Mask Detection and Classification with Different Convolutional Neural Networks

Qianru Li[1,*,†], Chenyu Song[2,†] and Xinhong Xie[3,†]

[1] School of Automation, Southeast University, Nanjing, Jiangsu, China

[2] School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR, USA

[3] School of Artificial Intelligence and Data Science, Hebei University of Technology, Tianjin, Hebei, China

[*]Corresponding author's e-mail: 213190947@seu.edu.cn

[†]These authors contributed equally.

Abstract. Since 2019, the COVID-19 has been hanging over the whole world, causing uncountable financial loss. In this regard, wearing masks becomes a precaution for the public. However, some people are wearing masks in a wrong way, which may cause virus infection. To detect the wrong wearing of masks, we use 3 classic Convolutional Neural Networks, namely LeNet-5, AlexNet, and VGGNet-16, based on a unique dataset, to train the model and analyze the results. On the unique dataset, LeNet-5 achieved an accuracy of 80.3%, which was the lowest among the three networks, AlexNet attained an accuracy of 90.6%, which is near the precision of VGGNet-16, 92.83%. This work may help the advance of a digital city, making COVID-19 precaution under control.

1. Introduction

Since 2019, the coronavirus disease (COVID-19) pandemic crossing worldwide has been the most serious problem in recent years. The World Health Organization (WHO) reported on 16th August 2020 that COVID-19 has already infected over 6 million people and cause over 300 thousand deaths. Until now, more than 100 million people have been infected, and also many people died because of it, as people know that the best way to control pandemics is to keep social distance and wear masks. Study shows that masks have protective efficacies in excess of 80% against clinical influenza-like-illness [1]. The mask here means medical masks instead of cloth masks, as simply cloth masks are often ineffective [2] . We can see that the virus through the cloth masks to be very high (97%) compared with medical masks (44%). To prevent the spread of COVID-19, more and more authorities ask people to wear medical masks in public areas, like the metro, market, and even streets. However, trying to monitor people by manpower is costly and almost impossible. They cannot set people everywhere like entrance, so using a machine to shot people's faces and find out if people wear masks correctly is cheap and convenient.

To help to keep security, many researchers engage in solving the problem of face mask detection, which is a key area in the field of Computer Vision and Pattern Recognition. We review the recent development of face mask detection and their methods.

Reference [3] focuses on mask detection based on ResNet, which is used for the object recognition benchmark. In that project, all problems related to a recognition task could be ensembled into three parts.

The first part corresponds to a baseline convolutional neural network (ResNet), which takes the job of extracting information and converting it into a feature map for further use. The second component contains all the pre-processing tasks that are needed for classification. The last one is the classifier in the model. By that model, it finally gains precision and recall of 98.86% and 98.22%, respectively.
In the model from [4], the first step is to transform each audio signal sample into an image-like representation by the discrete Short Time Fourier Transform. The learning model it uses is the ResNet to extract features as ResNet could eliminate the gradient vanishing problems. Then these features are given as input to an SVM classifier. And the output of SVM is whether the speaker wears a mask or not. The most inspired part is that they used Cycle-consistent GANS as a data augmentation method. Finally, they got a top score of 74.6%.
The model mentioned [5] is constructed in two parts. The first part is a face detection model from OpenCV, and with that model, they can obtain the number of faces and their location. In the second part, they use MobileNetV2 as a classifier. On its basis, to avoid any impartment to the weights loaded from TensorFlow, they add new training layers classifying if people wear masks or not, which could help reduce plenty of training time. Finally, they gained 92.64% on their datasets.
In their project [6], they created the dataset by Google and Bing API, crawled about 11000 images, and resized them into 416×416 pixels, which is just the input size of YOLO. Then they use the Labeling annotation tool to label approximately 50000 bounding boxes. Compared to other existing datasets, they often miss important information like annotation on people without masks or wearing masks incorrectly. Also, this dataset has achieved an mAP of 71.69% by original YOLO v4.

