Live-Stream Video Face Mask Detection

Yushan Wang[1,2] and Jiemin Tang[2]
[1]Department of Computer Science and Engineering,[2] Department of Mathematics
University of California, San Diego

ABSTRACT

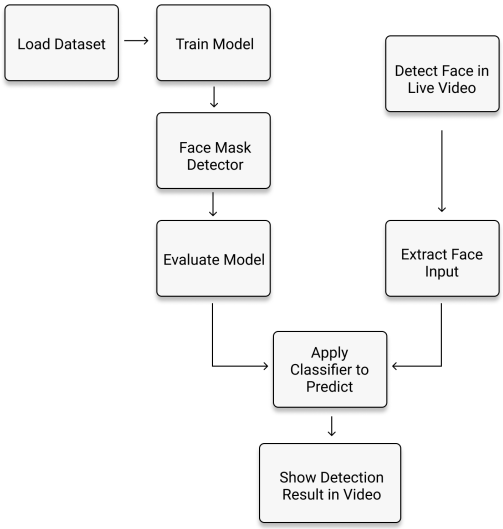
Due to the outbreak of COVID-19 in early 2020, the governments around the world have required their citizens to wear face masks when social contact with other people are unavoidable during outdoor activities. This serves both as a protection to the health of people and as a prevention of further spread of the disease. As we have learned from COGS 181, Convolution Neural Networks has achieved great success in image recognition and processing, and in this paper, we used CNN as a binary classifier to classify whether people wear face masks or not in photos and videos. There are two different data-sets and two different CNN designs used. We use the best training model as the prediction model to develop a program that allows the camera to capture the faces of individuals and identify whether they are wearing face masks or not.

I. BACKGROUND

In order to effectively prevent the spread of COVID-19 virus, almost every market and store requires their customers to wear a mask before entering during the coronavirus pandemic. However, the staffs are manually scrutinizing whether customers are wearing face covering or not. This process has low security and safety to both the customers and the working staff. At present, most stores and markets have installed security camera, and the video is live-streamed. If we could develop a program by deploying artificial intelligence to check whether customers are wearing face coverings are not through the video, then the efficiency and safety could be greatly improved.

II. INTRODUCTION

To realize the live-stream video face mask detection, we first need a face mask detector. A detector could also be understand as a binary classifier, either with or without a mask. Then we need to design a classifier and find the corresponding data sets for training. Once we have a trained the face mask detector, then we need to implement our classifier model into live-stream video. For live-stream video we need to detect a face first, in other words, face detection. This would be elaborated under the **FACE DETECTION** section. Once we find face in the live-stream video, we could take the face as the input to our classifier and the prediction would be our results to check whether the faces are wearing masks or not. The results would be rendered to the live-stream video, for example, if you are wearing a face mask, then a green rectangle would outline your face area, and says "with mask".Otherwise, a red rectangle would appear and says "without mask". The Complete work flow diagram is in below.



III. KEY CONCEPTS

A. Convolution Neural Networks and VGG 16

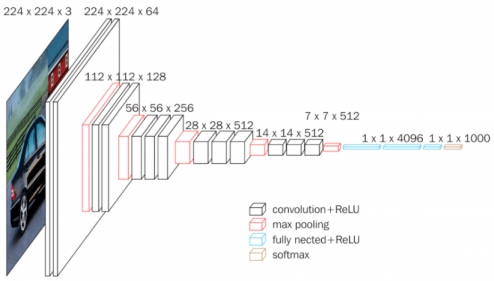
In the Convolution Neural Networks that we constructed to detect whether individuals are wearing face masks, there are two hidden layers in the structure, in which we used ReLu layer and Max Pooling layer. The choice of activation function we selected in the end is the softmax function.

The Rectified Linear Unit serves the purpose of breaking up the linearity during image processing. The convolution operations might impose linearity on images that are essentially non-linear, which would undermine the learning ability of the model. ReLu removes all the black elements in the image, leaving only the white and grey colors, so that we can observe the change in the patterns more abruptly after this operation. On the other hand, the max pooling layer is used to teach the convolution neural networks regardless of the background and texture differences that may affect the accuracy of the model.

This problem of detecting face masks is a transfer problem from the cat vs. dog classification, and it is also similar to the classification problem that we have written for the homework. Similar construction of the layers are being

specified when trying to train the models, and we have achieved considerable accuracy using this strategy.

We also used VGG-16, proposed by K. Simonyan and A. Zisserman from the University of Oxford, to cross-compare the result of our CNN classifier. VGG-16 replaces large size kernel filters with a stack of 3 x 3 filters, while each uses stride size of 1 in convolution layer. Furthermore, the spatial padding is 1 with a stride size of 1 for the 3x3 convolution layers, while the padding in 2x2 max pooling layers has a stride size of 2. Each hidden layer in the VGG-16 structure include a ReLu layer to deal with non-linearity. VGG-16 was mainly used for image classifications and detection, so we deployed this model to perform a transfer learning.



