DOI: 10.15439/2020F143
ISSN 2300-5963 ACSIS, Vol. 21

Face Mask Detection at the Fog Computing Gateway

Srinivasa Raju Rudraraju
School of Computer and Information Sciences
University of Hyderabad
Hyderabad, India
Email: r.srinivasaraju@gmail.com

Nagender Kumar Suryadevara
School of Computer and Information Sciences
University of Hyderabad
Hyderabad, India
Email: nks@uohyd.ac.in

Atul Negi
School of Computer and Information Sciences
University of Hyderabad
Hyderabad, India
Email: atul.negi@uohyd.ac.in

***Abstract*—This work proposes a fog computing-based face mask detection system for controlling the entry of a person into a facility. The proposed system uses fog nodes to process the video streams captured at various entrances into a facility. *Haar-cascade-classifiers* are used to detect face portions in the video frames. Each fog node deploys two MobileNet models, where the first model deals with the dichotomy between *mask* and *no mask* case. The second model deals with the dichotomy between *proper mask wear* and *improper mask wear* case and is applied only if the first model detects mask in the facial image. This two-level classification allows the entry of people into a facility, only if they wear the mask properly. The proposed system offers performance benefits such as improved response time and bandwidth consumption, as the processing of video stream is done locally at each fog gateway without relying on the Internet.**

I. Introduction

THE usage of face mask by the general public to impede the spread of the Corona Virus pandemic is highly essential. Wearing face mask limits the spread of virus through droplets, such as saliva or mucus [1]. Automatic entry and access control systems based on face mask detection are of immense help at several places such as workplaces, railway stations, shopping malls. These systems help in restricting the entry of persons not wearing a mask to a facility without manual intervention.

Fog Computing is a decentralized computing and storage infrastructure that brings processing closer to the data origin [2],[3]. It addresses the response time needs of real-time Internet of Things (IoT) applications, where Cloud-based IoT is not an ideal choice. Fog Computing disperses a portion of computation and storage from the cloud to devices at the network edge thereby improving response time, bandwidth consumption, data protection, and security of the IoT applications [4],[5].

The computation, storage, and bandwidth requirements of the face mask detection system increase with more number of such units being installed at several points in a facility. To address this problem, we proposed a fog computing-based face mask detection system for entry and access control. The Raspberry Pi (RPi) Camera attached to the RPi device is installed at each entrance of the facility to capture the video. The RPi fog node processes the video frames to track faces in the video stream. The *MobileNet* face mask detection models, deployed in each fog node, classify whether a person wears the mask properly or not using the facial images from the video frame, thereby controlling entry into the facility.

Our work attempts to meet the following objectives:

• A fog computing based face mask detection system to improve the response time and bandwidth consumption.

• Entry restriction to a facility based on the outcome of the face mask detection model.

II. Related Work

A. Entry and Access Control using Face Recognition

A. Nag *et al.* designed a face recognition based door access control in the IoT environment [6]. OpenCV functionality is used to detect and recognize faces of known people and thereby managing the door access automatically. P. Hu *et al.* proposed fog computing-based face identification and recognition scheme that tries to offload face recognition task from cloud to fog nodes [7].

B. Face Mask Detection Systems

Few works are developed to detect people wearing a face mask or not [8],[9]. These techniques have considered two classes of facial images for training the model: with mask and without the mask. The developed systems vary in terms of the framework and model chosen to build the model.

Our proposed system applies face mask detection models at two levels to deal with no mask, proper mask wear and improper mask wear cases separately for automatic entry and access control. The proposed system makes use of fog computing environment for the inference process to obtain performance benefits.

III. System Description

The proposed system employs a RPi fog node integrated with Raspberry Pi Camera and Relay Sensor at each entrance where entry control is required. The fog nodes are connected to the same Wi-Fi network. The basic architecture of the proposed system is shown in Fig. 1.

