How Different Types of Deep Learning Models Performs on Face Mask Detection

Anonymous CVPR 2021 submission

Paper ID ****

Abstract

With the spreading of COVID-19, face mask has become a regular item for our daily life. With the reopening of restaurants and business, face mask is essential to stop the spread of COVID-19 and protect people's health. Everyone should wear face mask in public places in order to protect both ourselves and others under this special situation. However, how to detect whether people are wearing face mask or not? In this paper, we are going to detect face mask with the help of deep learning method. We first gathered the pictures with different people with and without mask. Then, we explored different different models including VGG, ResNet, GoogleNet and DenseNet. With the convolutional neural network built by ourselves, we reached the accuracy of 0.84. We show that with our final model with VGG16 reached the accuracy of 0.98. In this regard we can generate a reliable face mask detector which can be used in public places to help remind people wearing mask.

1. Introduction

1.1. Background

With the continuous spreading of COVID 19, our life is being heavily influenced by it. Nowadays, restaurants, gyms and state parks are reopened and life is reaching to a new balance. People are getting used to a new mode of lifestyle and trying to live together with corona virus. As states reopen from the stay-at-home instruction, many states now requires people to wear face mask in public spaces to reduce the spread of COVID-19. Both Centers for Disease Control and Prevention and the World Health Organization recommend masks for general public. The more people wearing mask, the better we can fight with COVID-19. Therefore, face mask is becoming an essential item in our life in order to protect ourselves and the public. However, when there is a large crowd, how do we detect whether people are wearing mask properly? We can rely on staff in the public places to remind people wear mask. But in the large public area like train station, airport and schools, it is impossible to completely depend on staff to detect face mask wearing. Therefore, this question comes to us: is it possible to do face mask detection using deep learning method? With camera catching people's faces in public area, the model can tell us whether he or she is wearing mask or not.

1.2. Motivation

The model can help states implement their regulation on mask wearing in public places. In this way, we can protect the health of each other, which can further help control the spread of COVID-19 and make life get back to normal as soon as possible. What's more, controlling mask wearing by technology instead of by people supervising each other can help build up trust and make our community more united. Let's fight COVID-19 together.

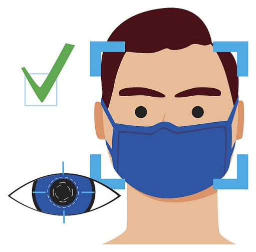

Figure 1. Face Mask Detection

2. Related Work

Since COVID-19 is changing everyone's life, topic related to COVID-19 is the most heated one nowadays. So does the face mask detection. It's a new topic with a lot of concerns. Therefore, There are many people working on it now.

2.1. Algorithm explored on face mask detection

For face recognition, Li et al[1] used the YOLOv3 algorithm. As the backbone, YOLOv3 utilizes Darknet-53. 93.9 percent precision was obtained by the proposed process. It has been conditioned on more than 600,000 photographs in CelebA and Larger Facial datasets. The research was conducted using the FDDB dataset.
A novel GAN-based network that can automatically erase masks covering the face region and recreate the image by constructing the missing hole was proposed by Nizam et al[2].
An immersive process called MRGAN was proposed by Muhammad et al [5]. The approach relies on getting the microphone area from the user and restoring this area using the Generative Adversarial Network.
Shaik et al[4] used real-time face emotion detection and recognition for deep learning. To classify seven facial expressions, they used VGG-16. On the KDEF dataset, the suggested model was trained and achieved 88 percent precision.

2.2. Companies working on face mask detection

2.2.1 NVIDIA Clara Guardian

NVIDIA developed NVIDIA Clara Guardian, an application platform and partner ecosystem that simplifies the creation and deployment in healthcare facilities of smart sensors with multimodal AI. A list of healthcare-specific, pre-trained models and reference applications driven by GPU-accelerated application frameworks, toolkits, and SDKs is provided by Clara Guardian. NVIDIA Transfer Learning Toolkit (TLT) can help build highly precise, smart video analytics. The developer recipe demonstrates the high-level workflow of pre-trained model downloading and datasets downloading and translating to the KITTI format for use with TLT. To detect masked and no-mask faces, the quantized TLT model is then deployed using the DeepStream SDK.[6]

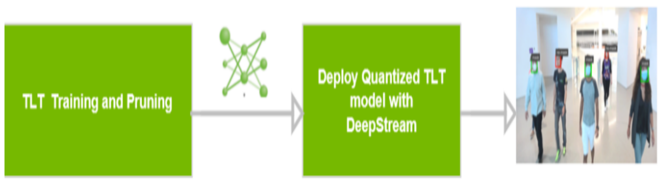

Figure 2. Workflow for developing an AI-based face mask detector.

