APS360 Project Proposal:
Real-Time Face Mask Detector

Prepared by

Team 16

Jiachen (Jason) Zhou 1003300545
Yuxuan (Sherry) Chen 1002942587
Zhiwei (Brian) Liu 1003493007
Salar Hosseini 1003142020

Prepared for

Prof. Sinisa Colic and Teaching Team
Oct. 18 2020

Word Count: 1390

Introduction

Since the unexpected arrival of the COVID pandemic, the most prominent form of protection has been through the use of a face mask. As an attempt to combat further spread of the disease, we propose to use machine learning to implement a face mask detector which can collect data on the levels of protection that people are employing, thus enabling novel enforcement of regulations. Moreover, the goal of the project is to generate bounding boxes with labels of all face_with_mask, face_without_mask, and face_with_mask_incorrect[1] given images of multiple people. In addition to high detection accuracies, the detector should achieve real-time inference on video streams. Machine learning is an appropriate tool for this task due to the recent advent of Convolutional Neural Networks (CNNs), which have been shown to learn meaningful features from images which can be used to tackle numerous computer vision problems such as object detection.

Illustration

An illustration of our proposed face mask detection pipeline is presented in Figure 1.

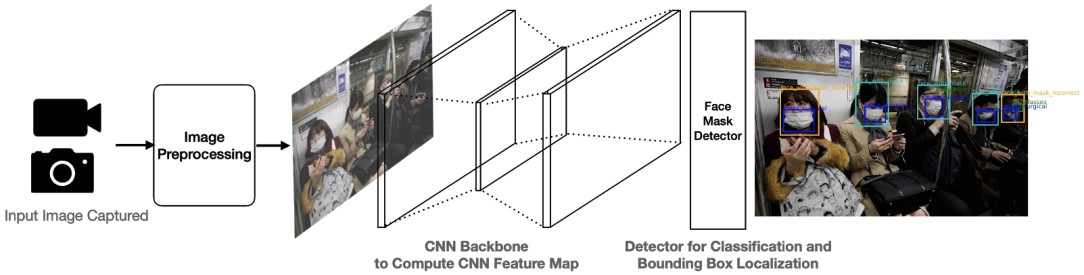


Figure 1. Illustration of the overall pipeline, which consists of image capture from a video stream, data preprocessing, a convolutional backbone, and a detector for face masks.

Background

Face mask detection is fundamentally an object detection task. A prominent object detection algorithm is Faster R-CNN, a two-stage detector which has shown strong detection accuracy at a processing speed of ~5 frames/second [1]. The architecture is a unified network composed of an initial stage called a Region Proposal Network (RPN) which produces region proposals, and a second stage called a Fast R-CNN detector [2] which simultaneously predicts bounding boxes and object labels [1], as shown in Figure 2.

[1] Mask worn incorrectly means the mask one is wearing does not cover nose, mouth, and chin as required by the health regulation.

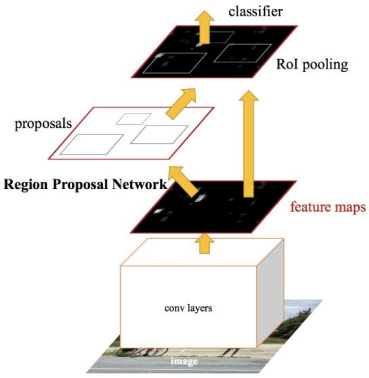


Figure 2: The Faster R-CNN object detection model [1].

Conversely, a one-stage object detector, such as YOLOv3 [3], has slightly lower detection accuracy than Faster R-CNN but a faster inference speed of ~30fps [3]. The main idea behind the original YOLO model is to divide the image into a 2D grid of cells, each pertaining to one possible object, and output for each cell a fixed number of bounding boxes and class label predictions [4], as shown in Figure 3. This is achieved by a CNN that reduces the spatial dimension to the desired grid size, and outputs features for a regression layer to predict bounding boxes and class labels [4]. Although this network can infer faster than Faster R-CNN, it imposes limitations on the vicinity of the objects due to grid size restriction.

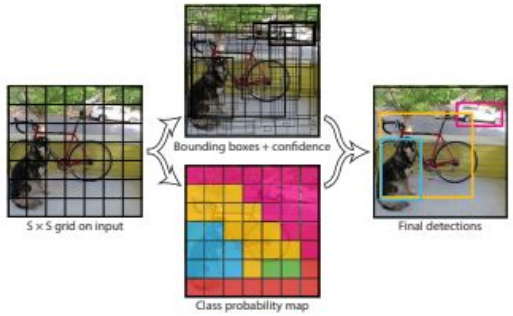


Figure 3: The YOLO object detection model [4].