2. Method

2.1. Unique datasets and preprocessing

In this paper, we use the dataset with 11000 pictures evenly separated into four different ways of mask-wearing on each person, namely without a mask, with nose and mouth exposed, with nose exposed, and proper way of mask-wearing. **Figure 1** illustrates an example of the four types of mask-wearing. The dataset is open to the public at https://www.kaggle.com/tapakah68/medical-masks-part1.
To fit the dataset into the three networks, we do some preprocessing on the pictures. Firstly, we change the scale of all pictures into 224×224 to fit the networks. Secondly, we split the dataset in a ration of 70:15:15 respectively for the training set with 6700 images, the test set with 1476 images, and the validation set with 1476 images. Finally, we do some normalization on the pictures, making the values of each layer between 0 and 1.

Figure 1. An example of the 4 type's mask-wearing.

2.2. LeNet-5

LeNet-5 is the most monumented and also the earliest neural network which resulted from biology. It contains seven layers, and they are respectively a convolutional layer with 5×5 kernel, a 2×2 subsampling layer, a convolutional layer with 5×5 kernel, a 2×2 subsampling layer, two full connection layers, and a Gaussian layer.

LeNet-5 was inspired by animals' brains. With convolutional kernels, LeNet-5 could extract more representative features that are similar to the process in animals' brains. Then, the full connection layers and Gaussian layer could classify the input by the features.[7]
In this experience, we used PyTorch to build the model. First of all, we constructed the model with the required layimagesttowe to resize all the images to 224*224 and normalized all the pixels'ratio to 0 to 1. After that, we use random to divide the data set to train set and test set, accounting for 80% and 20%, respectively. During training, we used four different optimizers: Adam[8], SGD[9], RMSprop, and Adadelta[9].

2.3. AlexNet

AlexNet revealed the possibility of GPU computation and CNN. However, compared to LeNet-5, AlexNet has better performance as it has bigger convolutional kernels, which in other words, makes it has a bigger conceptive field. A bigger conceptive field means this model could extract better and more representative features.
AlexNet contains eight layers; the first one is a convolutional layer with 11×11 and a stride of 4 kernels, and then, a 3×3, two as stride max pooling layers. Then a convolutional layer with 5×5 and a pad of 2 kernels, followed by three convolutional layers with 3×3 kernel. The last is three full connection layers, and the results come from the last full connection.[10]
In our experience, we use Keras and Opencv to build the model. First, we construct the model with the required layers. Then, after resizing the image into 224×224 and doing normalization to make all pixels' values between 0 to 1, taking them as input of the network. To save memory, we set batch size as 64, and also to help train the most robust model, we set the dropout rate at 0.5, which means every neuron has 50 percent to be invalid. During training, we used four different optimizers: Adam[8], RMSprop, Adadelta[9], SGD[11].

2.4. VGGNet-16

VGGNet is a Convolutional Neural Network architecture proposed by Karen Simonyan and Andrew Zisserman from the University of Oxford in 2014. This paper mainly focuses on the effect of the convolutional neural network depth on its accuracy.
As showed in Figure 2, this network is characterized by its simplicity, using only 3×3 convolutional layers and 2×2 pooling layers stacked on top of each other in increasing depth. Reducing volume size is handled by max pooling. Two fully-connected layers, each with 4,096 nodes, are then followed by a softmax classifier.
Compared to AlexNet, the VGGNet can reduce the number of parameters by piling two 3*3 convolutional kernels to replace one 5*5 convolutional kernel and piling three 3*3 convolutional kernels to replace one 7*7 convolutional kernel. This method can work because their receptive fields are the same. [12]
In our experience, by training the VGG-16 Network, we set the learning rate to be 0.01 and used the optimizer SGD and Adadelta. The dropout rate is set to be 0.5. The best result gained is the model with optimizer SGD, and the learning rate is 0.01, which reaches the highest accuracy at the 31st epoch.
By training the VGG-16 Network, we set the learning rate to be 0.01 and use the optimizer SGD and Adadelta. The dropout rate is set to be 0.5.