2.2.2 Xovis 3D

People wearing face masks can be detected by Xovis 3D sensors. By simply installing the Face Mask Detection plugin, people can gain from all the existing functions and add value. Sensors count all individuals and recognize whether or not a person is wearing a face mask AI firmware has been trained to identify individuals visually if they are wearing a face mask. Data provides information for an instant warning feature or further processing of analytics, as necessary.[11]

2.2.3 Neuromation

In real-time video streams and pictures, Neuromation aims to capture facial expressions and recognize if someone is wearing a protective mask with a 99.98 percent accuracy rate. The face mask detection system is equally powerful for both person and group detection and can complement or decrease the number of compliance agents on the field. The system classifies every person as ”wearing a mask” or flags as not wearing a mask” after performing the original analysis and sends an instant warning.[7]

2.3. Our goal

We want to provide our own platform for face mask detection to help the reopen for Georgia Tech. Right now, there is no open source code for it, that’s why we are going to create own model on face mask detection. Hopefully next semester when there are more hybird and in person class in Georgia Tech, the model can guarantee the mask wearing and protect Yellow Jackets!

Figure 3. Yellow Jacket helps Yellow Jackets

3. Dataset Analysis and Preparation

In this section, we are going to first introduce our dataset and how we pre select and clean it. Then we will go through the standard we evaluate our model. In the last part of this section, we explain each model and the result of it in details and reach a conclusion.

3.1. Dataset description

Our research conducted experiments on mainly two sources. First is Real-world Masked Face Dataset(RMFD) [10]. 5000 masked faces and 90,000 unmasked faces are consisted in the RMFD dataset.
The second source is from Kaggle. We downloaded the picture of people wear face mask to increase the percentage of masked faces.

Figure 4. Face Mask Dataset Image Samples

In order to make sure the run time of each simulation can be controlled in an acceptable range and the balance of the images with and without face mask, we finally decide to set our dataset with 3846 images in total, whereas 1923 images are with mask and 1923 images are without face mask.

3.2. Data preprocessing

3.2.1 Data labelling

We store our pictures under two different files: with mask and without mask. In this way, the label is contained in the path of every image. We can label each picture by mapping it with the name of the file when we read it.

3.2.2 Augmentation and normalisation

We do augmentation and normalisation for the images before training. First, we randomly extract a patch of size (224, 224) from the input image and then flip it horizontally. After the data augmentation process, we scale the data with the mean value as [0.485, 0.456, 0.406] and std value as [0.229, 0.224, 0.225]. The final input image is shown as Fig 5.

3.3. Workflow

Our main workflow is shown in the Fig 6. below. After data preprocessing, we focused on choosing and training

Figure 5. Input Image Sample

deep learning model to reach an ideal accuracy and avoid overfitting in order to make it useful in real life. We have

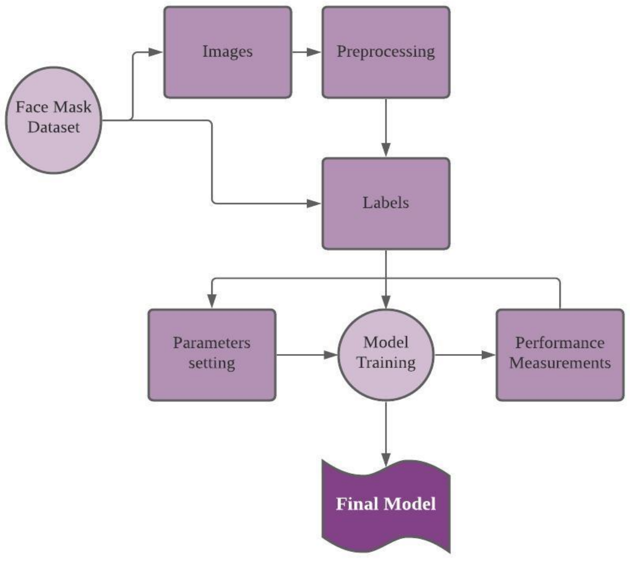

Figure 6. Workflow

several measurements to decide how our model is performing based on the data we have, including accuracy, loss function and cross validation.