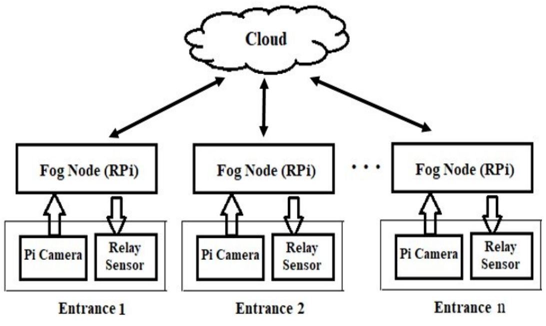

Fig. 1 The basic architecture of the proposed system

The frames in the video stream captured by Pi Camera are processed by the fog node. Whenever any face(s) is detected in the frame, the face mask detection model tries to identify whether the person(s) is wearing the mask or not. The RPi module sends a control signal to the relay to open the door, if the person wears the mask properly. The decision to open the door or not is taken completely on the fog node, and the event information can be sent to the Cloud optionally for further storage and processing. The various software used in the system design are explained below:

A. OpenCV

OpenCV (Open Source Computer Vision) is a library that has several built-in functions for performing computer vision and machine learning tasks [10]. The proposed system uses the *frontal face haar cascade* classifier from OpenCV to detect faces in the video frames.

B. Keras MobileNetV2

Keras is an open-source neural network (NN) library that provides a high-level API wrapper to TensorFlow [11],[12]. MobileNetV2 model is built in our experiment using Keras API, as it is a lightweight convolutional NN that reduces the inference cost on mobile and embedded devices [13],[14].

C. Basic Operation of the Proposed System

The proposed system employs two MobileNetV2 models for classifying whether a person is wearing the mask properly or not. The first model is a binary classifier that is trained using two classes of images – ***mask*** and ***no mask***. The ***mask*** class contains facial images of persons wearing a mask (includes both proper and improper mask wear). The second model is a binary classifier that is trained using ***proper mask wear*** and ***improper mask wear*** images. Mask is said to be properly put on if nostrils and mouth are covered. If nostril or mouth is detected even when the person wears a mask, the instance is classified as improper mask wear and entry should be restricted. In our experimental setup, these two models are trained on a single RPi fog node and deployed in all the fog gateways for inference purposes. The rationale behind choosing two binary classifiers instead of a single three-class classifier is to improve the classification accuracy, especially between *proper* and *improper mask wear* classes. The choice provides a tradeoff between classification accuracy and throughput.

Each fog node processes the video frames captured by Pi Camera and uses *Frontal Face Haar Cascade* classifier to detect the faces in those frames. When one or more faces are detected in a frame, level one binary classifier (***mask*** vs. ***no mask***) is applied on each face region of interest (ROI) in the video frame. If the person wears the mask, then level two classifier (***proper*** vs. ***improper mask wear***) is applied to identify whether the person is wearing the mask properly or not. This two-level classification restricts the entry of people with improper mask wear into the facility. The basic operation of the proposed system is shown in Fig. 2.

IV. IMPLEMENTATION DETAILS

The system provides a proof-of-concept for face mask detection using fog computing gateway. The processing of the video frames is done at the source of video capture. In reality, the frames in the video stream could be sent and processed using the resources in the fog gateway.

A. Face Mask Detection Model Training

Datasets used for Model Training: As discussed in Section III, the proposed system uses two binary classifiers based on the MobileNetV2 model. Classifier-1 (model-1) is trained using dataset-1 with a total of 770 facial images divided into two classes: *with mask* and *without mask*. The "*with mask*" class included images of faces with and without proper face mask wear. Classifier-2 is built using dataset-2 with a total of 500 facial images divided into two classes: *proper mask wear* and *improper mask wear*. The average size of training images is around 5KB (Image dimensions 250x160 with 96 dpi). Fig 3 and Fig 4 show a sample set of facial images from dataset-1 and dataset-2 respectively. Even though there are few face mask datasets available online [15], we have prepared our own dataset as the existing face mask datasets lack images for *improper mask wear* class.

Training the mask detection models: The various steps followed to train the mask classification models are given below:

Step 1. Load the images from the dataset using *load_img()* function from Keras API. The loaded images are resized to 224x224 format.

Step 2. The loaded images are normalized using *preprocess_input()* function. The data (facial image) and label lists are updated with images in the dataset.

