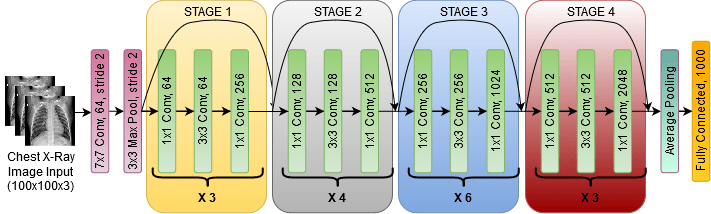
**Introduction to Artificial Intelligence IC-PBL Project Report**

1. **Diagram of the Resnet-50 architecture and detailed descriptions**



(source: https://open-instruction.com/dl-algorithms/overview-of-residual-neural-network-resnet/)

Before the advent of ResNet50, the most commonly used method of improving the performance of DNNs was to stack as many layers as possible and make the model deeper. However, the main problem was that the vanishing gradient problem obstructed the training of deeper DNN models and ultimately impeded model optimization. This led to the introduction of ResNet50, a CNN architecture that consists of several residual blocks that further improved the performance of the existing DNNs.

This architecture is composed of the following:

* A convolution layer with kernel size of 7x7, 64 different kernels and stride size 2
* A max pooling layer with kernel size of 3x3 and stride size 2
* Four residual block stages consisting of convolution layers with kernel size and quantities each shown in **Figure 1**
* An average pool and fully connected layer with 1,000 nodes

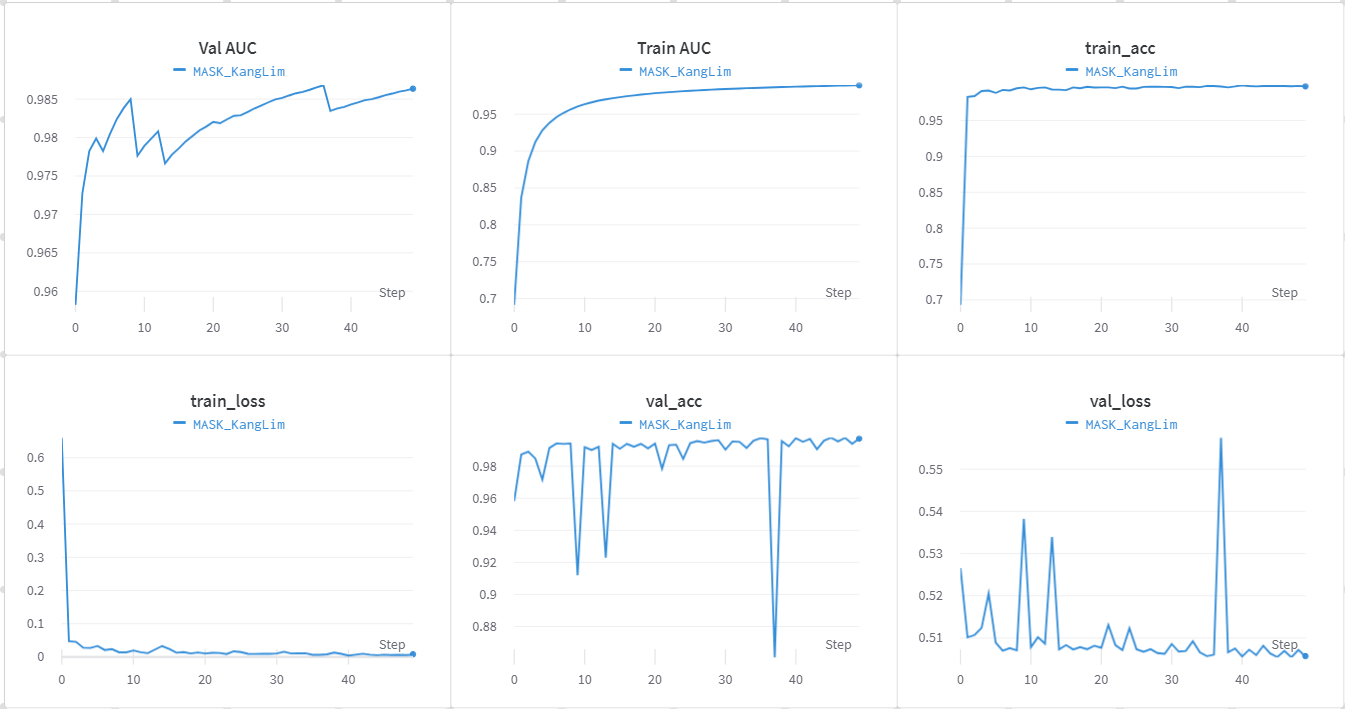
This gives 48 convolution layers in the inner four stages, 1 max pooling and 1 average pooling layer each taking part in the residual blocks, adding up to the total of 50 layers.