3.3.1 Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

whereas,
TP: observation is positive, and is predicted to be positive.
FN: observation is positive, but is predicted negative.
TN: observation is negative, and is predicted to be negative.
FP: observation is negative, but is predicted positive.

3.3.2 Loss function

Since we are having a classification model, we use cross-entropy as a loss function. It can lead to a faster training as well as an improved generalization. In a binary classification, where the number of classes is 2, cross-entropy can be calculated as:

$$Loss = -(y \log(p) + (1 - y)log(1 - p))$$

3.3.3 Overfitting

A model which is good with noise but not signal is considered as overfitting. Overfitting occurs when the accuracy on the test set is lower than the accuracy on the train set, which means that it performs badly with a new dataset. Usually, we avoid overfitting by training with more data, removing features, cross validation and early stopping.

4. Experiments and Evaluations

4.1. Experiments

4.1.1 Baseline Model

To begin with, we defined a Convolutional Neural Network model as the baseline. The baseline CNN model consists of 2 convolutional layers and 3 linear layers. The batch size was set to be 4.
We use the pytorch SGD optimizer and set the learning rate to be 1e-4, momentum to be 0.9. The data set is divided into train set and test set, and the size of test set is 20% the size of data set. Also, 25% of the train set is divided to be our validation set.
After running for 10 epoches, we tested the model on both train set and test set, and the accuracy is 85% and 89% respectively. The train and versus validation loss plot of the baseline model is shown in Figure 7.

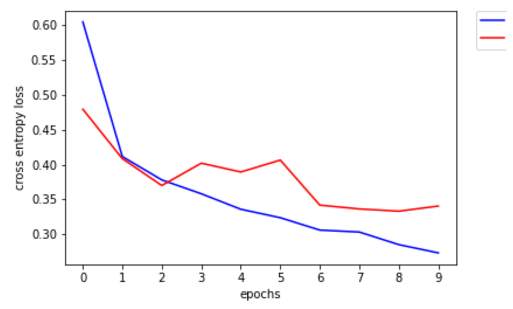

Figure 7. Baseline model train vs. validation loss plot.

4.1.2 Residual neural network(ResNet)

ResNet is an artificial neural network of a kind that builds on constructs known from pyramidal cells in the cerebral cortex. Residual neural networks do this by utilizing skip connections, or shortcuts to jump over some layers. Typical ResNet models are implemented with double- or triple-layer skips that contain nonlinearities (ReLU) and batch normalization in between.[9]
In this project, different variants of ResNet is applied, including ResNet18, ResNet34, ResNet50, ResNet101, ResNet152. Different variants indicate different depths, number of layer. The results are shown in the Table 1.

Model	Test Accuracy(%)	Train Accuracy(%)
ResNet18	92	94
ResNet34	93	95
ResNet50	94	96
ResNet101	94	95
ResNet152	91	94

Table 1. Results of Resnet networks.

As shown in the table, more layers does not guarantee better performance. In addition, as the model contains more layers, it takes longer to train. ResNet50 gives the best result in the experiments, the train and versus validation loss plot of it is shown in Figure 8.

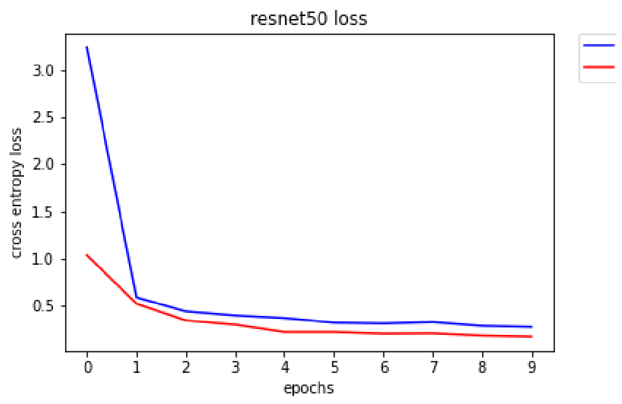

Figure 8. Resnet50 train vs. validation loss plot.

