Masked Facial Recognition in Security Systems Using Transfer Learning

M. Ramgopal1 · M. Sai Roopesh1 · M. Veeranna Chowdary1 · M. Madhav1 · K. Shanmuga1

Received: 28 May 2022 / Accepted: 2 September 2022 / Published online: 22 October 2022
© The Author(s), under exclusive licence to Springer Nature Singapore Pte Ltd 2022

Abstract
The COVID-19 is a crisis of unprecedented magnitude, which has resulted in countless casualties and security troubles. In view of recent events of corona virus people are required to wear face masks to protect themselves from getting infected. As a result, a good portion of face (nose and mouth) is hidden by the mask and hence the facial recognition becomes difficult. Many organizations use facial recognition as a means of authentication. Researchers focus on developing rapid and efficient solutions to deal with the ongoing coronavirus pandemic by coming up with suggestions for handling the facial recognition problem. This research paper aims to identify the person, while the face is covered with a facial mask with only eyes and forehead being exposed. The first step involves marking the facial region. Next, using the data set, we will implement an object detection model YOLOv3 to identify unmasked and masked faces. The YOLO v3 object detection model is the best performing model with a detection time of 0.012 s, F1 score of 0.90 and mAP score of 0.92. Experimental results on Real-World Masked-Face-Data set show high recognition performance.

Keywords Machine learning (ML) · Convolutional neural networks (CNN) · Face recognition · Transfer learning model · Object detection model

Introduction
With the recent outbreak of a virus called COVID-19 we came to know that what we use as security for giving access-like biometrics is also a means to spread the COVID-19 virus. As the virus can be spread through contact or contaminated surfaces, we can no longer believe that it is safe to use the normal fingerprint scanner or palm scanner to give access to authorized personnel [1].

So we have to resort to a way, where there is no contact while giving access, we can assume that we can use our face recognition for giving access but research on the COVID-19 virus proved that virus may be airborne. Research suggests that we need to wear a face mask all the time as to minimize the risk of being affected by the virus [2]. Hence, comes the problem with the face recognition we have used so far. As we cover our face with a face mask, the nose and mouth area will be covered by the mask and only the eyes and forehead are visible. Since our usual face recognition algorithm uses nose and mouth areas to recognize a face, we might find it difficult to recognize the face with the usual techniques. Covering the face also creates more problems as thieves are taking advantage of the mask for doing fraudulent activities.

Therefore, the need for masked face recognition is increasing rapidly. We came up with an approach to solve this particular problem. We are trying to use an object detection algorithm called “Yolo” to try to recognize the masked face [3].

Data Understanding
Data are the building block of any project, and therefore, an accurate knowledge of the data is very necessary before building the model. The data set of this project consists of images of 4 user face data. The data for each user’s face include equal numbers of masked and non-masked images. Since the goal of the project is to identify the individual user wearing a face mask, there is no competent image data set
that includes the faces of masked individuals and their identities. Hence, the need to create a custom data set. Therefore, a new data set was created with specific labeling for YOLO neural networks. The data set used for the search will contain the facial data of four different users. Each person's clear face and masked face will be used to train the model. This way the model will learn the identity of the person with and without a face mask. The data of each user must be collected with a clear background with a bright light source [4].

**Data Preparation**

The creation of the data set involves the facial data of 4 people in this research. A video of approximately 15 s of each user is captured, where the video consists of looking at the user from four directions (up, down, left and right) which capture all facial features. This action will be performed twice by a user in the video wearing a mask and with a clear face. Once the video is captured, it is processed and converted into multiple frames, then the frames are filtered for the required number of frames per person, then the images are labeled based on the user's identity using LabelImg.

LabelImg is a labeling tool which is used to label the images to create the custom data set for different object detection models. In LabellImg, each label creates a text file with the same name that of the image file, where text file contains the X and Y co-ordinates of the bounding boxes in the image associating to the respective class of the bounding box. The text label file contains class number, x_center, y_center, width, height (Fig. 1) [5].

![Labeling of an image using labeling tool](image1.png)

In addition, for labeling the Labeling tool must be supported by a class file which contains the list of all classes to be labelled in the data set. For example, if we are preparing a data set for 4 class objects then the root directory must have a file named classes.txt that contains class names of each class object along with the class number (Fig. 2) [8].

The images and text files that are generated in labeling process needed to be divided into train and test data sets. The folder structure must include folder named images contains images of the people and a folder named label contains of the text files that represents the labeling of images present in images folder further the images and label data are divided into train and test sets.

