Face Mask Detection for COVID-19 Prevention

ANTHONY COLAS*, University of Florida, USA
YANG BAI, University of Florida, USA
YUE WANG, University of Florida, USA

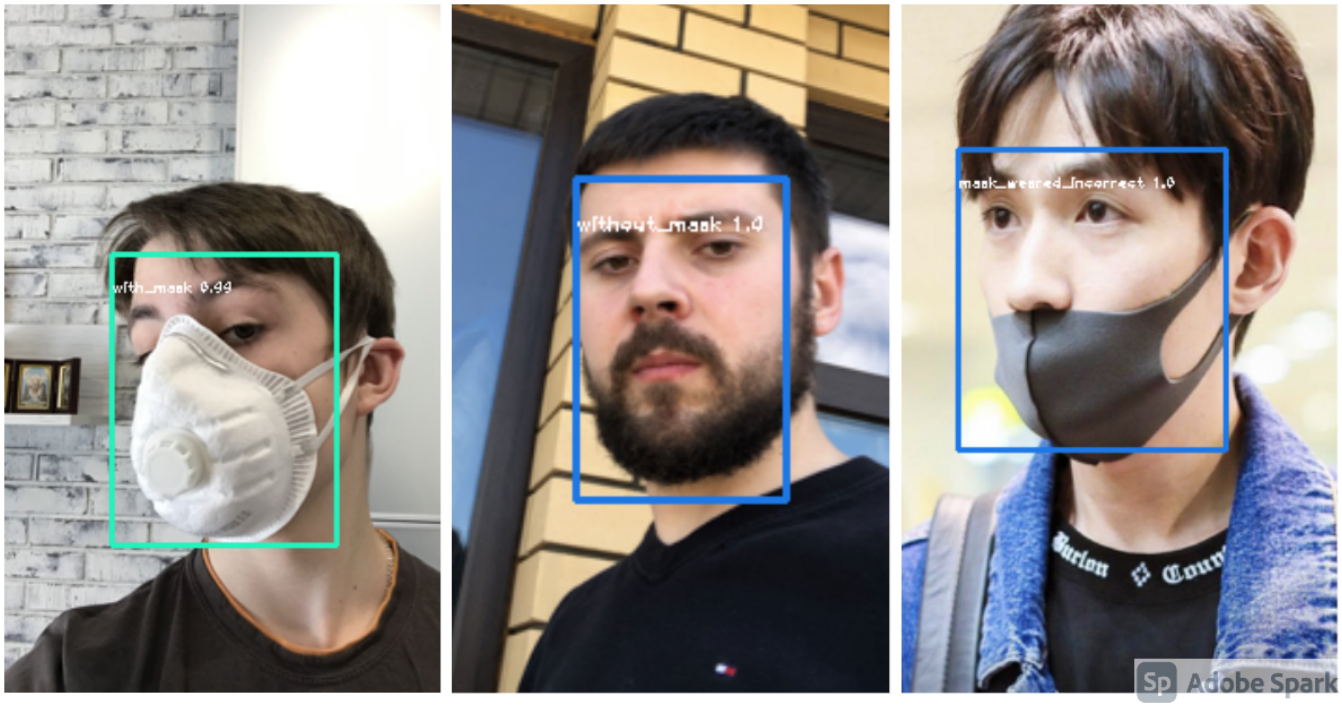


Fig. 1. Results of the Facemask identification for COVID-19 prevention system. These are the results of the YOLOv3 system, trained on the integrated dataset. The left image correctly outputs 'with_mask', the center image correctly outputs 'without_mask', and the right image correctly shows the model identifying when someone is not wearing their mask correctly ('mask_weared_incorrectly).

COVID-19 is the most prominent medical crisis today. To limit the spread, the World Health Organization (WHO) and Centers for Disease Control and Prevention (CDC) has recommended that everyone wear a face-mask when not in isolation. Although recommended, some people may not be wearing their mask correctly. Recent advances in object detection have made in possible to detect product, analyze traffic incidents, and even detect cancer in patient radiology scans. In this work, we analyze and evaluate various state-of-the-art object detection models to determine if someone is wearing a mask correctly, incorrectly wearing a mask, or not wearing a mask at all. Specifically, we will analyze a baseline sliding-window and CNN localization model and the Faster R-CNN and YOLOv3 state-of-the-art models. Further, we build a new face-mask identification dataset by combining and standardizing various open source face-mask data, including synthetic and real-world data. We demonstrate that for the proposed models, YOLOv3 works best on the face-mask detection task in both speed and mean average precision score.

CCS Concepts: • **Computing methodologies** → **Object identification**; • **Applied computing** → *Consumer health*.

Additional Key Words and Phrases: datasets, neural networks, object identification, face-mask

ACM Reference Format:
Anthony Colas, Yang Bai, and Yue Wang. 2020. Face Mask Detection for COVID-19 Prevention. *ACM Trans. Graph.* 37, 4, Article 111 (December 2020), 9 pages. https://doi.org/10.1145/1122445.1122456

*All authors contributed equally to this research.

Authors' addresses: Anthony Colas, acolas1@ufl.edu, University of Florida, Gainesville, Florida, USA; Yang Bai, University of Florida, Gainesville, Florida, USA, baiyang94@ufl.edu; Yue Wang, University of Florida, Gainesville, Florida, USA, yue.wang1@ufl.edu.