4.1.3 Densely connected convolutional networks (DenseNet)

DenseNet is a network architecture where each layer is directly connected to every other layer in a feed-forward fashion (within each dense block). For each layer, the feature maps of all preceding layers are treated as separate inputs whereas its own feature maps are passed on as inputs to all subsequent layers[3]. We experimented multiple Densenet networks including DenseNet121, DenseNet169, DenseNet201, and DenseNet161, using SGD optimizer with learning rate of 0.0001 and 10 epochs. Different networks indicate different depths, number of layers, or sizes

Model	Test Accuracy(%)	Train Accuracy(%)
DenseNet121	96	99
DenseNet169	97	99
DenseNet201	97	99
DenseNet161	98	99

Table 2. Results of Densenet networks.

of pretrained models. The results are shown in Table 2. From the results, DenseNet161 works better regarding to its accuracies and the model size. The train and versus validation loss plot of DenseNet161 is shown in Figure 9.

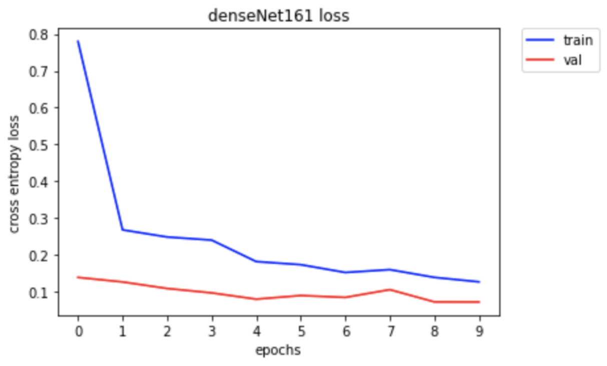

Figure 9. Densenet161 train vs. validation loss plot.

4.1.4 GoogLeNet

GoogLeNet is a 22 layers deep network based on a deep convolutional neural network architecture code named "Inception", which was responsible for setting the new state of the art for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge 2014 (ILSVRC 2014). After running for 10 epoches, we tested the model on both train set and test set, and the accuracy is 91% and 91% respectively. The train and versus validation loss plot of GoogLeNet is shown in Figure 10.

4.1.5 VGG

VGG networks are trained with large-scale image recognition setting, it increased depth using an architecture with very small (3x3) convolution filters and indicates a significant improvement on the prior-art configurations can be achieved by pushing the depth to 16-19 weight layers[8]. VGG networks are applied with batch normalization to improve computing speed and reduce overfitting. In this experiment, we trained various VGG networks with and without batch normalization, the results are shown in Table 3.

From Table 3, VGG16 outperforms other VGG networks and batch normalization leads negative impact on the training accuracies. The train and versus validation loss plot of VGG16 is shown in Figure 9.

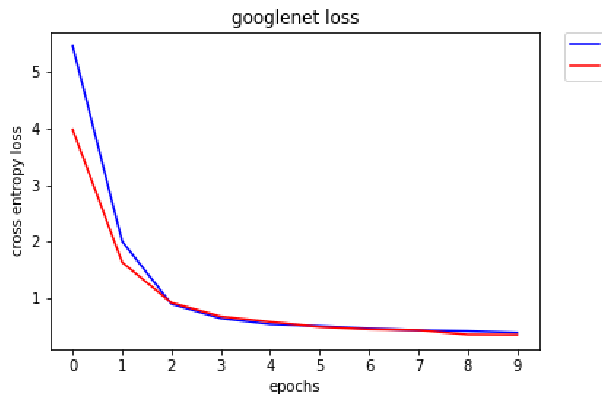

Figure 10. GoogLeNet train vs. validation loss plot.

