mask detection.

The dataset used to train the model was downloaded from github of X-zhangyang. The Real-World-Masked-Face dataset has two classes of images which are images with mask and images without mask. The model trained on the same number of images for each class individually gives the F1 score of 0.96 and accuracy of 0.95.

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

1.1 Introduction

Over the last 2 years, humans have seen extraordinary happenings amongst which the COVID-19 being the most life changing event which has surprised the whole world since beginning of the year 2020. COVID-19 has called for the strict measures to be followed in order to prevent the spread of diseases. From the very fundamental hygiene measures and treatment in the hospitals, everyone is trying whatever they can in order to protect themselves as well as society. People use the various personal protective equipment like face mask, gloves, face shield, sanitizer, etc. Face mask is the prominent from all of them for the safety of humans. People are bound to wear the face mask as soon as they leave their home. Many government authorities and organizations strictly ensure that people are wearing face masks while they are in public places and crowded area.

To keep a check if people are following this basic safety principle and guidelines presented by government, a plan of action must be developed. A face mask detector system can be used to implement this strategy. Face Mask detector system means to identify if the person is wearing the mask or not in any event or public places. The above-mentioned goal can be achieved in two steps: first to detect the faces from the live videos and second to determine if the face has the mask on it or not. The major part of object detection over the last few years is been the face detection. Face detection can be used in various fields like security, authentication, biometrics and many more. There are various face detector systems are being implemented in the world, but there is always a possibility for optimization of those systems. The optimized system can help to manage the COVID-19 in a better and efficient manner.

In “Detección de Mascara” project, we will be developing a face mask detector that is able differentiate between the faces with mask and faces without mask. Then those people not wearing masks images can be stored and their details can be sent to the organizer of the event or the government authority using this system. In this model, we have used Single Shot Detector (SSD) FaceNet for the face detection and MobileNetv2 as the neural network classifier for differentiating images with mask and without mask. This system can be used over the live video streams or videos.

1.2 Aims and Objective of Work

The real time face mask detection can be implemented using the artificial intelligence and neural networks to detect the objects in the frame of video at a moment. The system needs to recognize the humans in the frame and determine whether they are wearing the masks or not. The system can be very helpful because even if people try to trick the surveillance by keeping any different kind of occlusion like scarf, cloth, etc. in front of camera, system can detect that the occlusion is not a mask. The system develops a different colour of text for the person wearing mask which is green and for person not wearing mask, it is red. Therefore, it can be easy to recognize people without mask even if the number of people in the video are high.

When the system detects a person not wearing mask properly, it can notify the event organizer by sending the details of person through email and the authorities can take appropriate actions against the one. This ensures that the organization are very strict and concerned about the health of the people. The analyses of the system can help the organizations to form a better guidelines and action plan for the safety of people.

1.3 Brief Literature Review

Object detection is one of the trending topics in the field of image processing and computer vision. Ranging from small scale personal applications to large scale industrial applications, object detection and recognition is employed in a wide range of industries. Some examples include image retrieval, security and intelligence, OCR, medical imaging and agricultural monitoring.

For the object detection, the image is read and the objects in the images are classified and the location of the objects in the image are also displayed with the help of box drawn with the help of object's dimensions. This box is known as bounding box. The development of the Viola Jones detector, which is an optimized technique of using Haar, was a big step forward for face detection. However, it failed because it cannot perform well on faces in areas where lighting is not proper and non-frontal faces. Since then, the researchers are eager to develop new algorithms and techniques using deep learning to improve models. Deep learning allows us to learn features with end-to-end manner and removing the useful knowledge required for developing feature extractors.

Two neural networks are required to detect objects using the two stage detectors. These neural networks are mainly Region-based Convolutional Neural Network(R-CNN) and faster R-CNN. The first neural network is used to generate region proposals and the second one refines these region proposals, performing a coarse-to-fine detection. These strategy gives high performance but the speed of the detector is low. R-CNN uses selective search to propose some candidate regions which may contain objects. After that, the proposals are fed into a CNN model to extract features, and a support vector machine (SVM) is used to recognize classes of objects. However, the second stage of R-CNN is computationally expensive since the network has to detect proposals on a one-by-one manner and uses a separate SVM for final classification. By introducing a region of interest (ROI) pooling layer to input all proposal regions at once, Fast R-CNN solves this problem. Faster R-CNN is the evolution of R-CNN and Fast R-CNN, which has training and testing speed is greater than its predecessors. While R-CNN and Fast R-CNN use selective search algorithms limiting the detection speed, Faster R-CNN learns the proposed object regions itself using a region proposal network (RPN).