0730-0301/2020/12-ART111 $15.00
https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

With over 15 million confirmed cases and over 280,000 deaths (as of the time of this work) [1], the coronavirus (COVID-19) continues to spread at an accelerated rate [2]. On March 11, 2020, the World Health Organization (WHO) even declared the coronavirus a global pandemic. To slow the spread of COVID-19, the Centers for Disease Control and Prevention (DCD) [2] and WHO [3] have both recommended the proper use and wear of face masks. Taking this advice, many local, state, and national laws have made it mandatory to wear face coverings while in public; private businesses have taken it upon themselves to also enforce mask wearing while utilizing their facilities. To validate these precautions, previous studies have been conducted on the efficacy of face masks against the coronavirus [3; 17]. [17] show a correlation between a government mandate of mask usage and a drop in the daily growth of COVID-19. [3] create model simulations which express that even relatively ineffective face masks may meaningfully reduce the spread of COVID-19. While a public mandate on wearing face masks was and remains controversial [29], the evidence is clear that properly wearing face masks can reduce the spread of disease in infected patients [1; 29]. Thus, wearing a mask is an essential step one can take to slow the spread of COVID-19, potentially decreasing hospitalizations and deaths.

While not undercutting the severity and suffering caused by COVID-19, the pandemic has given rise to opportunities for novel AI research in the fields of epidemiology and consumer health. With the abundant amount of data, researchers can better model studies of viral activity in order to guide governmental policy, machine learning algorithms can give initial diagnoses of COVID-19 using deep learning, and medical chat-bots can proxy patient medical facility visits [28]. For instance, AI4COVID-19 utilizes cough samples for preliminary diagnosing COVID-19 [11]. AI on face mask detection has also been recently utilized in the Paris Metro surveillance system [4]. Face mask detection systems can be used by businesses and the public sector in order to ensure that their patrons are properly adhering to COVID-19 prevention guidelines. Furthermore, such systems can assist those who are unsure if their mask is properly worn.

In this work, we focus on *training, testing, and analyzing baseline and state-of-the-art object detection algorithms in order to determine if someone is wearing a face mask*. Thus, given a scene containing one or multiple people, our goal is to detect whether they are wearing a face mask, not wearing a face mask, or incorrectly wearing a face mask. Specifically, for our baseline algorithms we build a primitive sliding-window model with a fully connected network (FCC) classifier and a convolutional neural network (CNN) implementation of the sliding windows algorithm. We then fine-tune on the Faster R-CNN [22] and YOLOv3 [21] pre-trained state-of-the-art object detection models. We further detail the models in section 3, while analyzing our experiments in section 4. We experiment on three publicly available and annotated face mask datasets, which we cleanse, aggregate, and standardize. Our experiments vary across the various publically available datasets, reflecting that for the *Face Mask Detection dataset* [5] YOLOv3 works best. However, further improvements need to be made on our newly integrated and curated dataset.

Our contributions are as follows:

(1) We collect, curate, and combine three publicly available datasets for the face mask detection task. We show that the task's difficulty increases on this integrated dataset compared to solely testing on an individual dataset. Thus, we call for more work on our integrated dataset.
(2) We construct two baseline algorithms: 1) sliding window approach and 2) a CNN-based implementation of a sliding window in order to determine the effectiveness of these early models on the facemask detection task.
(3) We utilize two state-of-the-art models on object detection-Faster R-CNN [22] and YOLOv3 [21]-on the face mask detection task in order to analyze their performance on such task.
(4) We analyze the pros and cons of using such algorithms for the face mask detection task, in hopes that our research is considered when constructing and determining what algorithm to use for a face mask detection surveillance system in order to mitigate the spread of COVID-19.

2 RELATED WORK

The work related to face mask detection can be divided into two spaces: 1) Object detection and 2) Face mask efficacy. First we will review some work on object and face recognition. Note we will review some prominent works in both, but we do not extensively cover the area. For a thorough survey on general object detection see [16], and for a survey on face recognition see [18]. We now briefly review some works on object detection, particularly the YOLOv3 [21] and Faster R-CNN algorithm [22].

2.1 Object Detection

There has been extensive previous works on object detection [9; 15; 26] and face detection [8; 12; 24]. In [26] Szegedy et al. first use deep neural network in order to classify and localize objects, where they forumalte the problem as a regression to object masks. [9] propose an object relation module in order to identify objects where they analyze the different interactions between their features and geometry and see an improvement in the object recognition task. In [15], Kong et al. demonstrates an anchor-free framework for object detection by producing category-agnostic bounding boxes for each potential object in an image. While these models see improvements in the state-of-the-art prototypical object detection frameworks such as YOLOv3 and Faster R-CNN,they do require some augmentations to the annotations in object datasets that may not be easily available. Furthermore, there have been more studies on the applications of YOLO and Faster R-CNN, specifically on medical images containing malaria or cancerous tissues [10; 19; 30].

2.1.1 Faster R-CNN. We now briefly cover the Faster R-CNN architecture. For further reference, we recommend that the reader go

[1] https://covid.cdc.gov/covid-data-tracker/#cases_casesper100klast7days
[2] https://www.cdc.gov/media/releases/2020/p0714-americans-to-wear-masks.html
[3] https://www.who.int/publications/i/item/advice-on-the-use-of-masks-in-the-community-during-home-care-and-in-healthcare-settings-in-the-context-of-the-novel-coronavirus-(2019-ncov)-outbreak
[4] https://www.bloomberg.com/news/articles/2020-05-07/paris-tests-face mask-recognition-software-on-metro-riders
[5] https://www.kaggle.com/andrewmvd/face-mask-detection

over [22]. Unlike the the Fast R-CNN model [6], the *Faster* R-CNN model uses a region proposal network (RPN), thus making it faster than its predecessor. The RPN allows the network to generate object proposals containing different sizes. The RPN defines important regions in an image where an object may be, giving the model initial locations to look for an object. Further, the Faster R-CNN also utilizes anchors boxes, which can be though of as priors or references where object lie. The priors are generated by using k-means clustering on the image dataset to find common locations for object centers. These anchor boxes come in various shapes and sizes and define multiple regions. In order to speed-up the computation time, the RPN and Fast CNN share convolutions.