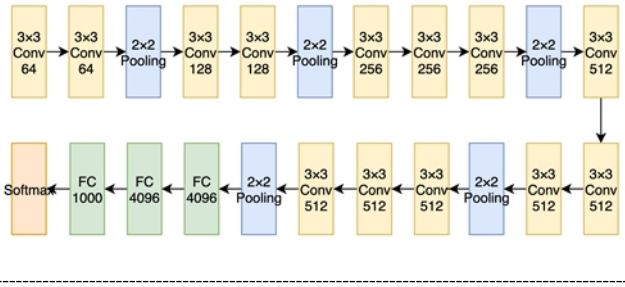

Figure 2. The structure of VGGNet-16.

3. Results and discussion

To evaluate the effectiveness of our three models, the research is conducted to answer the following question:

3.1. Experiment setup

The experience is consisted of three parts, three different models; the first part is LeNet constructed by PyTorch; by stacking all the layers mentioned in [7], we can use PyTorch to build the model and set the batch size to 70. We also operate four different optimizers. The learning rate of the Adam optimizer is 1e-3, the learning rate of the SGD optimizer is 1e-2, and the momentum is 0.9. The learning rate of the optimizer RMSprop is set to 1e-6, the alpha is 0.99, and the eps is 1e-08. For the Adadelta optimizer, we set rho to 0.9.

The second one is AlexNet; this model is easily implemented with Keras and OpenCV. By stacking layers mentioned in[10], we can build our AlexNet and set 64 as batch size. We also use four different optimizers. The Adam optimizer has 1e-3 as the learning rate. The learning rate of RMSprop is also 1e-3, and it also has epsilon=1e-8. The Adadelta's learning rate is 0.05, which is the highest one. The last one is SGD, which performs best; it has 1e-2 as a learning rate, with momentum=0.9 to prevent the "Canyon" and "Saddle point" problem.

The third part is VGGNet-16, which can be deployed by TensorFlow and Keras, by stacking layers with five pooling layers and 13 convolutional layers, setting batch size as 16, dropout rate as 0.5, learning rate as 0.01, we use SGD and Adadelta optimizers to build and train the model.

3.2. Results and evaluation of the three networks

In this section, we will discuss the performance of our models and decide which one is the most effective in this experience. We will compare LeNet-5, AlexNet, VGGNet-16 basing on the following aspects:

- Optimizer analysis: in a model, which optimizers perform the best.
- Model analysis: between different models, which model has the best result.

Table 1. A slightly more complex table with a narrow caption.

OPTIMIERS	Accuracy		
	Training Accuracy	*Validation Accuracy*	*Test Accuracy*
PMSprop	80.0%	57.1%	68.6%
Adadelta	79.9%	77.1%	74.2%
Adam	80.0%	75.7%	71.7%
SGD	79.8%	72.9%	**76.9%**

From the result showed in Table 1, we can see that LeNet-5, as one of the earliest neural networks, doesn't perform well in this data set, but still, after trying different optimizers (PMSprop, Adadelta, Adam, SGD), respectively get results from 79.9% to 80.0% in training data, and 57.1% to 77.1% in validation data. The best one could have around 76.9% accuracy on test data with optimizer SGD.

Table 2. Optimizer analysis on AlexNet.

OPTIMIERS	Accuracy		
	Training Accuracy	*Validation Accuracy*	*Test Accuracy*
PMSprop	92.4%	86.3%	88.2%
Adadelta	93.6%	91.4%	88.4%
Adam	87.3%	82.5%	77.4%
SGD	94.2%	93.2%	**90.6%**

About the results of AlexNet showed in Table 2, we can easily find out that its performance dramatically outweighs that of LeNet-5. The reason is easy to find out that due to AlexNet has much more convolutional layers to extract features. With different optimizers, our network respectively get 87.3% accuracy to 94.2% in training data and get 82.5% to 93.2% in validation data. In the end, the model also reaches 90.6% accuracy on test data.

Table 3. Optimizer analysis on vggnet-16.