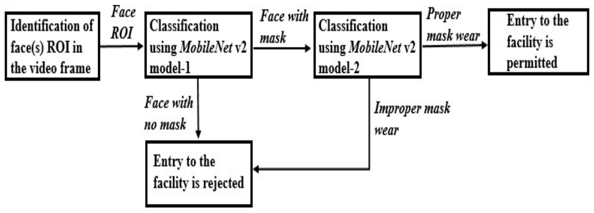
Identification of face(s) ROI in the video frame
Face ROI
Classification using MobileNet v2 model-1
Face with mask
Classification using MobileNet v2 model-2
Proper mask wear
Entry to the facility is permitted
Face with no mask
Improper mask wear
Entry to the facility is rejected

Fig. 2 The basic operation of the proposed system

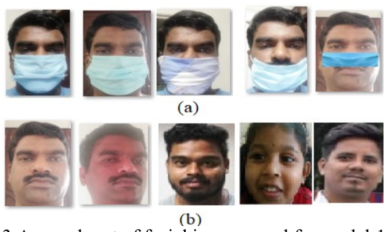
Fig. 3 A sample set of facial images used for model-1training (a) with mask (b) without the mask

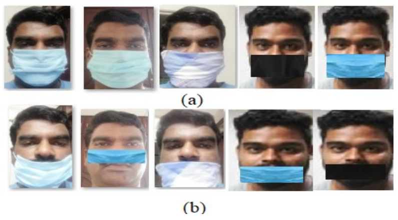
Fig. 4 A sample set of facial images used for model-2 training (a) with proper mask wear (b) without proper mask wear

Step 3. Convert the data and labels into *numpy* arrays. One-hot encoding is performed on the labels to represent them as binary vectors.

Step 4. The dataset is partitioned into training (80%) and testing (20%) sets using *train_test_split()* function.

Step 5. Instantiate MobileNetV2 model trained with ImageNet dataset. Load the model that doesn't include the classification layers at the top. Transfer learning is used in our experimental setup to transfer knowledge from the ImageNet dataset domain to our Facial dataset domain [16].

Step 6. The weights of all the layers in the convolutional base are frozen to prevent updates during training. We added the classifier on top of this base model and trained the top-level classifier.

Step 7. The model is compiled using *Adam optimizer* and *binary cross-entropy* loss function.

Step 8. The model is trained and serialized to disk for the usage during the inference process.

B. Classification of Facial Images

RPi 4 device integrated with Pi Camera is used as a fog node at each entrance to capture the video stream. The Pi Camera has 5MP resolution and can record 1080p videos at 30 frames per second. The face mask detection models trained in the previous step are loaded in each fog node for inference. The following are the various steps performed during the inference process on each video frame:

Step 1. Identification of face ROI in the frame using *frontal face haar cascade* classifier from OpenCV.

Step 2. If more than one face is detected in the frame, for each face do the following:

- Resize the face image to 224x224 RGB image.
- Convert the image into ***numpy*** array and preprocess it.
- Predict the output class (mask vs. no mask) of the image using the MobileNetV2 model-1 loaded in the node.
- If the prediction class on the image is ***mask***, then model-2 is used for further classification.
- If the prediction class by model-2 is "proper mask wear", then fog node controls the relay to open the door. Otherwise, entry is restricted.

Fig. 5 shows the screenshots of the outcome of the face mask detection model for proper and improper mask wear cases.

V. Results and Discussion

The face mask detection models (model-1 and model-2) are trained using different learning rates (LR) 0.001 and 0.0001 with two different numbers of epochs 10 and 20. The model-1 and model-2 training tasks have taken 34 minutes and 20 minutes respectively (with LR = 0.001 and #Epochs = 20) on RPi 4. The face detection and inference tasks for the given video frame have taken 0.3 seconds and 3.4 seconds respectively on average. Table I and Table II present the accuracy and loss values during training and validation phases of model-1 and model-2 respectively (with LR = 0.001 and #Epochs = 20).