2.1.2 YOLOv3. We now do the same for YOLOv3 and briefly cover the architecture. For further details, please see [21]. YOLOv3 first uses a feature extractor of 53 layers to embed an image's features, which includes 5 residual blocks. The features from the last 3 residual blocks are used in the multi-scale detector for small, medium, and large features. The output of the multi-scale detector then contains detections at three different scales. Like the Faster R-CNN model, YOLOv3 then uses anchor boxes at each scale, which are also pre-calculated by using k-means. YOLOv3 contains 3 anchor boxes per grid cell, where the grid is the square matrix output after applying a CNN to the entire image. After filtering and determining the best anchor boxes, YOLOv3 predicts the coordinate offset in relation to the anchor box, the objectness probability, and the class probabilities.

2.2 Face Mask Efficacy

There have been many previously worked studies on the efficacy of face masks in limiting the spread of COVID-19 an other viral diseases. In [25], studies the use of common cloth masks in combatting the spread of the coronavirus. They conclude that while cloth masks have limited efficacy, they can still be used in crowded areas to prevent the spread of SARS-CoV-2. In [5], Fischer et al. show that there are some commonly used face masks that approach the standard of surgical face masks. Finally, in [3], Eikenberry et al. simulate models in order to assess the value of face mask use to mitigate the spread of COVID-19. These models suggest that wearing face masks can potentially mitigate community transmission.

3 METHODOLOGY

Here we detail our various approaches in the face mask detection task. We will first give an overview of the task and then detail the two baseline methods. For more details on the experiments, including the metrics used and results, refer to section 4.

3.1 Task Overview

Our task is defined as follows: Given an image of a scene with people, where at least one face is clearly visible, determine if that face is *a. wearing a mask, b. not wearing a mask, or c. wearing a mask incorrectly*. We can mathematically formulate our task as:

$$f(X) = bb, c \tag{1}$$

where **X** represents a tensor image input and **bb** and **c** represent the bounding box and class as vectors, respectively. If we successfully identify a face in an image, our task becomes a three-way classification task. However, for the sliding window and convolution implementation of the sliding window, we include another class *background*, since these algorithms do not have an object detection stage and instead try to classify the various pixels/frames in an image. For experimental purposes, we also try ignoring the 'background' class which we detail in our experiments in section 4.

3.2 Baselines

3.2.1 Sliding Window. Our first baseline is a sliding window approach [20] which uses a pyramid-based sliding window in order to find objects using various locations and sizes. First a sliding window slides through the original image at various horizontal and vertical regions. Next, the image is shrunk down at various stages, until the image finally fits into only sliding window. These various portions and sizes of the image are then fed into a CNN layer (with maxpool), then being fed into two FCC layers, after which a softmax determines the class of the window. Figure 2 illustrates our sliding window model.

3.2.2 Convolution Implementation of sliding window. Because the sliding window model is extremely slow, having to process $(numberofimages * numberofwindows * numberresizes)$ images, we also propose and build a convolution implementation of the sliding window algorithm, where the sliding window is represented and replaced by a CNN. Furthermore, we replace the FCCs from the sliding window model with CNN layers. That way, the final output is a convoluted matrix (or grid) in which each cell represents a portion of the image for which we give a class label. We experiment with two types of models: one which includes the background classification and one which does not. Figure 3 illustrates an example of the CNN-based sliding window model, specifically the model which outputs a 29x29 grid/matrix to represent the bounding boxes.

3.2.3 Baseline Limitations. Of course these baselines present various limitations which the state-of-the-art object detectors alleviate. First, the size of the object to be detected is a hyper-parameter on the convolution implementation of the sliding window model. Thus it cannot detect various size objects, for which the YOLOv3 and Faster R-CNN use anchor boxes for. Thus, if one uses a size that is too small, it may classify one objects as multiple, sharply decreasing the precision of the model. Second, the sliding window approach may also present many false-positives, as it does not learn the important regions of a network, but instead tries to classify all of its sliding windows. Thus, its precision will also be extremely low. We dive deeper into these details in our analysis in section 5.

4 EXPERIMENTS

All experiments were run on an NVIDIA GeForce RTX 2080 TI GPU. All experiments trained on the integrated dataset ran for 100 epochs, except for the sliding window, which ran for 40 because of its temporal inefficiency and YOLOv3 which ran for 6,000. While training on the Face Mask Detection dataset, we ran Faster R-CNN for 1,000 and YOLOv3 for 6,000 epochs. We now review the various details of our experiments, including the datasets used, setup, evaluation metric and results.

