MASK DETECTION AND MASKED FACE RECOGNITION

[1]A.Punidha, [2]T.Arjun, [3]G.Naveenprabu

[4]R.Sandhiya Mathi, [5]K.Sashtika, [6]V.Sofiya

[1]Assistant Professor

[2,3,4,5,6]Students Pursuing Bachelor of Engineering

Department Of Computer Science and Engineering

Coimbatore Institute of Technology

ABSTRACT

Coronavirus disease 2019 has affected the world seriously in the recent times. The virus has globally infected over 2.7 million people and caused over 180,000 deaths and serious respiratory diseases. Therefore, concerning the health, one major protection method for people in this pandemic is to wear masks in public areas. Many public service providers need their customers to wear masks to use their service. Mask detection will provide a computerized solution to detect whether a person wearing a mask or not. Due to the current situation of COVID-19 the security systems based on passwords and fingerprints are unsafe. We have to rely on face recognition as it is much safer. Therefore, it is necessary to come up with face recognition approach that heavily rely on facial feature points that will not be covered under masks. The proposed system will deal with development of a model using OpenCV, Keras, Tenserflow, MobileNet for mask detection and face recognition using CNN in identifying the right person.

Keywords: Covid-19, Mask Detection, Face Recognition, OpenCV, Keras, Tenserflow, MobileNet, VGG – 16, CNN, MLP classifier.

I INTRODUCTION

Coronavirus disease 2019 (COVID-19) is a contagious disease caused by severe acute respiratory syndrome coronavirus2. The virus that causes COVID-19 spreads mainly when an infected person gets closely in contact with another person. Aerosols and minute droplets containing this virus can be spread from an infected person's nose and mouth as they breathe, cough, sneeze, sing, or speak. Other people also gets infected if the virus gets into their mouth, nose or eyes. Preventive measures include social distancing, quarantining, ventilation of indoor spaces, covering coughs and sneezes, hand washing, and keeping unwashed hands away from the face. Using face masks or coverings can be recommended in public to minimise the risk of transmissions.

Many public service providers need their customers to wear masks to use their service. Mask detection will provide a computerized solution to detect whether a person wearing a mask or not. Due to the current situation of COVID-19 the security systems based on passwords and fingerprints are unsafe. We have to rely on face recognition as it is much safer. As wearing masks becomes essential life saving

measure in this pandemic, face recognition techniques has nearly failed, that will affect the authentication applications that depends on face recognition. For example, entry and exit in public places, face attendance in organisations, face gates at railway stations etc.

Traditional systems cannot effectively recognize the faces with mask, but removing masks for passing authentication will increase the risk of virus infection.To solve this problem, it is necessary to come up with face recognition approach that heavily rely on facial feature points that will not be covered under masks.Our proposed system will deal with mask detection and masked face recognition.

II MASK DETECTION

TENSORFLOW : Tensorflow is an open-source library for machine learning and numerical computation in large-scale that ease Google Brain TensorFlow, the process of acquiring data, training models, serving predictions, and refining future results. Tensorflow bundles machine learning and deep learning models and algorithms together. Tenserflow uses python as a convenient front-end and runs it efficiently in optimized C++. Tensorflow allows developers to create and perform graph of computations. Each node in the graph represents a mathematical operation whereas each connection represents data. Hence, instead of dealing with low-details like figuring out proper ways to hitch the output of one function to the input of another, the developer focuses on the overall logic of the application. TensorFlow plays a role in text-based applications, image recognition, voice search, and many more.

KERAS : Keras is a deep learning Application Programming Interface written in Python which runs on top of the machine learning platform, Tenserflow. The main focus in developing this is for enabling fast experimentation. .Keras is the high-level API of TensorFlow 2 and an approachable, highly-productive interface to solve machine learning problems, in focus with modern deep learning. It provides necessary abstractions and building blocks for developing and shipping machine learning solutions with high iteration velocity.

OPENCV : OpenCV library is designed to solve many computer vision-related tasks like object detection, image manipulation, edge detection, shape detection, motion tracking, and many real-time computer vision tasks.

OpenCV is capable of image analysis and processing. It plays a vital role at taking frames out of video or taking in two frames from a stereoscopic camera and run algorithms to extract information. For example, using OpenCV can provide the mathematical tools required to capture images and track a particular object as it moves around. This is not provided directly but the mathematical tools needed to process the images to extract such information is available.

MOBILENET : MobileNet is a CNN architecture model for Mobile Vision and Image Classification . MobileNet uses very less computation power to run and apply transfer learning to. This makes it a perfect fit for Mobile devices,embedded systems and computers without GPU or low computational efficiency with compromising significantly with the accuracy of the results.