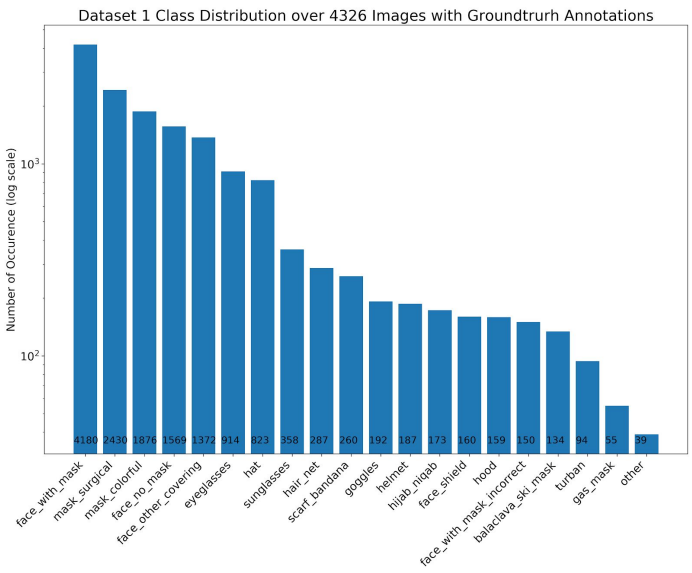
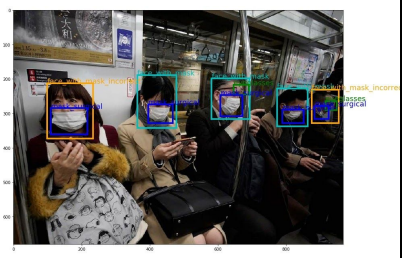
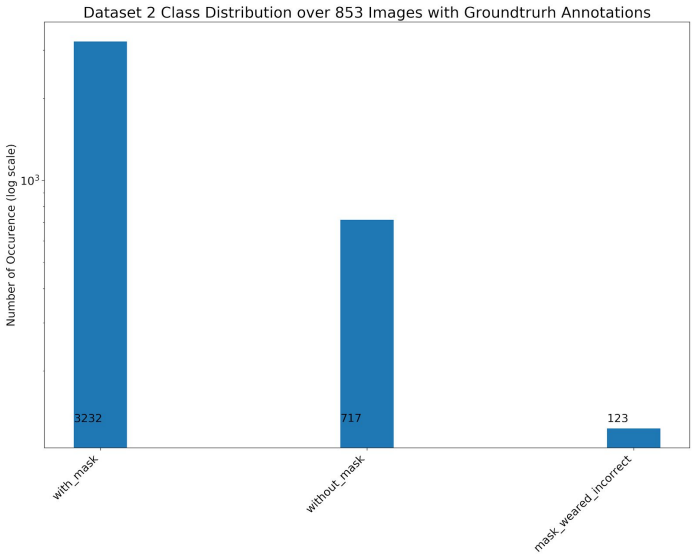
Data Processing

The two datasets that will be used are from Kaggle [5][6], as seen in the table below. The first dataset consists of 4326 annotated images of people wearing various masks, while the second dataset has 853. The bounding boxes for faces and masks are given for both the datasets, indicating whether a face is face_with_mask, face_with_mask_incorrect, or face_without_mask. For dataset 1, there is an additional label of face_other_covering, which we will group together with face_without_mask. Since we are focusing on detecting whether a mask is worn correctly or incorrectly, we will only consider the bounding boxes for faces. The mask type labels in dataset 1, such as surgical masks, scarves, ski masks, etc, will be ignored. Another useful pre-processing

step we could incorporate is data augmentation. By geometrically transforming our images, modifying the pixel values, or injecting noise, we can get a richer set of examples to train on [7].

For both the datasets, we will crop and resize the images to 224 by 224 pixels. We will use 70% of dataset 1 for training and 15% for validation. For testing, we will hold out the remaining 15% of dataset 1 and all of dataset 2, for testing transferability to new data.

Table 1. Datasets Statistics and Examples of Images and Annotations

	Class Distribution	**Data Example**
Dataset 1		 Cyan: face_with_mask Orange: face_with_mask_incorrect Red: face_without_mask Blue: mask_surgical
Dataset 2		 Cyan: with_mask Orange: mask_worn_incorrect Red: without_mask

Architecture

Our proposed face mask detector network will be based on the Faster R-CNN network [1]. We will use this architecture as a starting point over YOLOv3 because it has previously yielded higher detection accuracies and performed better for objects in a close vicinity [3].