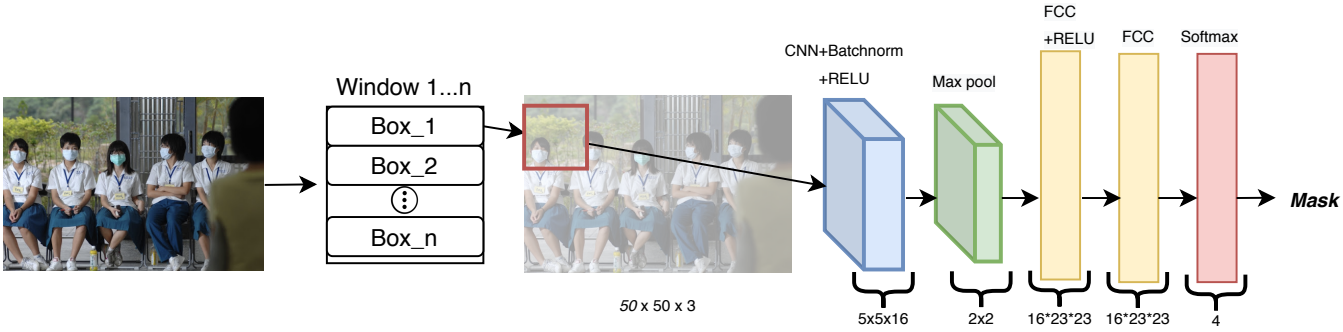


Fig. 2. An illustration of the sliding window model on a single image. Here 'box' is the square box that slides across the image, while 'window' is the resizes of the original image. We show how one box (the red square) is fed into the CNN, where the output is "Mask". Here convolution, max pool, FCC, and softmax are represented in blue, green, yellow, and red, respectively.

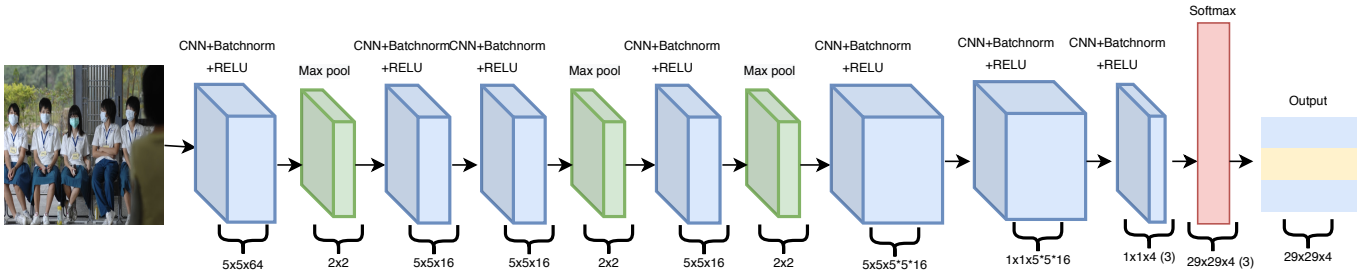


Fig. 3. An illustration of the convolution implementation of sliding windows on a single image. Specifically, we highlight the model which outputs the 29x29 convoluted matrix. Here convolution, max pool, and softmax are represented in blue, green, and red respectively. The "(4)" represents the dimensions if we are also predicting the 'background' class. Here, the yellow color in the output represents the 'mask' class, while the blue color represents the 'background'.

4.1 Datasets

In order to train and evaluate the models we found three publicly available and annoated datasets with people wearing masks. The following are the datasets that we used to evaluate the four models:

COVID-19 face mask detection dataset.[23]
This dataset consists of 1,376 images belonging to two classes: *with_mask*: 690 images and *without_mask*: 686 images. The dataset takes images of faces and applies a *.png* of a mask on top of the faces. Therefore, for the *with_mask* label, this dataset is a synthetically generated dataset. To label the locations of the faces in the images, we use a pretrained resnet10 model [7] to acquire the boundary boxes. We then create PASCAL VOC files with the boundar box and class label information.

Face Mask Detection dataset.[13]
With this dataset, it is possible to create a model to detect people wearing masks (*with_mask*), not wearing them *without_mask*, or wearing masks improperly (*wearing_mask_incorrectly*).This dataset contains 853 images belonging to the 3 classes, as well as their bounding boxes in the PASCAL VOC format. Therefore, this model required the least amount of preprocessing and was used for the initial results. Later, when combining images from the three datasets into our training set, we test on the Face Mask Detection test set.

Medical Mask. [14]
This dataset contains 4,326 images with up to 20 classes for the masks. These classes include *face_with_mask*, *face_no_mask*, and *mask_surgical* as well as other types of masks. Because our tasks is to determine whether one is wearing a mask, not wearing mask, or incorrectly wearing a mask, we only keep those data points with *face_with_mask* and *face_no_mask*. Next, we normalize the labels to *with_mask* and *without_mask* so that all of the data contains the same labels. The Medical Mask dataset has bounding boxes in the form of JSON files. The statistics in Table 1 outline the statistics of the original dataset before preprocessing.

Integrated dataset of above three dataset.
After collecting and cleaning the three aforementioned datasets above, we created an integrated dataset by combining and standardizing them. As stated before, we changed the name of all corresponding classes to *with_mask*, *without_mask*, and *wearing_mask_incorrectly*. Because some points in the boundary box were erroneous, e.g. the minimum x coordinate equal to the maximum x coordinate, we further clean the dataset of these points. Finally, we standardize all of the datasets into the PASCAL VOC format. By transforming the data into PASCAL VOC, we are able to more easily train the Faster R-CNN. Note, in our experiments we first use the Face Mask Detection

Table 1. Statistics for the various datasets explained in section 4.1. For Face Mask Detection Dataset, the bounding boxes were in JSON, while the labels were in a CSV file.