B. Local Binary Pattern and Haar Cascade

Local Binary Patterns (LBP) were originally developed as a means to describe texture images. Since then, they have been successfully applied to a wide range of other image recognition tasks, most notably the face recognition. LBP codes capture local image micro-textures. They are produced by applying thresholds on the intensity values of the pixels in small neighborhoods using the intensity of each neighborhood's central pixel as the threshold. The resulting pattern of 0s (lower than the threshold) and 1s (higher than the threshold) is then treated as the pixel's representation or code. When the neighborhood contains eight other pixels, this binary string is treated as an eight-bit number between 0 and 255. These codes are typically pooled over image regions using a histogram of code frequencies.

Haar Cascade is one of the most commonplace model for face detection. It performs by selecting Haar features from the photos and getting the rectangle dimension of the face outline. The face detection performs quite well with the testers' faces when they are not wearing masks. However, the pre-trained Haar Cascade and Local Binary Pattern Cascade models could not cope with the faces with masks on. The reason is that, the mask would block at least half of the facial features, which invalidates the predictions of the pre-trained model. Even those two models could not fulfill the requirements of our model, their purposes will be discussed later from another perspective.

To correctly perform face detection even with face mask on, we used the model published by Gopinath Balu's caffe-DNN. The accuracy of the face detection was greater then the former two choices.

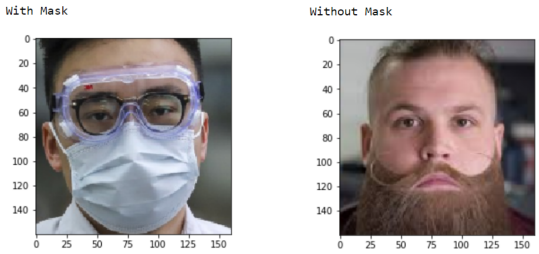


Fig. 1. Example of real mask dataset

IV. Methods

A. Proposed Dataset

The two sets of data we used to train the model are found from internet sources. The first one consists of human faces with and without real masks on, while the second one consists of human faces with and without photo-shopped masks on. For both sets of the data, people with and without face masks do not match up. That is, the images of people who are not wearing masks are completely different with those who are wearing the masks. If we simply use the same people either with or without mask, this would make the data-set invalid and useless, because we cannot assume the same characteristics and background information for each human face identified from the images. In addition, when training both data-sets, people with and without face masks are separated in two folders as a reason to easily classify the y-values to train the models. Details of each data-set are listed below.

TABLE I
DESCRIPTIONS OF REAL MASKS DATASET

Category	Data Size	Comments
No Mask	1281	label = 0,1
Mask	1486	label = 1,0
Total	2767	

TABLE II
DESCRIPTIONS OF GENERATED MASKS DATASET

Category	Data Size	Comments
No Mask	690	label = 0,1
Mask	686	label = 1,0
Total	1376	

B. Data Preprocessing

Each data-set contains multiple images of faces. Each image is read as input of data and each one has three channel B(blue),G(green),R(red). Each image is resized to (160,160). Therefore each single input has the shape of (160,160,3).

We also used gray-scale image for training input to train the classifier to cross compare which data set would produce the higher accuracy.

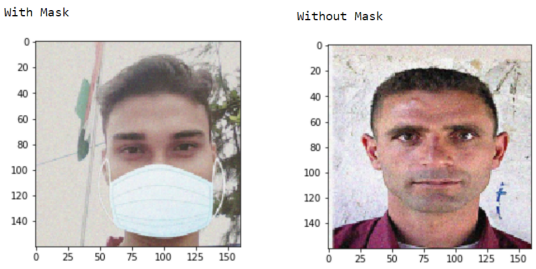


Fig. 2. Example of generated-mask data-set

C. Performance Metrics

In order to prevent over-fitting of our model, we used cross-validation to test the performance of both our training and testing sets. After the 80/20 train/test split, we obtained separate statistics for each set, with the loss and accuracy used to measure the performance on the training set, while val loss and val accuracy are to measure the performance on the testing(validation) set. From the statistics obtained, there is no sign of over-fitting of our model, and the accuracy appears to be relatively consistent in both the training and testing sets.

V. Experiments

A. Design of the CNN

As we can see from the figure below, the Convolution Neural Networks that we designed for each data-set has two hidden layers, and both include the ReLu and Max Pooling layer. As mentioned above, the ReLu layer is used to break up linearity during image processing, while the Max Pooling layer is used to down-sample the feature maps, which adds to the model's invariance to local translation.