The architecture consists of two stages, as shown in Figure 2. The first stage is a region proposal network (RPN) that generates region proposals of human faces, and the second stage takes the proposals as inputs to perform face mask detection. The RPN utilizes multiple predefined anchors with various sizes and aspect ratios, and predicts the possibility of an anchor being background or foreground, as well as the offsets from those anchors. The proposals are passed to an ROI Pooling layer that applies Max-Pooling on every region to reduce the feature maps to the same size, and then sends the channels to a classifier and regressor to detect the occurrence of objects. It refines the bounding boxes and classifies different types of masks.

We initialize the model with pretrained weights on MS-COCO [8], then finetune the network end-to-end on the face mask dataset. Given that one of the objectives is to achieve real-time inference, we will explore lighter one-stage networks such as YOLO V3 [4], RetinaNet [9], CenterNet [10] for this project.

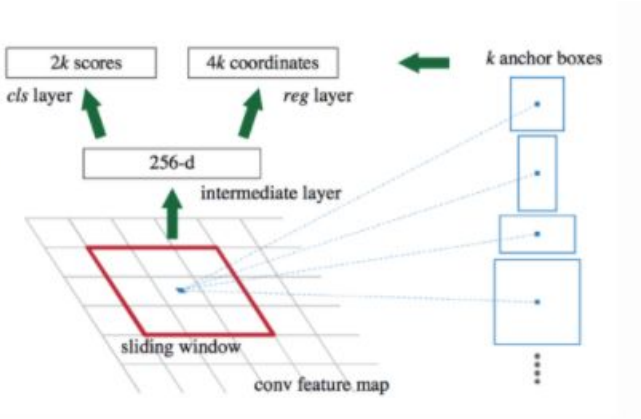


Figure 4: Detailed illustration of RPN in Faster R-CNN.

Baseline Model

We select an existing simple face mask detector network for the baseline model [11]. It consists of 2 convolutional layers, 1 max-pooling layer, and 1 FC layer. The original output layer consists of two neurons that predicts a binary score of with or without a mask. Each image is input to the classifier, which generates Region of Interests (ROIs). Then the ROIs are resized to 100x100 images before passing to a trained CNN to predict the score.

To make the baseline's performance comparable to our model, we will add an additional neuron in the output layer to predict masks that are worn incorrectly, and re-train the network on the same datasets. Then, we will evaluate the classification accuracy and inference time of our proposed model against the baseline.

Ethical Consideration

The project relies on being able to have access to surveillance, which might pose privacy issues. This is less of a concern in areas such as airports where surveillance already exists, but it is unrealistic that one can have widespread camera coverage in some general public space. Further, one may need to post-process the images of people (identified to be a risk) with techniques such as face recognition. It will also be important to make our system robust to differences in skin colour and the presence of traditional attire. To achieve this, we would need to ensure that our training data is diverse enough so that our model can learn to be insensitive to these differences.

Project Plan

We hold weekly meetings to discuss the current progress and the next focus. We will communicate via group chat throughout the week, and collaborate on a github repository. Each team member will create a new branch, so we don't overwrite each other's code. Developers will open pull requests once code changes are complete, then merge the code to master after code reviews.

A summary of our project tasks, assignees, and deadlines is shown below in Figure 5. See Appendix A for our full Gantt chart.

Project with Gantt Timeline

	At Risk	Task Name	Status	Start Date	End Date	% Complete	Notes
1		*Project Brainstorming - All*	Complete	25/09/20	06/10/20	100%	
2		Project Proposal (Survey) - All	Complete	06/10/20	09/10/20	100%	
3		Project Proposal (Formal) -All	In Progress	11/10/20	18/10/20	90%	
4		Dataset Download & Investigation (Statistics, Visualization) -Jason	Complete	15/10/20	18/10/20	100%	
5		Get up git repo, git clone Faster-RCNN, Upload datasets - Sherry	Not Started	24/10/20	25/10/20	80%	
6		Dataloader for Dataset 1 and 2- Jason	Not Started	24/10/20	26/10/20		
7		Download and Load pre-trained checkpoints/ weights for Faster R-CNN - Sherry	Not Started	24/10/20	26/10/20		
8		Investigate other architectures, YOLOv3, RetinaNet, CenterNet - All	Not Started	27/10/20	16/11/20		
9		Data augmentation - Salar	Not Started	24/10/20	28/10/20		
10		Baseline experiment - Sherry, Jason	Not Started	26/10/20	05/11/20		
11		Fine-tune on Dataset 1 - Brian	Not Started	30/10/20	06/11/20		
12		Performance Analysis - All	Not Started	06/11/20	08/11/20		
13		Progress Report - All	Not Started	08/11/20	16/11/20		
14		Hyperparameter Tuning - Jason	Not Started	09/11/20	15/11/20		
15		Prepare demo - All	Not Started	16/11/20	23/11/20		
16		Prepare presentation -All	Not Started	23/11/20	07/12/20		
17		Final Report - All	Not Started	02/12/20	10/12/20		

Figure 5. Project Gantt Chart Summary

Risk Register

Descriptions of the possible risks for our project, and their corresponding solutions are provided in Table 2.