Model	Test Accuracy(%)	Train Accuracy(%)
VGG11	97	99
VGG11_bn	96	99
VGG16	98	100
VGG16_bn	96	99
VGG19	97	99
VGG19_bn	96	99

Table 3. Results of VGG networks.

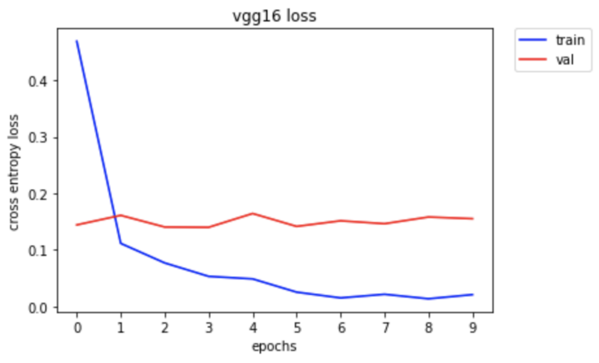

Figure 11. VGG16 train vs. validation loss plot.

4.2. Evaluations

Model	Test Accuracy(%)	Train Accuracy(%)
Baseline	85%	89%
ResNet50	94%	96%
DenseNet50	98%	99%
VGG16	98%	100%

Table 4. Results of different models

Based on the experiment results shown above, we compared different models and chose the best variant among each group. The results are shown in Table 4. In our experiment, VGG16 gives the best performance.

5. Conclusions

In this work, we developed a face mask detection model. With the convolutional nerual network developed by ourselves reaching the accuracy of 0.84, we further increased the accuracy to 0.98 by VGG16. The model can perform well without overfitting. Therefore, it is reliable to be implemented in real world performance. Hopefully, the model can help guarantee people wearing mask in public places. Let's defeat COVID together.

References

[1] J. Li L. Fei C. Li, R. Wang. Face detection based on yolov3. *Communication and Devices*, 277–284, 2020. 2

[2] Nizam Ud Din, Kamran Javed, Seho Bae, and Juneho Yi. A novel gan-based network for unmasking of masked face. *IEEE Access*, 8:44276–44287, 2020. 2

[3] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017. 4

[4] Shaik Asif Hussain and Ahlam Salim Abdallah Al Balushi. A real time face emotion classification and recognition using deep learning model. *Journal of Physics: Conference Series*, 1432:012087, 2020. 2

[5] Muhammad Kamran Javed Khan, Nizam Ud Din, Seho Bae, and Juneho Yi. Interactive removal of microphone object in facial images. *Electronics*, 8(10):1115, 2019. 2

[6] Amey Kulkarni, Amey Kulkarni, Amulya Vishwanath, and Chintan Shah. Implementing a real-time, ai-based, face mask detector application for covid-19, Oct 2020. 2

[7] Neuromation. Real-time face mask detection system, Oct 2020. 2

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 5

[9] C Szegedy, W Liu, Y Jia, P Sermanet, S Reed, D Anguelov, D Erhan, V Vanhoucke, A Rabinovich, et al. Going deeper with convolutions. arxiv 2014. *arXiv preprint arXiv:1409.4842*, 1409, 2014. 4

[10] 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, 2020. 3

[11] Xovis. Face mask detection, Oct 2020. 2

Appendix A. Code

The deliverable of this project can be found at https://github.com/TianxueHu/Face-Mask-Detection.

Tools:

- torch
- torchvision
- sklearn

Folder Structure:

- **dataset**: details in Appendix B.
- **mask_detect_exp.ipynb**: is the source code for processing data and train the model. The notebook was originally built in Google Colab, to run the notebook, please change file paths to the corresponding directory/folder.

Appendix B. Dataset

As detailed in Part3, our research conducted experiments on mainly two sources. First is Real-world Masked Face Dataset(RMFD). 5000 masked faces and 90,000 unmasked faces are consisted in the RMFD dataset. The second source is from Kaggle. We downloaded the picture of people wear face mask to increase the percentage of masked faces. We can get the dataset from https://github.com/X-zhangyang/Real-World-Masked-Face-Dataset and https://www.kaggle.com/andrewmvd/face-mask-detection.