**Modeling**

In this section, we discuss the transfer learning model and the yolov3 model.

**Transfer Learning**

An efficient way to solve problems related to unstructured data is “Deep Learning”. The most popular field researched
under the deep learning technique is Natural Language Processing.

With Deep Learning there are a few limitations:

1. Rather than a small volume of data, a deep learning model requires huge amount of unstructured data.
2. Deep Learning requires more computation power and takes longer CPU cycles and time, since the training process takes hours to complete even on high-performing machines.
3. High training time, since the training time is dependent on the amount of data to process and the number of layers in the network.

To overcome the major limitation of deep learning, which is high training time, the technique of transfer learning came into the picture.

Transfer learning models come with pre-trained neural network weights that are already learnt from the image data set.

These models can perform recognition immediately and can be further trained for specific image data sets. They are especially efficient for image recognition and natural language processing.

Yolo Objection detection technique was chosen for this paper.

**YOLO Object Detector**

Joseph Redmon developed a real time object detection system called “YOU ONLY LOOK ONCE” (YOLO). YOLO processes the detection faster than any other object detection model. It is run on “Darknet”, a neural network framework used as a backbone for training and testing the CV models. YOLO models scale an image by applying locations, and the high scoring region is considered detection among other regions. Yolo models proved to better than ranking-based models. Unlike RCNN models, which require thousands of network scores for a single image, Yolo models make predictions using a single network score.
This technique helps the model to perform simultaneous detection with multiple bounding boxes along with class probabilities for the bounding boxes (Medium, 2018).

This type of fast and seamless recognition paves the way for future innovations in image processing.

**Evaluation**

Evaluation of a model is the process through which we quantify the quality of a prediction given by the system. According to YOLO's documentation, each model needs to learn at maximum of 2000 iterations per class. The model creates an output chart file which plots both the mean average precision percentage and the average loss (IOU). On a graph with an x-axis indicating the number of training iterations and an y-axis indicating the average loss percent [9].

During training, at every 100th iteration, a weights file is created by the model. We can also customize the iteration in which we want to save the file. Weights file is a binary convolutional network file that is produced by the model that contains weights at each layer at that iteration. To develop this custom weights file, initially a corresponding pre-trained weights (Darknet). After all the iterations are completed, the model contains three weight files, namely, best weights, last weights and final weights. Among which "best weights" is always chosen to perform prediction as it is associated with a higher mean average percentage [10].

We utilized the graph processing unit provided by the "Google Colaboratory" to train the model developed. Google Colab provides free GPU for about 12 h a day.

**Evaluation Parameters**

1. Mean average precision (mAP)
2. Intersection over union (IOU)

**Mean Average Precision**

Average precision (AP) is a famous metric in measuring the degree of accuracy of object detectors. Average precision calculates the average precision value for recall value over zero to one [11].

**Precision and Recall**

Precision quantifies the accuracy of the model prediction, i.e., the percentage of your accurate predictions [12].

Recall estimates how acceptable you find all the positives.

\[
\text{Precision} = \frac{TP}{TP + FP}
\]

\[
\text{Recall} = \frac{TP}{TP + FN}
\]

\[
F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}
\]

\[
TP = \text{True positive}
\]

\[
TN = \text{True negative}
\]

\[
FP = \text{False positive}
\]

\[
FN = \text{False negative}
\]

**Intersection over Union**

Intersection over Union is an evaluation metric used to quantify the accuracy of an object detector on a specific data set.

To apply Intersection over Union to evaluate an object detector we need. The ground-truth bounding boxes (i.e., the hand labeled bounding boxes from the testing set that specify where in the image our object is) [12].

The predicted bounding boxes from our model.

Intersection over Union is simply a ratio between the Area of Overlap and Area of Union.
An Intersection over Union score > 0.5 is normally considered a “good” prediction.

Results

The evaluation time and mean average accuracy (mAP) of the yolov3 model are generated. Training a model is done by plotting the percentage of the map at specific iteration checkpoints. According to YOLO documentation (arXiv. 2020), each model should learn a maximum of 2000 iterations per class. Since there are 4 classes (4 user faces) for detection in this research, each model needs to be trained up to 12,000 iterations.

The model creates an output graph file that plots both the percentage of the map and the average loss in the graph with the x-axis indicating a number of training iterations and the y-axis indicating the percentage of average loss. During learning, a file of weights is created at each hundredth iteration. The file is a binary convolutional network file created from the model. A corresponding pre-trained weight is used to develop a custom weight file from the model through training. After the last iteration, the model will have three types of weights, namely, the best weights, the last weights and the final weights. Among which the best weight is always chosen to make the prediction, because it is associated with the highest card percentage during the iterations.