On the other hand, one stage detectors use only a single neural network for region proposals and detection. The primary, one stage detectors are Single Shot Detector (SSD), You Only Look Once (YOLO) which are mostly used in all the applications of face detection in today's world. For the one stage detectors, the bounding boxes must be predefined. YOLO divides the image into various cells and then matches the bounding boxes to the objects for each cell. YOLO is one of the fastest object detection

algorithms but it is not good for small sized objects. SSD can detect objects of varying sizes in an image. Later, in order to improve detection accuracy, Lin et. al proposes Retina Network (RetinaNet) by combining an SSD and FPN (feature pyramid network) to increase detection accuracy and reduce class imbalance. One stage detector has higher speed but the trades off detection performance but then only preferred over two stage detectors.

1.4 Problem Definitions

We want to develop a system which can detect faces in real-world videos and identify if the detected faces are wearing the masks or not.

In the videos captured by the CCTV cameras, the faces of people are small, blurry, distorted and low resolution. The angle of the faces also varies from time to time. These real-world videos are very different from the videos captured by webcams, making the face mask detection problem much more difficult in practice.

1.5 Plan of our work

In this project, we will first explore mask/ no mask classification in webcam videos, and next, shift to the mask/ no mask classification problem in real-world videos as our final goal. Our reported model can detect faces and classify masked faces from unmasked ones in webcam videos as well as real-world videos where the faces are small and blurry and people are wearing masks in different shapes and colors.

2 Technology and Literature Review

Face mask detection adopts the same architectures as one-stage and two-stage detectors, but in order to improve the accuracy, more face like features are being added. The some already existing face mask detectors have been modelled using OpenCV, MobileNet, RetinaNet and Support Vector Machines (SVM). Our project used Real World Masked Face Dataset (RMFD) which contains 1970 masked faces and 1935 normal faces.

These images are of various dimensions and are unbalanced. They are pre-processed each in the 240x240 dimensions. The model requires the images in the 224x224 dimensions as input, so images are transformed into 224x224 by resizing it. Moreover, this project uses TensorFlow then they convert images to Tensors, which is the base data type that TensorFlow can work with. The ratio of the samples in test while splitting the dataset was kept 20% using the train test split function of sklearn library. Moreover, to deal with unbalanced data, they passed this information to the loss function to avoid unproportioned step sizes of the optimizer. They did this by assigning a weight to each class, according to its representability in the dataset. They assigned more weight to classes with a small number of samples so that the network will be penalized more if it makes mistakes predicting the label of these classes. While classes with large numbers of samples, they assigned to them a smaller weight. This makes their network training agnostic to the proportion of classes.

The flask library is used to develop the graphical user interface (GUI) for the project and the database used to store the details of the people attending an event. When the system finds a person is not wearing face mask, it searches the database to find his/her details and after finding out it sends an email to the organization.

3. System Requirement Study

3.1 User Characteristics

3.2 Hardware and Software Requirements

3.3 Assumptions and Dependencies

3.1 User Characteristics

User characteristics are the important aspects of any project. It helps us to define properly and focus on who the end users are for the system. The system will support two types of user privileges, attendee and organizers. The attendee will be able to register himself/herself for attending any event with all of its details. The organizers are given the rights to have all the information about the people attending their event. They can use model to detect face mask in their premises.

The organizer must have following characteristics:

1. Organizer must have basic knowledge of Deep Learning
2. Organizer must understand the use of model
3. Organizer must also be aware of the file system of the project

3.2 Hardware and Software Requirements

Hardware Requirements:

Processor	- i5 or higher
Processor Speed	- 1.1 GHz
RAM	- 4 GB RAM
Hard Disk	- 25 GB (minimum)

Software Requirements:

Operating System	- Windows, macOS, Linux
Software	- Jupyter Notebook
Backend Technology	- Python 3.8
Datasets	- Github
Libraries	- OpenCV, TensorFlow, Keras, sklearn, imutils, flask

3.3 Assumptions and Dependencies:

Assumptions:

They must know how to implement the libraries appropriately on various learning models.

The algorithms and classifiers which are used should provide more accurate classifications.

Dependencies:

This project is totally depended on some libraries and their particular versions:

TensorFlow	1.15.2
Keras	2.3.1
Imutils	0.5.3
NumPy	1.18.2
opencv-python	4.2.0.*
matplotlib	3.2.1

4. System Diagrams

MobileNetV2 is a neural network architecture released by Google. The architecture delivers high accuracy and is optimized for mobile devices. The MobileNetV2 is an improved version of MobileNetV1 and is available as a module of TensorFlow-Hub. MobileNets are small and low-power models developed to meet the resource constraints for a variety of use cases. MobileNetV2 improves the state-of-the-art performance of mobile models on multiple tasks and benchmarks as well as across a spectrum of different model sizes.