Table 2. Project risks and their corresponding likelihoods and response strategies.

Risk	Likelihood	Response/ Contingency
A team member drops the course.	< 5%	Since all members have machine learning research experiences, the rest of the team should be able to complete this project with a modified work load distribution.
The model training takes longer than expects.	30%	The estimated training time of our model on 3000+ images takes about 10 hours on 4 GPUs, which can be acquired through Google Colab and Cloud.
Busy work schedules make it hard for teammates to work on the project.	10%	We can mitigate this risk by preemptively setting weekly meetings.
The proposed method does not yield satisfactory performance.	20%	Inference time is a key metric for our project. We plan to experiment with other light-weight and efficient models.

Link to Github/Colab

Our github repository containing all source code is publicly available at https://github.com/sherrychen127/face_mask_detector.

References

[1] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[2] R. Girshick, "Fast R-CNN," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[3] J. Redmon, A. Farhadi. YOLOv3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[5] Kaggle.com. 2020. *Face Mask Detection Dataset*. [Online] Available: https://www.kaggle.com/wobotintelligence/face-mask-detection-dataset [Accessed: 18-Oct-2020].

[6] Kaggle.com. 2020. *Face Mask Detection Dataset*. [Online] Available: https://www.kaggle.com/wobotintelligence/face-mask-detection-dataset [Accessed: 18-Oct-2020].

[7] Shorten, C., Khoshgoftaar, T.M. A survey on Image Data Augmentation for Deep Learning. *J Big Data* 6, 60 (2019). https://doi.org/10.1186/s40537-019-0197-0

[8] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Doll´ar, and C. L. Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, 2014.

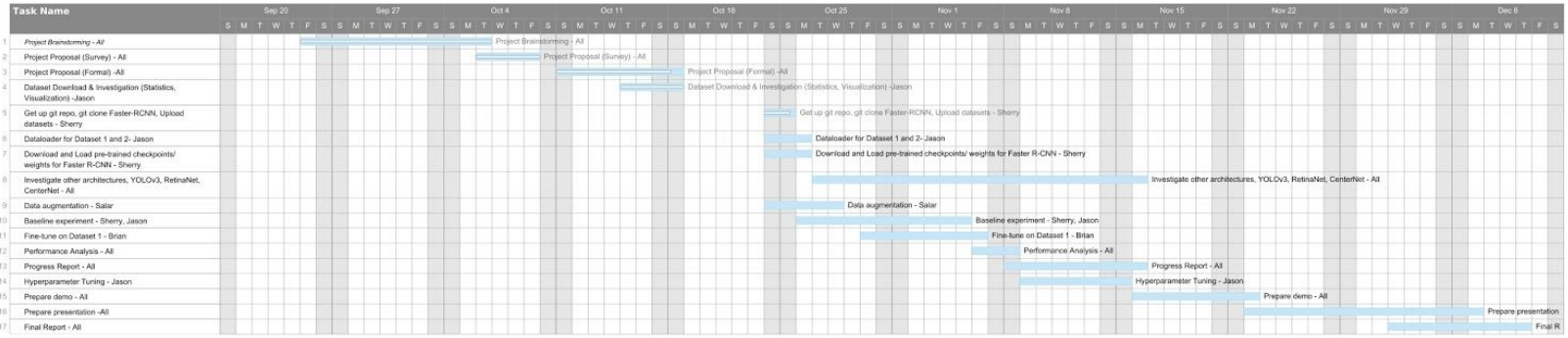
[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[10] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," 2019, arXiv:1904.07850. [Online]. Available: http://arxiv.org/abs/1904.07850

[11] A. Partners, "Face Mask Detection using CNN in Python," *Medium*, 05-Jun-2020. [Online]. Available: https://medium.com/@arbalestpartners/face-mask-detection-using-cnn-in-python-3148f82dcfe7. [Accessed: 18-Oct-2020].

Project with Gantt Timeline

smartsheet



Appendix A Gantt Chart Visualization