YOLO models are made of 53 convolutional layers. Where 53 convolutional layers, summing to 106 layers of architecture for YOLO v3, where detections are made at three layers (82, 94, and 106).

Fig. 3 YOLO bounding box predicting the persons
In YOLO object detection, a class means the category of an object. Number of class refers to the total number of objects used in the model for developing the neural weights. For this research 4 people are used for model training. The object name of each person will be the person’s name. Hence, the bounding box around the person’s face will be displayed by the name of the person in the results (Fig. 3). The bounding box in the prediction displays name of the person with confidence level in the shown image.

mAP calculation for YOLO V3: calculation mAP (mean average precision)
Detection layer: 82 - type = 28
Detection layer: 94 - type = 28
Detection layer: 106 - type = 28 1200
unique_truth_count = 1204

class_id = 0,
   name = roopesh, ap = 95.40% (TP = 63, FP =14) class_id = 1,
   name = veeranna, ap = 93.00% (TP = 200, FP =0) class_id = 2,
   name = madhav, ap = 94.50% (TP = 200, FP =0) class_id = 3,
   name = shanmuga, ap = 92.73%(TP = 179, FP =11)
for conf_thresh = 0.25, precision = 0.83, recall = 0.86, F1-score = 0.85
    for conf_thresh = 0.25, TP = 1039, FP = 212, FN = 165,

Average IoU = 68.47 % IoU threshold = 50 %
mean average precision (mAP@0.50) = 88.36

% Total Detection Time: 18
Conclusions

This paper focuses on the ability of YOLO object detector models to recognize the user’s identity.

Identifying the face with a face mask is challenging for an object detector model. Since there is no availability of relevant data set, a new custom data set of 5 users was prepared. Custom data set contains both the masked face and clear face of the corresponding user and a total of 2000 images was prepared (1500 for training and 500 for testing) for this paper. Experiments performed on 1500 images over Tesla K80 cloud GPU, showed that the YOLOV3 model’s detection with F1 score of 0.85 and mean Average Precision of 88.36%. Model was trained for about (100) epochs. We considered YOLOV3 as it overcomes the setbacks of accuracy and time limitations of other versions. The considered model has a detection time of 18 s, F1 score of 0.86 and mean average precision score of 90.3%. As a result, YOLOV3 is considered the best performing model among the other versions of YOLO family with accurate and fast masked face detection which can be deployed over the lower computing environment of smartphone.

Funding  The authors declare that they have not received any funding.

Declarations

Conflicts of interest  The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Dollar P, Wojek C, Schiele B, Perona P. Pedestrian detection: an evaluation of the state of the art. IEEE Trans Pattern Anal Mach Intell. 2012;34:743–61.
2. Liu L, Ouyang W, Wang X, Fieguth P, Chen J, Liu X, Pietikainen M. Deep learning for generic object detection: a survey. 2018. arXiv preprint arXiv:1809.02165
3. Xie S, Girshick R, Dollar P, Tu Z, He K. Aggregated residual Transformations for deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1492–1500, 2017.
4. Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K. Squeezenet: Alexnet- level accuracy with 50x fewer parameters and 0.5 mb model size. 2016. arXiv preprint arXiv:1602.07360
5. Wang RJ, Li X, Ling CX. Pelee:a real-time object detection system on mobile devices. In: Advances in neural information processing systems, pp. 1963–1972, 2018
6. Qureshi S. A survey of the recent architectures of deep convolutional neural networks. 2019. arXiv preprint arXiv:1901.06032
7. Li L, Feng X, Xia Z, et al. Face spoofing detection with local binary pattern network. J Vis Commun Image Represent 182–192, 2018
8. S Kim and K Cho. Trade-off between Accuracy and Speed for Pedestrian Detection Using HOG Feature, pp. 207–209, September 2013.
9. Ren S, He K, Girshick R, et al. Faster R-CNN: towards real-time object detection with region proposal networks. Advances in neural information processing systems, pp. 91–99, 2015.
10. Liu W, Anguelov D, Erhan D, et al.: Ssd: single shot multibox detector. In: IEEE ECCV, pp. 21–37, 2016
11. Redmon J, Divvala S, Girshick R, et al. You only look once: unified real-time object detection. In: IEEE CVPR, pp. 779–788, 2016
12. ’Yang S, Luo P, Loy CC, et al. From facial parts responses to face detection: a deep learning approach. In: IEEE ICCV, pp. 3676–3684, 2015

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.