Datasets	#Images	Annotation Format	Classes	Experimented
COVID-19 face mask detection dataset	1,376	PASCAL VOC	2	-
Face Mask Detection dataset	853	PASCAL VOC	3	x
Medical Mask	4,326	CSV/JSON	20	-
Integrated dataset	5,537	PASCAL VOC	3	x

dataset since it is ready to use. Moreover, we hold its test dataset to analyze the effects of adding different kinds of face mask data from the other two datasets.

We use a 90/10 split in our experiments, training on 90% of images and testing on 10% of images. For the Face Mask Detection dataset, this corresponds to 768 training images and 85 testing images, and for the integrated dataset, this corresponds to 4,983 training images and 554 testing images.

The statics of above datasets are shown in Table 1.

Specifically, we evaluate the Faster R-CNN and YOLOv3 using the Face Mask Detection dataset[13] and the Integrated dataset. Because different models require different annotation formats, e.g., Fater R-CNN uses PASCAL VOC annotation format, while YOLOv3 uses the YOLO annotation format, we implemented scripts to translate the different forms of annotations.

4.2 Metric

The metric that we use to compare the models is the **mAP** (mean average precision). mAP is the mean of the APs(average precision) for all classes. It is the actual metric for object detection problems[22].

AP is a summarization of the shape of the precision-recall curve. It is defined as the mean precision at a set of eleven equally spaced recall levels [0, 0.1, ..., 1][4]:

$$AP = \frac{1}{11} \sum_{r \in \{0, 0.1, \ldots 1\}} p_{interp}(r) \tag{2}$$

Precision-Recall Curve. This curve describes the relation between the precision and recall values. It helps to make tradeoff decisions between the two important metrics and to set the best threshold that maximizes them. Figure 4 shows the precision-recall curve of YoLoV3 over three prediction classes: with mask, mask weared incorrect, and without mask. This was trained on the integrated dataset and tested on the test data set aside for the Face Mask Detection dataset.

4.3 Results

Table 3 shows the results of different models when training on different datasets for the YOLOv3 and Faster R-CNN models. We test on the Face Mask Detection test data. Here, for the boundary box we use a minimum overlap of 0.5. As the tables shows, YOLOv3 outperforms Faster R-CNN with both the Face Mask Detection and integrated training data.

For the two baseline models, we only use the integrated data as training data, comparing and reporting their true positive rate in Table 2 for the Face Mask Detection test set. This is because their mAP scores are all below 1%, even with the minimum boundary box overlap (less than 1% overlap). We further analyze this in section 4, but overall the low mAP scores are caused by large the number of windows and static boundary box size set as a hyper-parameter. We test four different models, three for the CNN and only one for the sliding window, because of the amount of time needed to train the sliding window. We leave further modifications on this model for future work. For the CNN, we try to also classify the eliminate classifying the background in order to improve results, giving instead a three class output tensor. As can be seen in Table 2 the CNN that does not classify the background performs better. The third CNN is a variation of the "no background" CNN, where we attempt to classify larger bounding boxes. This improves the true positive rate.

Table 2. False positive rate for the baseline models on all three class labels. Here we show three different CNN models with differing output grid sizes. The poor results can be explained by the rigidness of the bounding box. Instead of learning a bounding box, these models use predefined bounding boxes. Thus, they do not learn a good location for an object.

Baseline Models	With Mask	Without Mask	Incorrect
CNN with background (4x29x29)	26.7	11.8	0
CNN no background (3x29x29)	36.2	15.7	0
CNN no background (3x13x13)	40.5	25.5	0
FCC sliding window	14.6	11.8	0

Table 3. Detection results on the test set when training on the Face Mask Detection Dataset and the integrated Dataset. (images in the test set are removed from the training set.)

Model	training set	mAP(%)
Faster R-CNN	Face Mask Detection dataset	49.91
Faster R-CNN	Integrated dataset	66.10
YOLOv3	Face Mask Detection dataset	70.80
YOLOv3	Integrated dataset	83.90

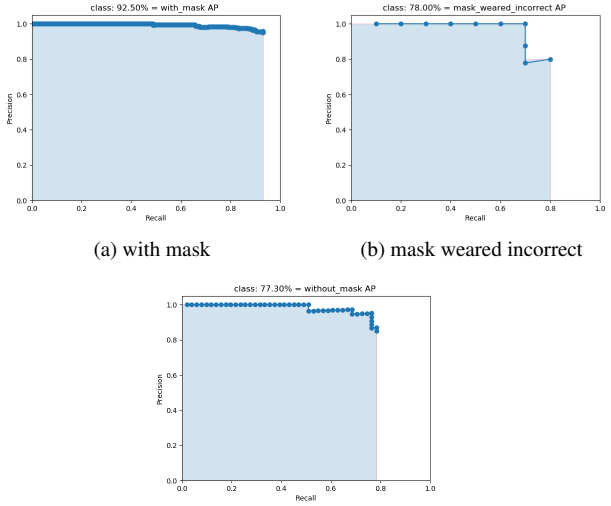
Figure 5 shows the improvement of YOLOv3 when training on the integrated dataset, compared to training only on the Face Mask Detection dataset. Although improved, both show approximately the same results when comparing one class label to another, e.g. both perform best on *with_mask* and worst on *mask_weared_incorrectly*.

Figure 6 compares the results of training Faster R-CNN and YOLOv3 over the integrated dataset and teting on the Face Mask Detection test set. Overall, YOLOv3 outperforms Faster R-CNN when training on both the smaller datase - Face Mask Detection - and the integrated dataset.