From the results in Table I and Table II, we can observe that training, validation accuracy and loss values are improving during the models' training. After few number of epochs (epoch #15 for model-1 training, and epoch #10 for model-2 training approximately), we get fluctuations in the accuracy and loss values. The variations in the accuracy and loss values are due to *model overfitting* with more number of epochs. Overfitting of model could happen with the selection of small dataset for training the model. As the RPi device is resource-constrained, we have chosen small datasets of facial images for model training. One solution to address this problem is *early stopping* using *callback* mechanism.

While training model-1 on the Raspberry Pi node using dataset with 770 images, a warning message is received with respect to allocation of memory exceeding 10% of system memory. If we want to train the model with a large dataset to avoid model overfitting, we can train it on a high-end machine and deploy the model on the fog node for inference. Alternatively, we can distribute the model training to several nodes in the fog cluster. Fig. 6 presents performance metrics related to model-1 and model-2 training.

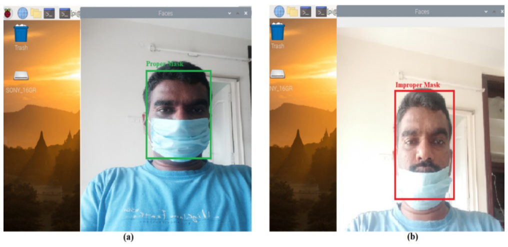

Fig. 5 Face mask detection model prediction when (a) the person wears the mask properly (b) the person wears the mask improperly

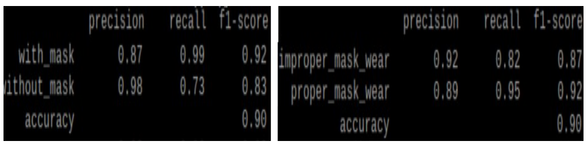
```
             precision   recall  f1-score
   with_mask      0.87     0.99      0.92
ithout_mask      0.98     0.73      0.83
    accuracy                         0.90
```
```
                    precision   recall  f1-score
improper_mask_wear       0.92     0.82      0.87
  proper_mask_wear       0.89     0.95      0.92
          accuracy                          0.90
```
Fig. 6 Performance metrics related to model-1 and model-2 training

TABLE I.
ACCURACY AND LOSS VALUES DURING MODEL-1 TRAINING

Epoch	Train_Loss	Train_Acc	Val_Loss	Val_Acc
1	0.64	0.69	0.49	0.71
2	0.39	0.82	0.39	0.82
3	0.36	0.88	0.48	0.74
4	0.35	0.88	0.47	0.74
5	0.32	0.89	0.45	0.75
6	0.24	0.9	0.47	0.74
7	0.22	0.91	0.46	0.78
8	0.21	0.92	0.44	0.79
9	0.21	0.91	0.43	0.79
10	0.19	0.93	0.44	0.78
11	0.19	0.92	0.41	0.8
12	0.18	0.93	0.34	0.82
13	0.17	0.94	0.4	0.78
14	0.17	0.93	0.35	0.83
15	0.17	0.94	0.42	0.81
16	0.18	0.93	0.47	0.77
17	0.17	0.93	0.56	0.75
18	0.16	0.94	0.57	0.74
19	0.17	0.93	0.54	0.76
20	0.16	0.94	0.55	0.75

TABLE II.
ACCURACY AND LOSS VALUES DURING MODEL-2 TRAINING

Epoch	Train_Loss	Train_Acc	Val_Loss	Val_Acc
1	0.71	0.66	0.44	0.77
2	0.5	0.72	0.34	0.79
3	0.42	0.79	0.32	0.8
4	0.39	0.8	0.28	0.85
5	0.35	0.88	0.29	0.84
6	0.32	0.89	0.29	0.9
7	0.22	0.91	0.27	0.88
8	0.22	0.9	0.26	0.9
9	0.2	0.92	0.24	0.91
10	0.29	0.85	0.25	0.9
11	0.28	0.89	0.28	0.88
12	0.25	0.9	0.29	0.88
13	0.24	0.92	0.3	0.85
14	0.25	0.9	0.32	0.84
15	0.22	0.92	0.35	0.82
16	0.21	0.92	0.38	0.78
17	0.22	0.93	0.39	0.76
18	0.21	0.91	0.38	0.77
19	0.19	0.92	0.37	0.8
20	0.19	0.92	0.38	0.78