B. Training on the classifier

We performed several training on the two different data-sets by using the same CNN design. First, we use the original data-set with basic image preprocessing, and then we use CNN to train the model using the data-set of mask/no_mask classifier.Below is the training accuracy and training loss.

As shown from the figures above, the classifier performs better on original pictures compared to gray-scale images for both CNN and self-generated pictures. For the CNN gray-scale images, the training loss curve starts to converge at around epoch 10, and the training accuracy curve starts to converge at around epoch 8,validation accuracy curve at epoch 5, while there is no clear convergence in the validation loss curve. This is likely due to the over-fitting of the classifier on gray-scale images. On the other hand, the training and validation loss curves as well as the training and validation accuracy curves of the original CNN images show some convergence at around epoch 5. Therefore, the performance of classifier on original images are relatively better than gray-scale images.

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 158, 158, 32)	896
max_pooling2d_1 (MaxPooling2	(None, 79, 79, 32)	0
conv2d_2 (Conv2D)	(None, 77, 77, 32)	9248
max_pooling2d_2 (MaxPooling2	(None, 38, 38, 32)	0
conv2d_3 (Conv2D)	(None, 36, 36, 64)	18496
max_pooling2d_3 (MaxPooling2	(None, 18, 18, 64)	0
dropout_1 (Dropout)	(None, 18, 18, 64)	0
flatten_1 (Flatten)	(None, 20736)	0
dense_1 (Dense)	(None, 64)	1327168
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 2)	66

Total params: 1,357,954
Trainable params: 1,357,954
Non-trainable params: 0

Fig. 3. Layers of the designed Convolution Neural Networks

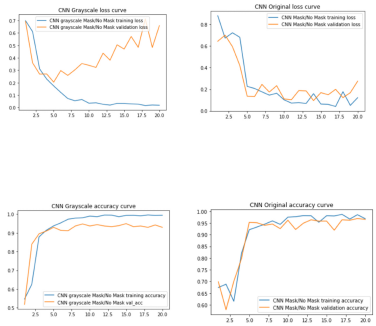


Fig. 4. Loss and accuracy curves of CNN images

As for the self-generated images, the performance of classifier is also fairly poor on gray-scale images. Although the training/validation loss and accuracy curves of the original images show a sign of convergence at around epoch 10, the overall performance is not as desirable as the CNN images. Therefore, it seems that the classifier we trained is more likely to classify the images in which people are actually wearing face masks compared to those pictures in which face masks are photo-shopped.

C. Testing

We used the same metrics for the testing. Here below is the table for each dataset to CNN

TABLE III
TEST ACCURACY AND TEST LOSS

Category	Test Accuracy	Test loss
Mask/No$_{M}askCNN$	0.965	0.275
Mask/No$_{M}askCNN$ GrayScale	0.929	0.658
Auto-generated Mask CNN	0.971	0.110
Auto-generated Mask CNN GrayScale	0.481	0.693

The gray-scale and original images for mask_no mask data-set has quite high accuracy during the testing. Also, the original images for the auto-generated masks dataset

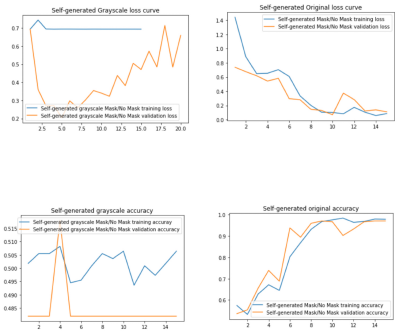
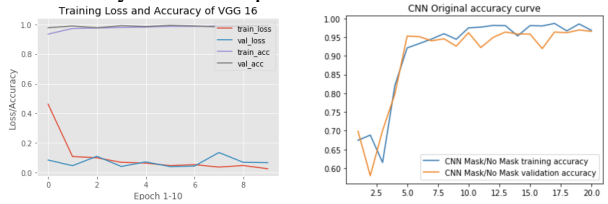


Fig. 5. Loss and accuracy curves of self-generated images

also have quite high accuracy. However, the accuracy for the gray-scale auto-generated mask dataset does not reach an desirable accuracy. Below is an example of our testing.

D. Performance of face mask classifier

The performance of the model for static image is working quite well. Below is our cross comparison of our model with VGG-16. The two models perform with high accuracy to the static images. However, when our own CNN design used in the real-live video-stream was not works so well, the vibrations are much. The face without mask could be predicted correctly but when mask is wearing, the model was not that accurate. Compared to VGG, VGG model since the overall training converge much faster, accuracy is slightly higher and more stable, the live-video=stream also works well. It was quite stable and accurate during the whole performance testing phase. Both of the models were not perform highly well, there are more than one individual in the video. We are not completely certain whether is the face detection phase cause this accuracy downward or the model itself. The time for prediction for both model are just a few milliseconds, basically could be considered as simultaneously and thanks to the functionality of openCV, the time delay was not a problem worth to consider.