5 ANALYSIS

Baseline Models

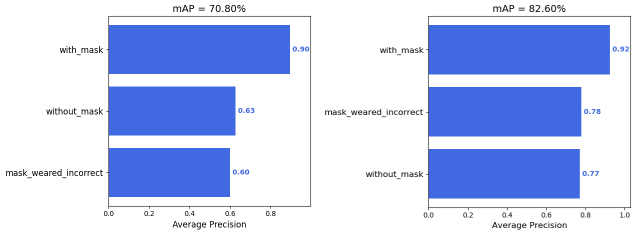
As shown in Figure 7 our baseline models do not perform well on the mask identification task. For the sliding window approach, this is expected, because for each frame (or window) the model must decide what class it belongs to. When given a frame vastly differing from the ground truth bounding boxes, it well not perform well. The siding window model is also especially slow since it needs to scan through an entire image. However, this approach may still be a cheap approach, since it is the most shallow network. For the



(a) with mask (b) mask weared incorrect

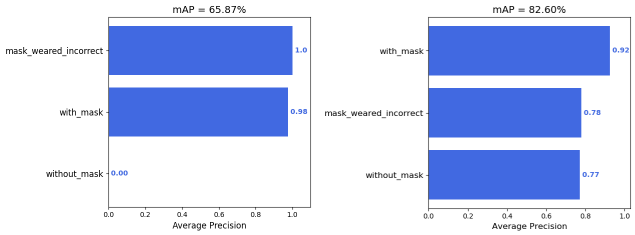
(c) without mask

Fig. 4. Precision-Recall curve of YOLOv3 that is trained over the integrated dataset and tested over the Face Mask Detection test set.



(a) Face Mask Detection dataset (b) Integrated Dataset

Fig. 5. Improvement of YOLOv3 when training on the integrated dataset.



(a) Faster R-CNN (b) YOLOv3

Fig. 6. Comparison between Faster R-CNN and YOLOv3 over the integrated dataset on the Face Mask Detection test set.

CNN-based sliding window, we need to build the CNN network such that the output gives a pre-set size bounding box. As can be seen in Figure 7, when these bounding boxes become too small, the network may detect one object as multiple objects. When the bounding box is set too large, it may combine multiple objects into one. Thus,

Fig. 7. Example images from the baseline models. Green boxes represent 'with_mask' predictions, while red boxes represent 'without_mask' predictions.

these methods do not perform well on such complex scenes as the ones from our datasets. The anchor boxes used by YOLOv3 and the Faster R-CNN help alleviate the issue of differing bounding box sizes. Lastly, since we train our baseline models by calculating the binary cross entropy loss at every cell in the window/box, our model may overfit on the background, if it constitutes most of the cells.

Effects of a larger training set.

As shown in Table 1, the Integrated dataset is around six times larger than the Face Mask Detection Dataset. When training on the Integrated dataset, YOLOv3 get 16.67% improvement in performance as shown in Figure 5.

Performance comparison and limitations of different models

Figure 6 demonstrates that YOLOv3 performs better than the Faster R-CNN in the general case. However, for the predictions over the two classes: 'with_mask' and 'mask_weared_incorrect', the Faster R-CNN has a higher precision than YOLOv3. The Faster R-CNN model does not seem to recognize class 'without_mask', as it does not make this prediction at all. This may be because the network does not generate accurate RPNs on the regions containing no masks. Future work could look into analyzing RPN-based methods for finding objects on regions which have large overlap, e.g. face vs. mask.

Types of mask the system works for

Since YOLOv3 is the best performing system in our experiments, we now use some predicted images from this system to demonstrate some of its output and capabilities.

In Figure 8 we can see various types of masks: surgical masks, cloth masks, and an n95 mask. All of these are detected by YOLOv3, demonstrating that if properly trained, the model works with various types of masks. Masks with designs or imaging on the mask can also

Fig. 8. Examples showcasing YOLOv3's capability to work with various types of masks.

Fig. 9. Examples showcasing YOLOv3's capability to detect masks in cluttered scenes and crowds.

be successfully recognized.

Masks in cluttered scenes (or crowds)
Figure 9 shows that our system works well with detecting masks in cluttered scenes and in crowds. Particularly, in the top image we can see someone wearing their mask incorrectly (not covering their nose), and in the bottom image we can see someone without a mask, for which the models predicts both correctly.

Cases where the image space features of the mask can fool the learning model
Figure 10 shows some examples where our system failed. In the left image, it may be because of the hand covering the mask. On the other cases in the images on the right, we found that these masks/faces are either at a peculiar angle or the pattern on the mask is another (cartoon) face. Future work can look into improving these corner cases which exist.

6 FUTURE WORK

Future work can look at improving the current object detection models on the face mask detection task, particularly the Faster R-CNN model with no mask. In general this can call for better models on detecting overlapping objects. In [27], Tian et al. attempt to solve this problem by making predictions per pixel. Additionally, future work on face mask detection can look at classifying different types of masks, i.e. surgical, cloth, m95, etc. While in this work we generalize all masks, it may be important to classify the different kinds of masks

Fig. 10. Some cases that fooled the system

in a hospital setting. One can begin by expanding on the Medical Mask dataset, which contains 20 classes. A further study can also

be done on our dataset in order to test on a larger nubmer of images or using a 'leave one out' approach by training on 2/3 datasets and testing on the third. Lastly, because our model can be tricked on corner cases such as sharp angles, one's hands covering their faces, and faces within a mask, it is important in future work to expand the datasets with such examples in order to achieve better performance.