VI. CONCLUSION AND FUTURE WORK

This work presented a proof-of-concept fog computing-based face mask detection system for automatic entry and access control into a facility. The fog gateway processes the video stream captured at the entrance to recognize whether a person is wearing a mask or not. Two *MobileNet* models are deployed in each fog node located at each entrance to the facility. The first model deals with the dichotomy between mask and no mask case. The fog node uses the second model to detect whether the person wears the mask properly or not, in case the first model detects the person wearing the mask. The proposed system allows entry into the facility, only if the person wears the mask properly. The results of the classification are encouraging with model accuracy value around 90%.

As the processing of video stream is done locally at each fog node, the proposed system offers performance benefits such as improved response time and bandwidth consumption. The proposed system could be integrated with Cloud, optionally, for further storage and processing of video stream. In the future, the system could be extended to distribute the model training among several fog nodes in the network.

REFERENCES

[1] J. Howard, A. Huang, Z. Li, Z. Tufekci, V. Zdimal, H. van der Westhuizen *et al.*, "Face Masks Against COVID-19: An Evidence Review", Preprints 2020, 2020040203, http://dx.doi.org/10.20944/preprints202004.0203.v1

[2] S. Sarkar, R. Wankar, S. Srirama and N.K Suryadevra, “Serverless Management of Sensing Systems for Fog Computing Framework”, IEEE Sensors Journal, ISSN: 1530-437X, 20(3):1564-1572, 2020, http://dx.doi.org/10.1109/JSEN.2019.2939182

[3] S. R. Rudraraju, N. K. Suryadevara and A. Negi, "Face Recognition in the Fog Cluster Computing," IEEE International Conference on Signal Processing, Information, Communication & Systems, 2019, pp. 45-48, http://dx.doi.org/10.1109/SPICSCON48833.2019.9065100

[4] N. N. Khan, “Fog computing: A better solution for IoT,” International Journal of Engineering and Technical Research., vol. 3, no. 2, pp. 298–300, 2015, ISSN: 2321-0869

[5] M. Aazam, S. Zeadally, and K. A. Harrass, “Fog Computing Architecture, Evaluation, and Future Research Directions”, IEEE Communications Magazine (2018), pp. 46-52

[6] A. Nag, J. N. Nikhilendra, and M. Kalmath, "IOT Based Door Access Control Using Face Recognition", International Conference for Convergence in Tech., http://dx.doi.org/10.1109/i2ct.2018.8529749

[7] P. Hu, H. Ning, T. Qiu, Y. Zhang, and X. Luo, "Fog Computing Based Face Identification and Resolution Scheme in Internet of Things", IEEE Transactions on Industrial Informatics, 13(4), 1910–1920, 2017, http://dx.doi.org/10.1109/tii.2016.2607178

[8] COVID-19: Face Mask Detector, Last accessed 05 Jun 2020, URL:https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/

[9] Face Mask Detection, Last accessed 10 Jun 2020, URL: https://www.towardsdatascience.com/covid-19-face-mask-detection-using-tensorflow-and-opencv-702dd833515b

[10] OpenCV, Last accessed 19 May 2020, URL: https://opencv.org/ about

[11] Keras Guide, Last accessed 9 May 2020, URL: https://keras.io/guides/

[12] TensorFlow Tutorials, Last accessed 10 Jun 2020, URL: https://www.tensorflow.org/tutorials

[13] MobileNet, Last accessed 20 May 2020, URL: https://www.tensorflow.org/api_docs/python/tf/keras/applications/mobilenet

[14] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand *et al.*, “Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv:1704.04861, 2017

[15] Face Mask Dataset, Last accessed 21 Jun 2020, URL: https://github.com/X-zhangyang/Real-World-Masked-Face-Dataset

[16] Transfer Learning, Last accessed 21 May 2020, URL: https://www.tensorflow.org/tutorials/images/transfer_learning