OPTIMIERS	Accuracy		
	Training Accuracy	*Validation Accuracy*	*Test Accuracy*
Adadelta	97.1%	92.3%	87.7%
SGD	98.4%	94.5%	**92.8%**

The last model is VGGNet-16, which we believe will have the best performance; the results in Table 3 that come out that this model, as expected, achieves the best outcome that achieves about 92.8% on test data with optimizer SGD.

Table 4. Performance of the three networks with best results.

NETWORKS	Accuracy	
	Training Accuracy	*Test Accuracy*
LeNet-5	79.8%	76.9%
AlexNet	94.2%	90.6%
VGG-16	98.4%	**92.8%**

After we compare these three models, the results in Table 4 are obvious here that VGGNet-16 reaches the best results as we expect it should be which get 92.8% accuracy with SGD optimizer in test accuracy.

4. Conclusion

In this article, we aim to detect whether the individuals are wearing the mask or not using the deep learning method. Three deep learning classic network models are used to identify people's facial masks, including LENet, AlexNet, and VGG-16. We first collect the dataset from Kaggle and apply the data preprocessing. The processed images are sent to those models to obtain the results. After training and verification, the results show that VGGNet-16 can achieve the highest accuracy rate of 92.8% under the

SGD optimizer. Our model provides a brief attempt on mask detection application, which can be used in a real-world application with optimization.

References

[1] Y. Wang, Z. Deng, and D. Shi, “How effective is a mask in preventing COVID‐19 infection?,” Med Devices Sens, vol. 4, no. 1, Feb. 2021, DOI: 10.1002/mds3.10163.Podani, J. (1994) Multivariate Data Analysis in Ecology and Systematics. SPB Publishing, The Hague.

[2] L. Szarpak, J. Smereka, K. J. Filipiak, J. R. Ladny, and M. Jaguszewski, “Cloth masks versus medical masks for COVID-19 protection,” p. 2.Thompson, J.N. (1984) Insect Diversity and the Trophic Structure of Communities. In: Ecological Entomology. New York. pp. 165-178.

[3] S. Sethi, M. Kathuria, and T. Kaushik, “Face mask detection using deep learning: An approach to reduce risk of Coronavirus spread,” Journal of Biomedical Informatics, vol. 120, p. 103848, Aug. 2021, doi: 10.1016/j.jbi.2021.103848.

[4] N.-C. Ristea and R. T. Ionescu, “Are you wearing a mask? Improving mask detection from speech using augmentation by cycle-consistent GANs,” arXiv:2006.10147 [cs, eess], Jul. 2020, Accessed: Sep. 10, 2021. [Online]. Available: http://arxiv.org/abs/2006.10147

[5] P. Nagrath, R. Jain, A. Madan, R. Arora, P. Kataria, and J. Hemanth, “SSDMNV2: A real time DNN-based face mask detection system using single shot multibox detector and MobileNetV2,” Sustainable Cities and Society, vol. 66, p. 102692, Mar. 2021, doi: 10.1016/j.scs.2020.102692.

[6] A. Kumar, A. Kalia, K. Verma, A. Sharma, and M. Kaushal, “Scaling up face masks detection with YOLO on a novel dataset,” Optik, vol. 239, p. 166744, Aug. 2021, doi: 10.1016/j.ijleo.2021.166744.

[7] Y. LeCun, Y. Bengio, and T. B. Laboratories, “Convolutional Networks for Images, Speech, and Time-Series,” p. 14.

[8] D. P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” arXiv:1412.6980 [cs], Jan. 2017, Accessed: Sep. 22, 2021. [Online]. Available: http://arxiv.org/abs/1412.6980

[9] M. D. Zeiler, “ADADELTA: An Adaptive Learning Rate Method,” arXiv:1212.5701 [cs], Dec. 2012, Accessed: Sep. 22, 2021. [Online]. Available: http://arxiv.org/abs/1212.5701

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” Commun. ACM, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[11] S. Sra, S. Nowozin, and S. J. Wright, Eds., Optimization for machine learning. Cambridge, Mass: MIT Press, 2012.

[12] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” arXiv:1409.1556 [cs], Apr. 2015, Accessed: Sep. 11, 2021. [Online]. Available: http://arxiv.org/abs/1409.1556