CS767 Term Project
Mask detection using transfer learning based on InceptionV3 NN

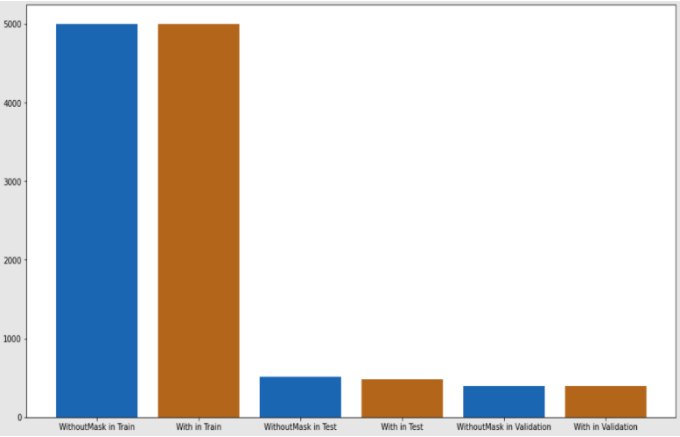
Tianyi Liang

In this epidemic, the use of masks is essential. For some public occasions, supervision means are needed to ensure the use of masks. It is not only about keeping oneself safe, but also a responsibility to others around. Using a human detection at the entrance to monitor whether people are wearing masks is a risk for the monitoring personnel. It is possible to use machine detection to reduce the use of humans also protect them from the risk of infection.

In the project, I intend to train a highly reliable model to detect whether a person is wearing a mask using Convolutional Neural Network, Wide and Deep Network, and Transfer learning methods.

Dataset

The data I use to train, and evaluate, the model comes from the public domain of Kaggle (https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset), which contains three folders (train, test, and validation sets) of images. There are 118,000 images of human faces marked as “WithMask” or “WithoutMask”. The data is well-balanced in terms of the distribution of the two classes.



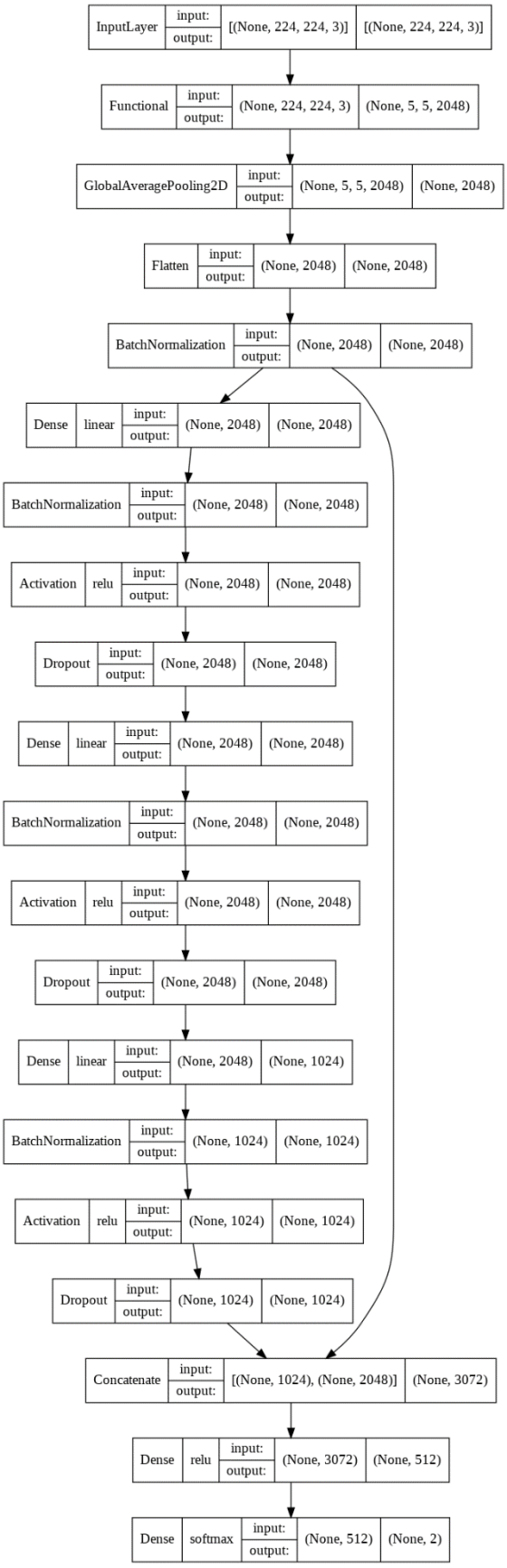
Data Preprocessing

The resolution of the images in the dataset is very inconsistent. For the convenience of subsequent training, images need to be upsampled/downsampled to the same dimension. In order to protect the images from loss of information due to dimensionality reduction and distortion, I chose a relatively larger resolution (224 X 224 X 3) as the unified resolution. This action may raise the required RAM of the script significantly, but it is worth considering the loss of information.

Min-max normalization is applied to the dataset considering this is a problem related to images. It is also computationally cheaper to divide the whole dataset by 255 which is always the maximum of image arraies.