7 CONCLUSION

In this work we proposed and analyzed several models including two baselines—sliding window and a CNN implementation of sliding window—and two state-of-the-art object detector models, YOLOv3 and Faster R-CNN, for the purpose of detecting people wearing face masks inside of images. We also introduced a larger and more complex (Integrated dataset) which combines three publicly available datasets for the face mask detection task. After training our models, we affirm that the YOLOv3 object detection model works best in terms of mAP. Finally, we analyze several example outputs from our system and make recommendations for future work to expand on this task in order to identify several kinds of masks as well as consider corner cases where a face and mask may overlap. Our aim is for our work to be adopted in order to mitigate the spread of COVID-19.

8 ACKNOWLEDGMENTS

The authors would like to thank Dr. Corey Toler-Franklin for their support on this project, especially their recommendations when analyzing the data output and corner cases.

A RUNNING THE CODE

File structure
- Project is contained in 'src/'
- Data is contained in 'data/data2'. This includes the input, outputs, and model weights as detailed in the sections below.
- Results are found in the Dropbox link [6], but can also be found in /data/data2/<model_name>/output/images

Running the Faster R-CNN
- create a new python virtual environment of python 3.8
- install the required libraries using command at the root directory of the project: *pip install -r requirement.txt*
- cd to the 'src' folder
- run the code using command: *python MaskDetection_FasterRCNN_main.py -mode eval*
- To train the model switch mode to 'train', however this will take about 18 hours to train

Running the YOLOv3 test
- create a new python virtual environment of python 3.8
- install the required libraries using command at the root directory of the project: *pip install -r requirement.txt*
- cd to the './src/yolov3' folder
- To test with images run the code using command: *python YoLoV3MaskDetectionTest.py*
- To test with a video clip, run the code using command: *python YoLoV3MaskDetection.py*

Running the CNN test
- create a new python virtual environment of python 3.8
- install the required libraries using command at the root directory of the project: *pip install -r requirement.txt* *Note: this only has to be done once if running all models in one session.
- cd to the './src/cnn' folder
- To test the CNN with background: *python CNN_withBG_main.py*
- To test the CNN with no background: *python CNN_withoutBG_main.py*
- To train the model switch mode to 'train'

Running the FCC sliding window test
- create a new python virtual environment of python 3.8
- install the required libraries using command at the root directory of the project: *pip install -r requirement.txt*
- cd to the './src/fcc' folder
- To test the CNN with background: *python FCC_main.py* *Note: to train the model switch mode to 'train'

Input Data
- For the purposes of the submission and to save time/space, we only share the Face Mask Detection dataset, for which the training and testing data can be found in the 'data/data2/training' and 'data/data2/testing' folders, respectively.
- For the weights/model files, they can all be found in the respectively named folder in 'data/data2/<name of mode>/checkpoint'
- For all of the input data please see: [7]

Results Data
- For a compilation of the results please see this link: [8]. We do not explicitly have a 'results' folder in our submission because of the size of the data.
- For some output results, they can all be found in the respectively named folder in 'data/data2/<name of mode>/output' with both their 'images' and 'annotations'. After running the above models with 'eval', the output will also be put in these folders.
- For our mAP graph results/output please see: [9]. *Note, we used the following mAP generator package in the link below. [10].

YOLOv3 Docker
- Because YOLOv3 requires many libraries and packages to run, we have put it on dropbox. Please see the link in the following footnote [11].

[6] https://www.dropbox.com/sh/hrb52jopky7p14a/AAA5mvGfJs2UkbotTwkQuZUea?dl=0

[7] https://www.dropbox.com/sh/ubpfd6od7rfas48/AACB7ETS2_cfKjC73yOUQgyca?dl=0

[8] https://www.dropbox.com/sh/hrb52jopky7p14a/AAA5mvGfJs2UkbotTwkQuZUea?dl=0

[9] https://www.dropbox.com/sh/tpn1l7tknvzbc4g/AACjAN3HM9yK1Be6ij71uHlLa?dl=0

[10] https://github.com/Cartucho/mAP

[11] https://www.dropbox.com/sh/ixwb2olz6tiim17/AAD2e0Xebpo7mRtGZ8XOnaZoa?dl=0

- After downloading the docker tar file, import the tar file into a docker image with:
  'docker import yolo_test.tar yolo_test:v1'.
- Then, one can build a container from docker image with a command such as:
  'docker run -it –gpus all –name yolo_test2 yolo_test bash'.
  Note please use the '–gpus' flag in order to allow Docker to access your gpus.
- Once inside the docker container, go to the darknet directory: 'cd /home/workspace/darknet'.
- Once inside the darknet directory, 'ls' to make sure the 'darknet53.conv.74' is in the directory. If not, please download it with:
  'wget https://pjreddie.com/media/files/darknet53.conv.74'.
- To train YOLOv3, run './darknet detector train data/obj.data cfg/yolov3_training.cfg darknet53.conv.74 -dont_show'. All of the images/annotations should already be in their corresponding folders.
- The weights/model will be saved in '/home/workspace/data/yolov3'.
- For more information please do not hesitate to contact us.

B BUGS/DIFFICULTIES

- Because the FCC model took so long to train (about 20 hours), we were not able to make modifications on it to make incremental improvements. Nevertheless, the model served as a baseline to show that these kinds of models do not work well for he face mask identification task.
- There were some initial difficulties with getting YOLOv3 from Darknet [12] worked. However, we were able to get the required libraries, packages, and software installed in order to get it to run. Due YOLOv3's complexity, the above steps (Running the Code) instead show how to run Faster R-CNN. However, one can attempt to set up our Docker environment in order to get YOLOv3 to run.
- Because of the large size of the dataset, we give a subset of the dataset (Face Mask Detection dataset) in order to test and run.