MobileNetV2 is a very effective feature extractor for object detection and segmentation. When used with Single Shot Detector Lite (SSDLite), MobileNetV2 is 35% faster with same accuracy than MobileNetV1. MobileNetV2 is based on the inverted residual structure, the non-linearities in narrow layers are removed. There are two types of blocks, one is residual block with stride of 1 and another is block with stride 2 for downsizing. There are two main features of MobileNetV2 one is Linear bottlenecks between layers and other is shortcut connections between bottlenecks.

There are 3 layers for both types of blocks. First layer is 1x1 convolution with ReLU6. Second layer is depth wise convolution. Third layer is 1x1 convolution but without any non-linearity. The bottlenecks of the MobileNetV2 encode the intermediate inputs and outputs while the inner layer encapsulates the model's ability to transform from lower-level concepts such as pixels to higher level descriptors such as image categories. There are Pooling layers which allows us to make calculations go faster by allowing the reduction in the size of input matrix without losing any features. There are mainly two types of Pooling one is Max Pooling and second is Average Pooling.

To reduce the overfitting of the model, it can use the dropout layers. Dropout Layer helps us in dropping random biased neurons from model. These neurons can be the part of any layer. The non-linear layer makes use of non-linear activation functions. Most commonly used non-linear functions are Rectified Linear Unit (ReLU), Sigmoid function, Leaky ReLU, etc. The Fully-Connected (Dense) layers help in classifying given images in binary classification. SoftMax is the activation function used in these layers.

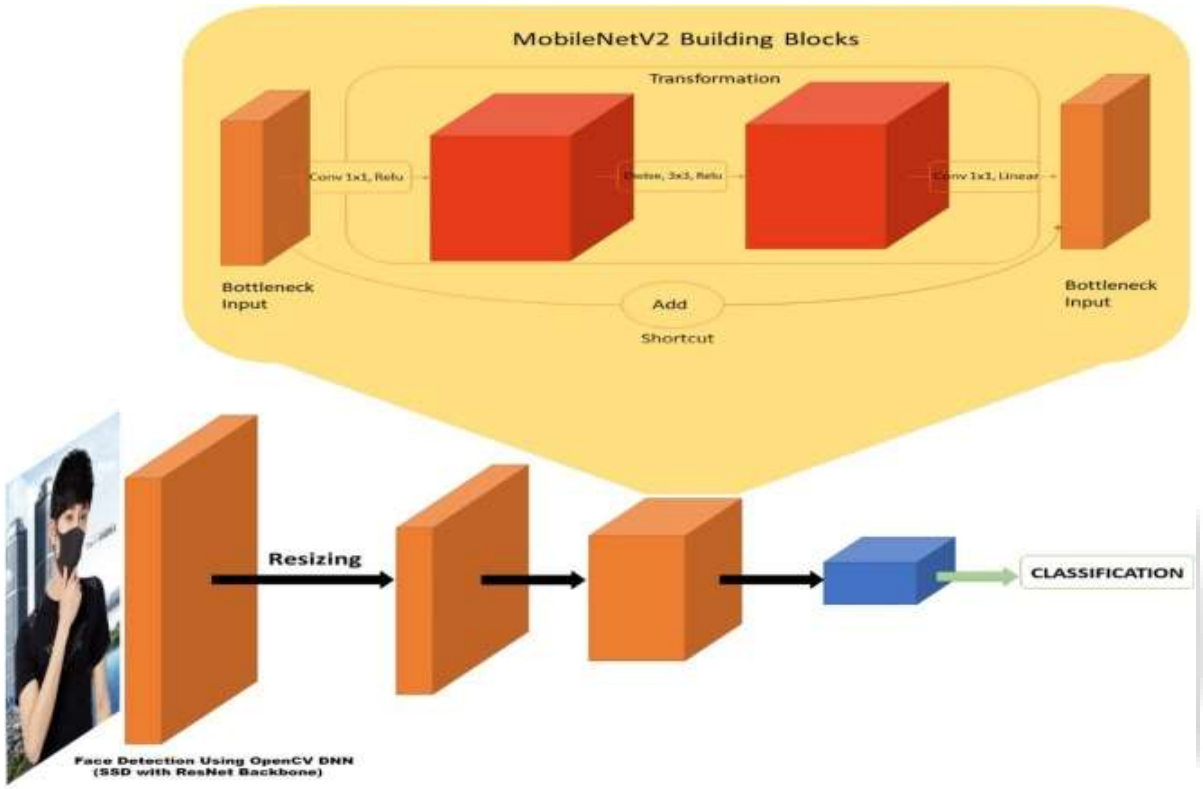

Fig 1 Architecture of MobileNetV2.

5. Flow of Implementation

Phase 1: Cleaning of Dataset

Step 1: After gathering the images of people with facemask and without facemask from GitHub, the cleaning of dataset includes:

- Only having all the images in same dimensions.
- Removing unclear and blur images.
- Manual removal of images, which are not in the correct category.

