Conference Proceeding Issue Published in International Journal of Trend in Research and Development (IJTRD), ISSN: 2394-9333, www.ijtrd.com

Smart-Brains: Face Mask Detector

[1]Somya Jaiswal, [2]Suryansh Yadav
[1,2]Department of Computer Science & Engineering, Global Nature Care Sangathan's Group of Institutions, Jabalpur, M.P., India

Abstract - COVID-19 is spreading all over the world, we need to stay safe by taking some simple precautions, such as physical distancing, wearing a mask, keeping rooms well ventilated, avoiding crowds, and cleaning our hands. According to World Health Organization (WHO) guidelines, when we wear a mask, we protect others as well as ourselves. Masks work best when everyone wears one. Masks should completely cover the nose and mouth and fit snugly against the sides of the face without gaps. Reports show that wearing a face mask reduces the risk of transmission[2]. Still, many people are not following these guidelines. So, we need an efficient solution for monitoring these people and creating a safe environment. Smart Brains is a face mask detection system which detects whether people are wearing a face mask or not and alerts everyone in the surrounding if someone is found without a face mask. In this project, we have used the MobileNetV2 Convolutional Neural Network (CNN) developed by Google to train a classifier model that can classify faces with masks and without masks in the live video stream. For training purposes, the dataset we have used consists of images with masks and without masks. After training, we have deployed this model on Raspberry Pi using its camera module to detect the faces without a mask on a live video stream. If any face is found without a proper mask, the system will alert people in the surrounding using a buzzer sound.

Keywords *- Deep Learning, Computer Vision, Tensorflow, Scikit-Learn, OpenCV, Keras, MobileNetV2, RaspberryPi*

I. INTRODUCTION

The COVID-19 virus can spread from an infected person's mouth or nose in small liquid particles when they cough, sneeze, speak, sing or breathe [1]. These particles range from larger respiratory droplets to smaller aerosols.

- Current evidence suggests that the virus spreads mainly between people who are in close contact with each other, typically within 1 meter (short-range). A person can be infected when aerosols or droplets containing the virus are inhaled or come directly into contact with the eyes, nose, or mouth.
- The virus can also spread in poorly ventilated and/or crowded indoor settings, where people tend to spend longer periods. This is because aerosols remain suspended in the air or travel farther than 1 meter (long-range).
- People may also become infected by touching surfaces that have been contaminated by the virus when touching their eyes, nose, or mouth without cleaning their hands.

The above points justify the importance of wearing a face mask for our protection. Today, there are more than 50 million active cases of coronavirus and a recorded death count of more than 3 million people around the world. These figures show how dangerous this virus is.

But the spread of the coronavirus pandemic has given rise to a remarkable degree of worldwide cooperation. Machine Learning and Deep Learning can help us to fight against coronavirus in many ways. Machine Learning algorithms allow us to evaluate a large amount of data to forecast the trend. In the case of coronavirus also, artificial intelligence and machine learning are used to better understand the infection rates and predict the spread of coronavirus. People are forced by laws to wear face masks in public in many countries. These rules and laws were developed as an action to the exponential growth in cases and deaths in many areas. However, the process of monitoring large groups of people is becoming more difficult. The monitoring process involves the detection of a person who is not wearing a face mask or wearing it improperly.

Here comes the **Smart-Brains: Face Mask Detector** which is based on computer vision and deep learning concepts. We have trained a classifier model using the concepts of deep learning and machine learning. This model is trained to identify all types and colors of face masks. We have also tried different algorithms to train this model with different weights and choose the most suitable algorithm that has the highest accuracy and consumed the least time in the process of detection. This system as a whole consists of a RaspberryPi, its camera module, and an integrated buzzer. The camera module is used to take a live video stream as an input, the processing of the input stream is done with the RaspberryPi, and if anyone is found without a face mask the buzzer will produce an alert sound as an output. We can also integrate our trained model with surveillance cameras.

1.1 DEEP LEARNING

Deep Learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign or to distinguish a traffic signal from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It's achieving results that were not possible before.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance.

1.2 COMPUTER VISION

Computer vision is the field of computer science that focuses on replicating parts of the complexity of the human visual system and enabling computers to identify and process objects in images and videos in the same way that humans do. But since we're not sure about how the human brain and the eyes process images, we can say that on a certain level computer vision is all about pattern recognition.

Deep learning has its approach to solving computer vision problems i.e. neural networks, a general-purpose function that can solve any problem representable through examples. When you provide a neural network with many labeled examples of a specific kind of data, it'll be able to extract common patterns between those examples and transform them into a mathematical equation that will help classify future pieces of information [4].

For example, creating a facial recognition system with deep learning only requires us to develop or choose a preconstructed algorithm and train it with examples of the faces of the people it must detect. Given enough images, the neural network will be able to detect faces without further instructions on features or measurements.

1.3 TENSORFLOW

TensorFlow is a powerful open-source software library for numerical computation, particularly well suited and fine-tuned for large-scale Machine Learning. Its basic principle is simple: you first define in Python a graph of computations to perform, and then TensorFlow takes that graph and runs it efficiently using optimized C++ code.