REFERENCES

[1] Ka Hung Chan and Kwok-Yung Yuen. Covid-19 epidemic: disentangling the re-emerging controversy about medical facemasks from an epidemiological perspective. *International Journal of Epidemiology*, 2020.

[2] Pradip Dashraath, Wong Jing Lin Jeslyn, Lim Mei Xian Karen, Lim Li Min, Li Sarah, Arijit Biswas, Mahesh Arjandas Choolani, Citra Mattar, and Su Lin Lin. Coronavirus disease 2019 (covid-19) pandemic and pregnancy. *American journal of obstetrics and gynecology*, 2020.

[3] Steffen E Eikenberry, Marina Mancuso, Enahoro Iboi, Tin Phan, Keenan Eikenberry, Yang Kuang, Eric Kostelich, and Abba B Gumel. To mask or not to mask: Modeling the potential for face mask use by the general public to curtail the covid-19 pandemic. *Infectious Disease Modelling*, 2020.

[4] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[5] Emma P Fischer, Martin C Fischer, David Grass, Isaac Henrion, Warren S Warren, and Eric Westman. Low-cost measurement of face mask efficacy for filtering expelled droplets during speech. *Science Advances*, 6(36):eabd3083, 2020.

[6] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Erik Hjelmås and Boon Kee Low. Face detection: A survey. *Computer vision and image understanding*, 83(3):236–274, 2001.

[9] Han Hu, Jiayuan Gu, Zheng Zhang, Jifeng Dai, and Yichen Wei. Relation networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3588–3597, 2018.

[10] Jane Hung and Anne Carpenter. Applying faster r-cnn for object detection on malaria images. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 56–61, 2017.

[11] Ali Imran, Iryna Posokhova, Haneya N Qureshi, Usama Masood, Sajid Riaz, Kamran Ali, Charles N John, and Muhammad Nabeel. Ai4covid-19: Ai enabled preliminary diagnosis for covid-19 from cough samples via an app. *arXiv preprint arXiv:2004.01275*, 2020.

[12] Huaizu Jiang and Erik Learned-Miller. Face detection with the faster r-cnn. In *2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017)*, pages 650–657. IEEE, 2017.

[13] kaggle. Face mask detection, 2020.

[14] kaggle. Face mask detection dataset, 2020.

[15] Tao Kong, Fuchun Sun, Huaping Liu, Yuning Jiang, Lei Li, and Jianbo Shi. Foveabox: Beyound anchor-based object detection. *IEEE Transactions on Image Processing*, 29:7389–7398, 2020.

[16] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128(2):261–318, 2020.

[17] Wei Lyu and George L Wehby. Community use of face masks and covid-19: Evidence from a natural experiment of state mandates in the us: Study examines impact on covid-19 growth rates associated with state government mandates requiring face mask use in public. *Health affairs*, 39(8):1419–1425, 2020.

[18] Iacopo Masi, Yue Wu, Tal Hassner, and Prem Natarajan. Deep face recognition: A survey. In *2018 31st SIBGRAPI conference on graphics, patterns and images (SIBGRAPI)*, pages 471–478. IEEE, 2018.

[19] VV Molchanov, BV Vishnyakov, YV Vizilter, OV Vishnyakova, and VA Knyaz. Pedestrian detection in video surveillance using fully convolutional yolo neural network. In *Automated Visual Inspection and Machine Vision II*, volume 10334, page 103340Q. International Society for Optics and Photonics, 2017.

[20] Kevin Murphy, Antonio Torralba, Daniel Eaton, and William Freeman. Object detection and localization using local and global features. In *Toward Category-Level Object Recognition*, pages 382–400. Springer, 2006.

[21] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[22] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[23] Adrian Rosebrock. Covid-19: Face mask detector with opencv, keras/tensorflow, and deep learning, 2020.

[24] Henry A Rowley, Shumeet Baluja, and Takeo Kanade. Neural network-based face detection. *IEEE Transactions on pattern analysis and machine intelligence*, 20(1):23–38, 1998.

[25] Suresh K Sharma, Mayank Mishra, and Shiv K Mudgal. Efficacy of cloth face mask in prevention of novel coronavirus infection transmission: A systematic review and meta-analysis. *Journal of education and health promotion*, 9, 2020.

[26] Christian Szegedy, Alexander Toshev, and Dumitru Erhan. Deep neural networks for object detection. *Advances in neural information processing systems*, 26:2553–2561, 2013.

[27] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 9627–9636, 2019.

[28] Daniel Shu Wei Ting, Lawrence Carin, Victor Dzau, and Tien Y Wong. Digital technology and covid-19. *Nature medicine*, 26(4):459–461, 2020.

[29] Sunny H Wong, Jeremy YC Teoh, Chi-Ho Leung, William KK Wu, Benjamin HK Yip, Martin CS Wong, and David SC Hui. Covid-19 and public interest in face mask use. *American Journal of Respiratory and Critical Care Medicine*, 202(3):453–455, 2020.

[30] Qingchao Zhang, Coy D Heldermon, and Corey Toler-Franklin. Multiscale detection of cancerous tissue in high resolution slide scans. *arXiv preprint arXiv:2010.00641*, 2020.

[12] https://github.com/AlexeyAB/darknet