III IMPLEMENTATION

The Face Mask Detection is applied in two different stages. First stage includes training of Face Mask Detector and second stage deals with applying Face Mask Detector model on to the test images or real time video streams. For the first stage, initially start loading the face mask dataset , following it with the training of face mask classifier using some of Keras and Tensorflow libraries and lastly serializing face mask classifier to disk. In second stage, first load the face mask classifier from disk. Then, use it to detect faces in image or video stream and for each face, extract

the region of interest. Apply face mask classifier to each region of interest part of the face to determine whether person has wore mask or not and at last display the results. Also find the accuracy gained in determining the correct result.

The datasets used for training the model is of two – masked and unmasked datasets.

Firstly, import all the necessary libraries required for implementation. Some of which are:

(i) tensorflow, keras for Data augmentation, Loading the MobilNetV2 classifier, Building a new fully-connected (FC) head, Pre-processing, Loading image data, sklearn, imutils, matplotlib, numpy etc.

(ii) sklearn for binarizing class labels, segmenting our dataset, and printing a classification report

(iii) imutils paths implementation to find and list images in our dataset.

(iv) matplotlib to plot our training curves.

Then, construct the argument parser and parse command line arguments required while running the code. Also specify hyperparameter constants including initial learning rate, number of training epochs, and batch size.

Phase 1: Training

Step 1: Load and preprocess the training data and then prepare it for data augmentation which includes:

(i) Grabbing the list of images in the dataset directory, then initializing the list of data (i.e.,images) and class images.
(ii) Preprocessing images.
(iii) Converting training data into Numpy array format.
(iv) Encode labels (One-hot encoding).
(v) Partition of dataset into training (80%) and testing set (20%) using scikit-learn library .

Preparing MobileNetV2 for fine-tuning:

(i) Load Mobilenet with pre-trained ImageNet weights, leaving off head of network.
(ii) Construct a new FC hesd, and append it to the base in place of the old head.
(iii) Freeze the base layers of the network. The weights of these base layers will not be updated during the process of backpropagation, whereas the head layer weights will be tuned.

Step 2: Compile and train face mask detector network

(i) Compiling model with Adam optimizer, a learning rate decay schedule and binary cross entropy.

(ii) Training the head of the network using model.fit method and its required arguments.

Evaluating the resulting model on the test set by making predictions on the test set, grabbing the highest probability class label indices. Then, print a classification report for inspection.

Step 3: We then serialize the face mask classification model to disk. Also, plot the accuracy and loss curves.

Phase 2: Deployment

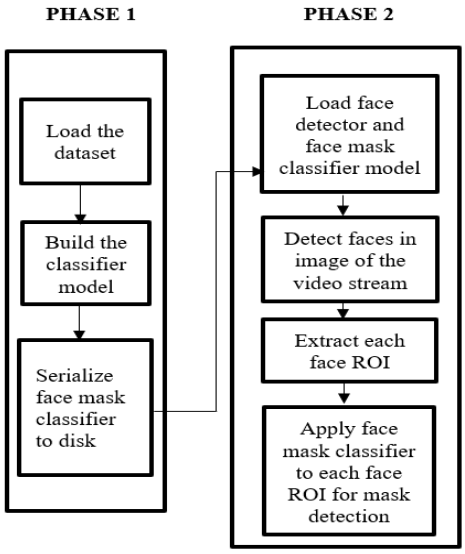
(i) Load both the face detector and face mask classifier models. In the next step, load and preprocess an input image. Pre-processing is handled by OpenCV's blobFromImage function.
(ii) Then loop over detections and extract the confidence in measuring against the confidence threshold.
(iii) Compute bounding box value for a

particular face and make sure that the box falls within the boundaries of the image.

(iv) Next, run the face ROI (extracted through Numpy Slicing) through MaskNet model.

(v) After that, determine the class label based on probabilities given by the mask detector model and assign an associated color for the annotation. The color here will be "green" for with_mask and "red" for without_mask. Then draw the label text, bounding box rectangle for the face, using OpenCV drawing functions. Once all detections have been processed, the output will be displayed.

FLOWCHART



IV MASKED FACE RECOGNITION

Masked Face Recognition

We are proposing a reliable method that will discard masked region on face and used deep learning-based features to address the problem of the masked face recognition process. The first step is to eliminate the masked face region. Next, we apply pre-trained deep Convolutional Neural Networks (CNN) to extract the best features from the obtained regions (mostly eyes and forehead regions). Finally, the Bag-of-Features paradigm is applied on the feature maps of the last convolutional layer to quantize them in order to get a slight representation comparing to the fully connected layer of classical CNN. Finally, MultiLayer Perceptron (MLP) applied for the classification process.

We localize the mask region by applying a cropping filter in order to obtain only the informative regions of the masked face (i.e.forehead and eyes). Nest, we describe the selected regions using a deep learning model. This strategy is more desirable in real-world applications compared to restoration approaches.