- Most importantly, it is possible to break up the graph into several chunks and run them in parallel across multiple CPUs or GPUs.
- TensorFlow also supports distributed computing, so you can train colossal neural networks on humongous training sets in a reasonable amount of time by splitting the computations across hundreds of servers.
- TensorFlow can train a network with millions of parameters on a training set composed of billions of instances with millions of features each.
- TensorFlow is not limited to neural networks or even Machine Learning; you could run quantum physics simulations if you wanted.

This should come as no surprise since TensorFlow was developed by the Google Brain team and it powers many of Google's large-scale services, such as Google Cloud Speech, Google Photos, and Google Search.

1.4 SCIKIT-LEARN

Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering, and dimensionality reduction via a consistent interface in Python. Rather than focusing on loading, manipulating, and summarising data, the Scikit-Learn library is focused on modeling the data. Scikit-learn comes loaded with a lot of features. Here are a few of them :

- Preprocessing
- Feature extraction
- Feature selection
- Parameter Tuning
- Clustering
- Cross-Validation
- Dimensionality Reduction
- Ensemble methods

We have used the Sklearn library for one-hot encoding of labels, splitting the available data into train data and test data, and generating the classification report of the trained model.

1.5 OPENCV

OpenCV is an open-source cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture, and analysis including features like face detection and object detection. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. OpenCV was originally developed in C++. In addition to it, Python and Java bindings were provided. OpenCV runs on various Operating Systems such as Windows, Linux, OSx, FreeBSD, Net BSD, Open BSD, etc.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, there are many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses span the range from stitching street-view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan.

We have used OpenCV for loading the face detection model and preprocessing the captured video in real-time which involves converting the color space and resizing the captured frame.

1.6 KERAS

Keras is a minimalist Python library for deep learning that can run on top of Theano or TensorFlow. It was developed to make implementing deep learning models as fast and easy as possible for research and development. It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs were given the underlying frameworks. It is released under the permissive MIT license. Keras contains numerous implementations of commonly used neural-network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural networks.

We have used the Keras library for loading image, converting loaded image to an array, preprocessing image using preprocessing of MobileNetV2, data augmentation, creating the neural network using the MobileNetV2 neural network as a base layer, modifying the output layer of the neural network, and in the deployment module it is used for loading the saved model.

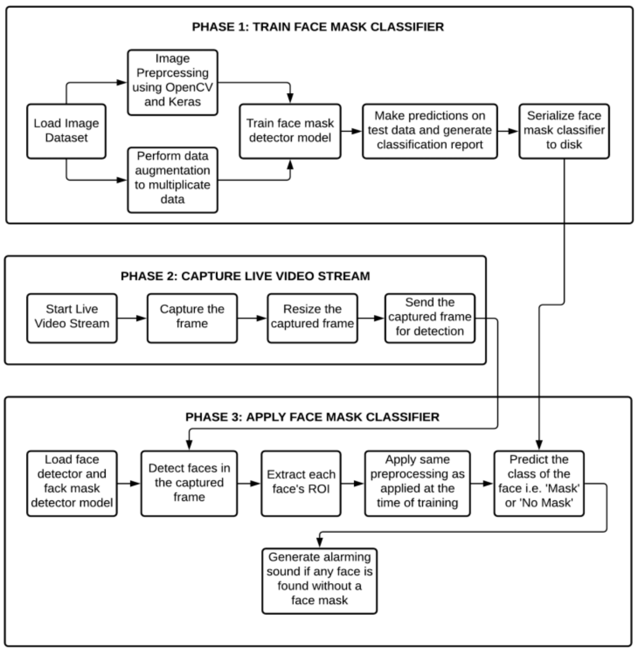
II. SYSTEM DESIGN

Smart-Brains focuses on capturing live video streams using RaspberryPi and identifying whether the person on the video stream wearing a face mask or not with the help of computer vision and deep learning algorithms by using the OpenCV, TensorFlow, Scikit-Learn, and Keras library.

2.1 OVERVIEW

i Train face mask classifier model using MobileNetV2
ii Capture real-time video using RaspberryPi
iii Apply the trained model over the captured video stream

2.2 FLOWCHART


PHASE 1: TRAIN FACE MASK CLASSIFIER
Load Image Dataset
Image Preprcessing using OpenCV and Keras
Perform data augmentation to multiplicate data
Train face mask detector model
Make predictions on test data and generate classification report
Serialize face mask classifier to disk
PHASE 2: CAPTURE LIVE VIDEO STREAM
Start Live Video Stream
Capture the frame
Resize the captured frame
Send the captured frame for detection
PHASE 3: APPLY FACE MASK CLASSIFIER
Load face detector and fack mask detector model
Detect faces in the captured frame
Extract each face's ROI
Apply same preprocessing as applied at the time of training
Predict the class of the face i.e. 'Mask' or 'No Mask'
Generate alarming sound if any face is found without a face mask

2.3 IMPLEMENTATION

Data Preprocessing

We did image preprocessing using OpenCV and Keras library, also used the standard preprocessing of MobileNetV2.

i Converting color space from BGR to RGB.
ii Resize the input frame (224 x 244).
iii Convert the image into a NumPy array.
iv Apply preprocess input of MobileNetV2.