VI. CONCLUSIONS

In conclusion, our CNN design works well for the mask/no mask image classification. Face detection works well for detecting human faces in live-stream video. However, the CNN classifier we have does not work well in live-stream video prediction, partial predictions are correct, but the variations and errors are higher than VGG model. Furthermore, the VGG model fits well when there is only one single individual in the frame. When there are more than one tester in the frame, the accuracy slightly downward. Especially when the two faces have partial overlap. The accuracy of predicting all tester in the frame wearing masks or not is challenging.

VII. APPLICATION STATUS

The current model is tested successfully through webcam phases. The live-stream requirement has been satisfied. However, we were not certainly about the result for this model to apply to monitor camera. The CCTV footage might have lower resolution to individual face images. Also, there are certainly will be multiple individual human beings would appear in the image. The rectangles which contain each face might have overlapping. The result might yield to higher error rate. Those still need to be tested. Even though, we have already tested when there are more than one face in the video, the model could successfully predict the mask and no mask situation but the accuracy was slightly decreased compared to only one tester in the frame.

VIII. FUTURE IMPROVEMENTS

Due to the time and resource limit, this is the best model we so far can come up with. No question that VGG is a highly robust model. We could increase our accuracy by design filter for multiple faces in the frame. Also, we could decrease the resolution of our datasets. Since in real -live monitoring footage might have that high resolution when extracting an individual face image. Also, the visualization could also be improved. We only used the basic face detection techniques to round up the face area in the video and print the "mask" and "no_mask" statement near the face.

IX. BONUS

There are several points in this paper would deserve the extra credits. The Local Binary Pattern and Haar Cascade model for face detection. Furthermore, the live-video face detection and our classifier to the face detection capture images application are surely beyond the requirement of this project.

X. ACKNOWLEDGMENT

We thank Prajna Bhandary for the auto-generated mask dataset, Gopinath Balu for the caffe-dnn face detection model and Weihao Zeng for comments that greatly improved the manuscript.

REFERENCES

[1] Jeng-Hau Lin, Justin Lazarow, Yunfan Yang, Dezhi Hong, Rajesh K. Gupta, Zhuowen Tu, "Local Binary Pattern Networks "

[2] Juefei-Xu, Vishnu Naresh Boddeti, Marios Savvides,"Local Binary Convolutional Neural Networks"

[3] Tasnuva Hassan, Haider Adnan Khan,"Handwritten Bangla Numeral Recognition using Local Binary Pattern"

[4] Gil Levi, Tal Hassner, "Emotion Recognition in the Wild via Convolutional Neural Networks and Mapped Binary Patterns"

[5] Timo Ojala, Matti Pietikäinen and Topi Mäenpää, "Multiresolution Gray Scale and Rotation Invariant Texture Classification with Local Binary Patterns"

[6] Rao Muhammad Anwer, Fahad Shahbaz Khan, Joost van de Weijer, Matthieu Molinier,Jorma Laaksonen,"Binary Patterns Encoded Convolutional Neural Networks for Texture Recognition and Remote Sensing Scene Classification"

[7] Zhongyuan Wang, Guangcheng Wang, Baojin Huang, Zhangyang Xiong, Qi Hong, Hao Wu, Peng Yi, Kui Jiang, Nanxi Wang, Yingjiao Pei, Heling Chen, Yu Miao, Zhibing Huang, and Jinbi Liang, "Masked Face Recognition Dataset and Application"

[8] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval"

[9] Adrian Rosebrock,"COVID-19: Face Mask Detector with OpenCV, Keras/TensorFlow, and Deep Learning"

[10] T. Ahonen, A. Hadid and M. Pietikainen, "Face Description with Local Binary Patterns: Application to Face Recognition,"

[11] Yang, H. and Wang, X. A. (2016) 'Cascade classifier for face detection', Journal of Algorithms Computational Technology, pp. 187–197. doi: 10.1177/1748301816649073.

[12] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen, "Face Recognition with Local Binary Patterns"

[13] Phillip Ian Wilson and John Fernandez. 2006. Facial feature detection using Haar classifiers. J. Comput. Sci. Coll. 21, 4 (April 2006), 127–133.

[14] Tammina, Srikanth. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. International Journal of Scientific and Research Publications (IJSRP). 9. p9420. 10.29322/IJSRP.9.10.2019.p9420.

[15] Karen Simonyan and Andrew Zisserman,"Very Deep Convolutional Networks for Large-Scale Image Recognition"

[16] Adrian Rosebrock, "Local Binary Patterns with Python OpenCV", December 7, 2015