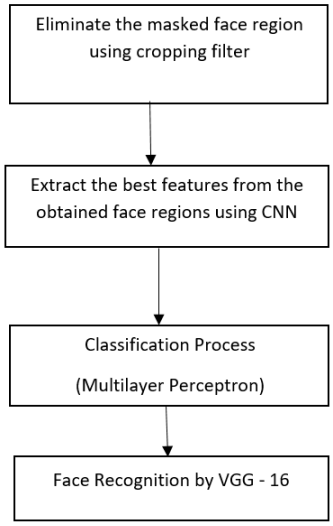
In this paper, we present an efficient quantization based pooling method for face recognition using the VGG-16 pre-trained model. Therefore, we consider only the feature maps at the last convolutional layer using Bag-of-Features (BoF) paradigm.

The basic idea of the classical BoF paradigm is to illustrate images as orderless sets of local features. To get these sets, the first step is to extract local features from the training images, each feature represents a region from the image. Next, the whole features are quantified to compute a codebook. Test image features are then assigned to the nearest code in the codebook to be depicted by a histogram. In this paper, BoF is considered as a pooling layer in our trainable convolutional layers which aims in reducing the number of parameters and making it possible to classify masked face images.

This deep quantization technique presents many advantages. It ensures a lightweight representation that makes the real-world masked face recognition process a feasible task. Moreover, the masked regions vary from the face to another, which leads to informative images of different sizes. The proposed deep quantization allows classifying images from different sizes in order to handle this issue. In addition, the Deep BoF approach uses a differentiable quantization scheme that enables simultaneous training of both the quantizer and the rest of the network, instead of using fixed quantization merely to minimize the model size. It is worth stating that our

proposed system consist of RBF neurons, each neuron is referred to a codeword.

FLOWCHART



THE PROPOSED METHOD

Preprocessing and cropping filter

The images of this dataset are already cropped around the face, so we don't need a face detection stage to localize the face from each image. However, we need to precise the rotation of the face so that we can eliminate the masked region efficiently.

The next step is to apply a cropping filter to extract only the non-masked region. Finally, we exclude the rest of the numbers of the blocks .

Feature extraction layer

We extract deep features using VGG-16 face CNN descriptor from the 2D images. It is trained on the ImageNet. Its name VGG-16 was derived from the fact that it has 16 layers. It contains different layers including convolutional layers, Max Pooling layers, Activation layers, and Fully connected layers. There are 13 convolutional layers, 5 Max Pooling layers, and 3 Dense layers which sum up to 21 layers in total but only 16 weight layers. In this work, we only examine the feature maps at the last convolutional layer, also called channels. These features will be used in the following quantization stage.

Deep bag of features layer

From the image, we extract feature maps using the feature extraction layer described. In order to measure the similarity between the extracted feature vectors and the codewords also called term vector, we applied the RBF kernel as similarity metric. Thus, the first sublayer will consist of RBF neurons, each neuron is referred to a codeword.

The size of the extracted feature map determines the number of the feature vectors that will be used in the BoF layer. Feature vectors are to be fed into the quantization step using the BoF paradigm. To build the codebook, the initialization of the RBF neurons can be carried out manually or automatically using all the extracted feature vectors overall the dataset. The most used automatic algorithm is definitely k-means.

Fully connected layer and classification

Once the global histogram is computed, we pass to the classification stage to assign each test image to its identity. To do so, we apply the Multilayer Perceptron classifier (MLP) where each face is represented by a term vector. Deep BoF network can be trained using back-propagation and gradient descent. Test faces are defined by their codeword . MLP uses a set of term occurrences as input values and associated weights and a sigmoid function that sums the weights and maps the results to output.

CONCLUSION

In this paper, we have concluded that use of mobilenet for mask detection and CNN , MLP classifier for face recognition of masked face gave better results. Since the existing traditional face recognition techniques failed in recognising the masked faces, our proposed system will recognise masked faces in a much efficient way. We addressed the issue of recognizing masked faces through existing face recognition. The proposed system will detect the mask in face in video stream and if it succeed then the image will be given as input. The create mask model will append the mask on the dataset images. This results in the creation of a large dataset of masked faces. The dataset generated with this can then used towards training an effective facial recognition system with target accuracy for masked faces.

REFERENCES

[1] Xiao Han; Qingdong DuResearch on Face recognition based on Deep learning.
[2]Weihong Wang, Jie Yang, Jianwei Xiao, Sheng LiDixin Zhou Face Recognition Based on Deep Learning.
[3]Antonio J.Colmenarez, Thomas S.Huang Face Detection And Recognition.
[4]Hong Zhao, Xi-Jun Liang, Peng Yang Research on Face Recognition Based on Embedded System.
[5]Diaa Salama AbdELminaam, Abdulrhman M. Almansori, Mohamed Taha, Elsayed Badr A deep facial recognition system using computational intelligent algorithms.