Training MobileNetV2

MobileNetV2 builds upon the ideas from MobileNetV1, using depthwise separable convolution as efficient building blocks. However, MobileNetV2 introduces two new features to the architecture:

- linear bottlenecks between the layers, and
- shortcut connections between the bottlenecks.

MobileNets are small, low-latency, low-power models parameterized to meet the resource constraints of a variety of use cases. MobileNetV2 models are faster for the same accuracy across the entire latency spectrum [3].

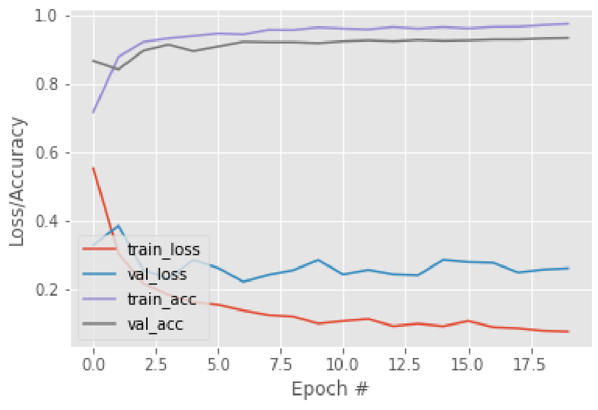


Fig. 1. Training Loss and Accuracy

MobileNetV2 has many layers. The weights of each layer in the model are predefined based on the ImageNet dataset. The weights indicate the padding, strides, kernel size, input channels, and output channels. We can use the applications library in Keras to create the MobileNetV2 model instead of defining/building convolutional layers ourselves.

In the output layer of the created model, we have added four customized fully connected layers. The layers are:

1. Average Pooling Layer with 7x7 weights and flattening this layer.
2. Dense Layer with 128 neurons and relu activation function.
3. Dropout Layer.
4. Dense Layer with 2 neurons and softmax activation function.

In the final layer, using the softmax function provides us the probabilities of each class i.e. ‘Mask’ and ‘No Mask’. We have taken 75% as the threshold probability i.e. if the probability of the ‘No Mask’ class exceeds 75% then the system will produce an alarming sound.

Face Mask Detection in Real Time

The approach of detecting face mask in real time consists of two steps:

1. Face Detection
2. Classifying the face using the model.

We have used OpenCV’s Caffe model of the DNN module for face detection [5] and served the detected ROI as an input for the CNN.

CONCLUSION

The technology is blooming with emerging trends, we have created a face mask detector which can possibly contribute to public healthcare. The architecture consists of Mobile Net as the backbone, it can be used for high and low computation scenarios. In order to extract more robust features, we utilize transfer learning to adopt weights from a similar task face detection, which is trained on a very large dataset.

This specific model could be used as a use case for edge analytics. Furthermore, the proposed method achieves state-of-the-art results on a public face mask dataset. By the development of face mask detection we can detect if the person is wearing a face mask and allow their entry would be of great help to the society.

Acknowledgment

We are greatly indebted to **Prof. Sumit Nema,** HOD, CSE Department, GNCSGI, Jabalpurwithout his encouragement and constant guidance we could not have finished this project. He was always a source of inspiration and motivator for innovative ideas during the entire span of this work.

We are equally grateful to **Dr. Rajiv Khatri,** Director, GNCSGI, Jabalpurfor providing all the necessary resources to carry out this Project work. We would like to thank all staff members of CSE, GNCSGI. And friends, for their support.

References

[1] P. A. Rota, M. S. Oberste, S. S. Monroe, W. A. Nix, R. Campagnoli, J. P. Icenogle, S. Penaranda, B. Bankamp, K. Maher, M.-h. “Characterization of a novel coronavirus associated with severe acute respiratory syndrome,” science, vol. 300, no. 5624, pp. 1394–1399, 2003.

[2] N. H. Leung, D. K. Chu, E. Y. Shiu, K.-H. Chan, J. J. McDevitt, B. J. Hau, H.-L. Yen, Y. Li, D. KM, J.

Ipetal., "Respiratory virus shedding in exhaled breath and the efficacy of face masks."

[3] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen "MobileNetV2: Inverted Residuals and Linear Bottlenecks'' 2018, pp. 4510-4520.

[4] Z. Wang, G. Wang, B. Huang, Z. Xiong, Q. Hong, H. Wu, P. Yi, K. Jiang, N. Wang, Y. Peiet al., "Masked face recognition dataset and application," arXiv preprint arXiv:2003.09093, 2020.[10]Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232, 2019.

[5] A. Kumar, A. Kaur, and M. Kumar, "Face detection techniques: a review, "Artificial Intelligence Review, vol. 52,no. 2, pp. 927–948, 2019.D.-H. Lee, K.-L Chen, K.-H. Liou, C.-L. Liu, and J.-L. Liu, "Deep learning and control algorithms of direct perception for autonomous driving,2019.