Phase 2: Training

Step 1: We load and preprocess our training data and then prepare it for data augmentation which includes:

- Fetching the list of images in our dataset, then initializing the list of images for the specific classes.
- Converting the training data into NumPy array format.
- We needed to give labels to our images with their class names (with mask, without mask), so we use One- Hot Encoding.
- The dataset is then partitioned into 80% training and 20% testing datasets using the train_test_split function of scikit-learn library.

Step-2: Preparing MobileNetV2, which includes:

- We define the learning rate, epochs and the batch size of our model.
- We load MobileNetV2 with pre-trained weights of ImageNet as a base model, the head (top) layer is not loaded as we want to integrate our own layer to it.
- We construct a new head (top) layer and add to the base model.
- The weights of all the base layers will not be updated, so we freeze it by making trainable attribute as false. The weights of the head layer will be changed.

Step-3: Implementing Data Augmentation which includes:

- We define some rules for data augmentation using the ImageDataGenerator() method of TensorFlow library.
- These rules help in developing more data for training the model.

- It uses properties like rotation_range,zoom_range,etc to perform data augmentation.

Step-4: Compiling and training our model:

- The model is compiled using Adam Optimizer, a learning rate and binary cross entropy as loss function (because there are 2 classes: with mask and without mask).
- Model is trained using the fit() method by passing all the required parameters.

Step-5: Saving the model.

Phase 3: Testing

- The trained model is then tested on the test dataset (20%) to evaluate the model.
- The classification report is then displayed for inspection.
- We get an accuracy of 0.95 and f1-score of 0.96 for each class.
- We then plot the graph of accuracy and loss for training.

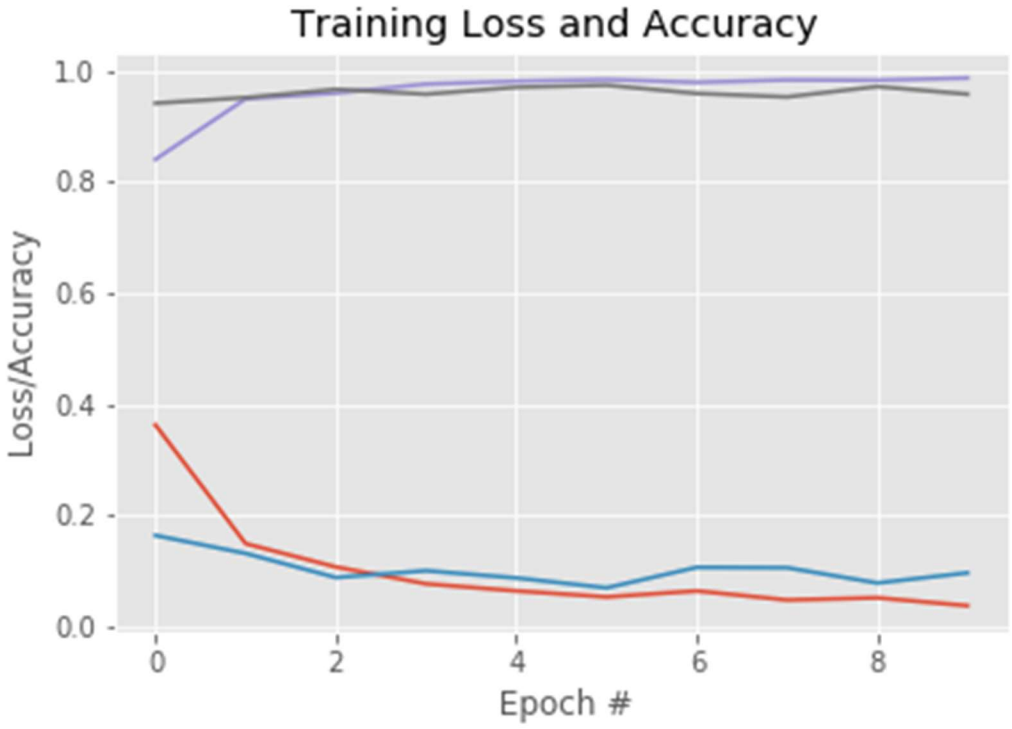

Fig 2 Training Loss and Accuracy Graph.

Phase 4: Face Mask Detection

- We import TensorFlow and keras libraries to load the saved model and for preprocessing images.
- To start our webcam, for displaying the live images and making bounding boxes on them we import OpenCV.
- With the help of OpenCV, the webcam is started and the images read in a frame.
- The faces are detected in the frame and bounding boxes are displayed around them using rectangle() method of OpenCV.
- The images are then preprocessed and passed to the model for the prediction.
- Based on the values of its prediction the result which is 'Mask' or 'No Mask' is decided.
- This result is shown on frame using putText() method of OpenCV.

Future Work:

- The faces, which are detected as not wearing masks are saved. These images can be searched through our database so that we can find details of person and notify the organizer using email.