Data Augmentation

Considering the small size of the training set (only 10k images), I also applied image augmentation in the training set with rotation(30°), height/width shifting(10%), zomming(10%), and horizontal flipping. This enables the training set to go through a random transformation every epoch. This figure is a demonstration of the transformation generator performing 20 epochs only at the first element of the training set. These images do not look very different to human eyes, but when feeding them into the neural network, the network treats them as 20 completely new, distinguishable training sources. Due to the huge number of weights of Wide & Deep Neural Networks, a constant and small amount of training data will eventually lead to overfitting. The line formed by weights in the higher dimensional space will end up going through exactly all data points, which makes the model lose all generalizability. A transformation generator makes it nearly impossible for the model to meet exactly the same image twice.



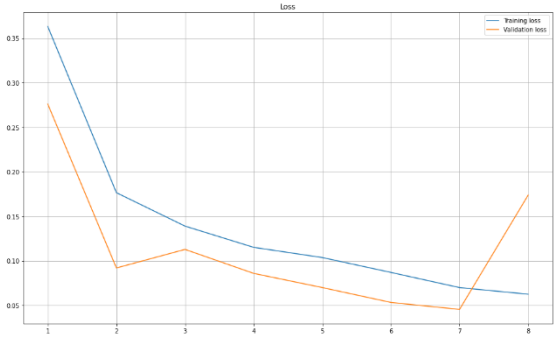
Network Structure

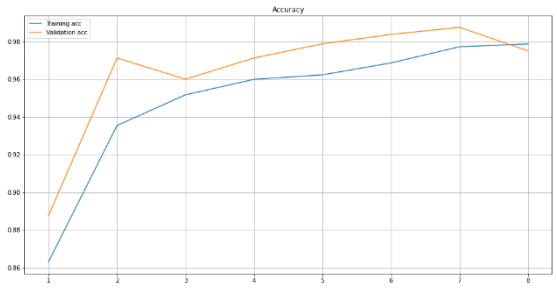
By implementing Keras Application "*InceptionV3*", the first part of my network inherits the structure of "*InceptionV3*". I did not use the pre-trained weights "*imagenet*" of "*InceptionV3*". I chose this built-in structure based on its ability to process image problems. Every block in this structure contains several paths. In each of its paths, it uses many small convolution kernels instead of larger kernels, which reduces the number of parameters, and applies asymmetric convolution kernels like (1X3). This well-known network structure has numerous applications in computer vision.

The "*InceptionV3*" network gives a (1000, 1) dimension output with softmax activation. The rest of the network is a self-defined Wide and Deep Network using Keras functional API with implementations of average pooling, batch normalization, Glorot kernel initializer, dropout, and various activation functions. All the details regarding to perceptron numbers, activation functions, and orders of the layers could be found in this figure. It is worth noting that I concatenated the output of "*InceptionV3*" with the output of the previous Dense layer before connecting to the last Dense layer. This improves the performance of the network significantly. I also implemented a self-defined learning rate scheduler as a callback function. Stochastic gradient descent is used as the optimizer of the model, which gives the best performance after trying out all the others.

Assess the model

By feeding in the transformation generator with the training set to the network, the model reaches 98.75% accuracy on validation set at the seventh epoch. The figures show how the accuracy and loss change during training.



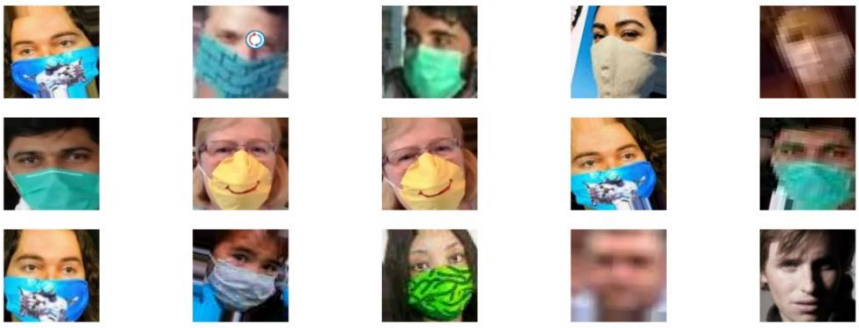


Evaluation is also done on the test set. 97.28% is given by the finalized model. To further assess the result, I also calculated the four values of the confusion matrix (TN, FN, TP, FP).

TP: 502 FP: 20
TN: 463 FN: 7

By defining WithoutMask as the positive event, recall rate is given by TP / (TP + FN) (98.62%). Precision rate is given by TP / (TP + FP) (96.17%), and Specificity rate is 95.86%. In this case, we are more careful about the recall rate rather than precision and specificity rate. Correctly indentifying each positive evnet (Without mask) is the core of the model. To achieve this goal, a few false alarms to those who wear masks are not intolerable. Missing an alarm to an unmasked person is the worst thing in this case. Recall rate is the highest over all the three metrics, which is exactly what we want to see.

As a reference, I have also printed those individuals whose predicted results did not match their actual status. It can be observed from the figure below that most of them are wearing masks. Some of them are wearing very special masks; others even have a pattern of mouths on their masks. Some images have very low origin resolution or high contrast between light and dark. It is not so hard to understand why the model is making wrong predictions on these individuals.



Reference

Jangra, A. (2020, May 26). *Face mask detection ~12K images dataset*. Kaggle. Retrieved April 14, 2022, from https://www.kaggle.com/datasets/ashishjangra27/face-mask-12k-images-dataset

Team, K. (n.d.). *Simple. flexible. powerful.* Keras. Retrieved April 14, 2022, from https://keras.io/

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2016.308