1. **Some samples of augmented images using MaskTheFace**

|  |  |  |
| --- | --- | --- |
|  |  |  |

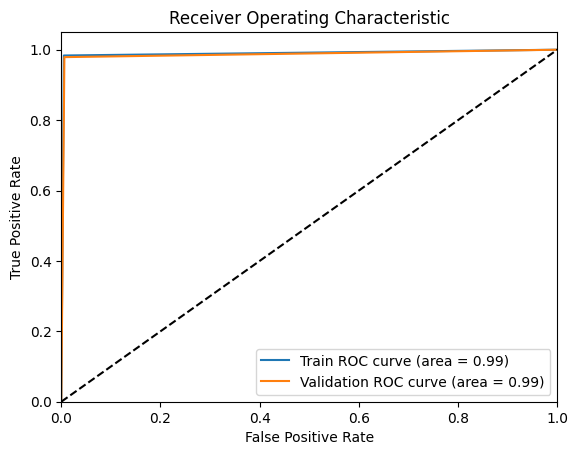
The types of masks chosen for augmentation are N95, KN95, and cloth, and the colors are white(#FFFFFF), black(#000000), and beige(#E9DAC4). These mask types and colors were chosen to be implemented in the training code in order to reflect the most prevalent appearances of masks used today.

1. **Plots**

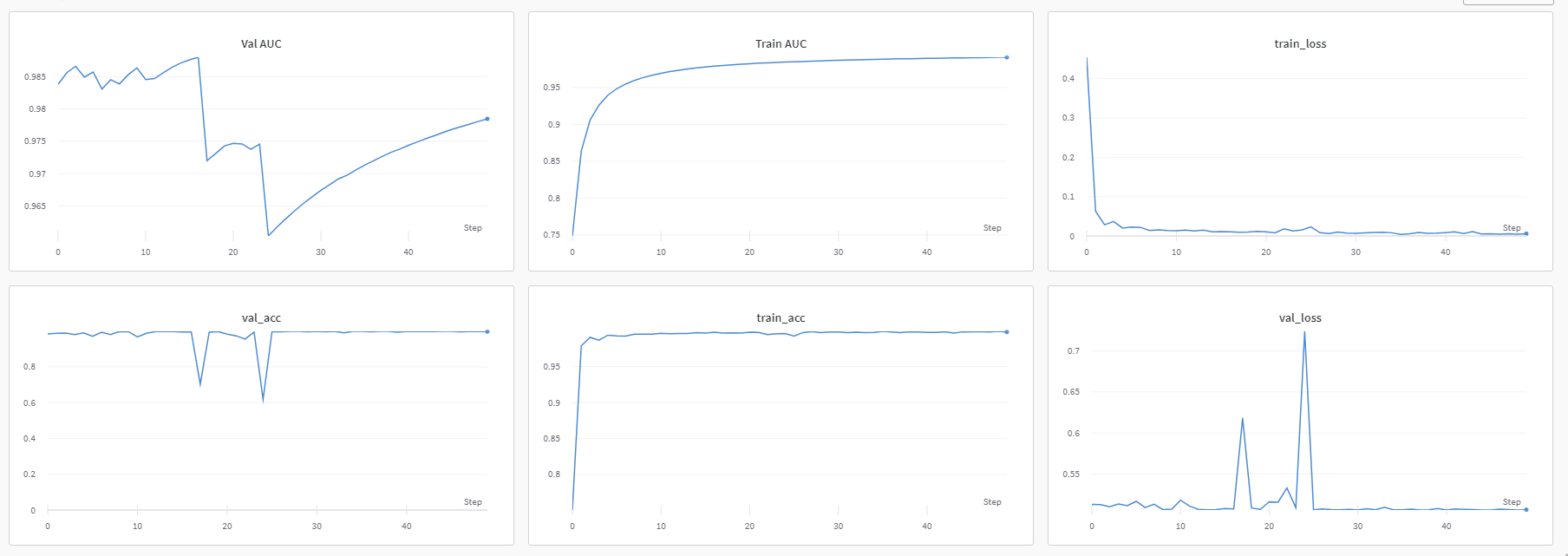
****

No learning rate scheduler, pretrained: false. At epoch 50,

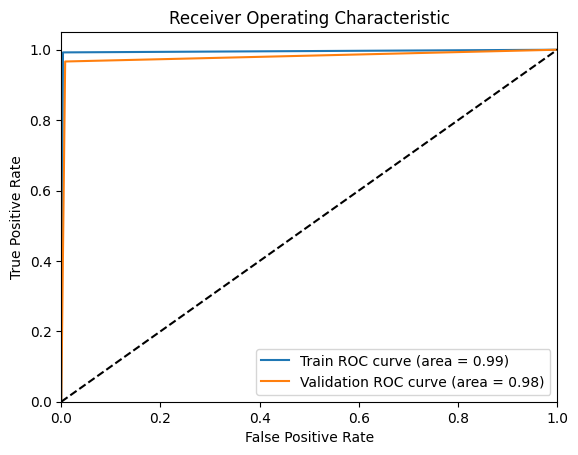
* train\_acc: 0.99748, train\_loss: 0.00799, train\_auc: 0.9892
* val\_acc: 0.99722, val\_loss: 0.5057, val\_auc: 0.98635

****

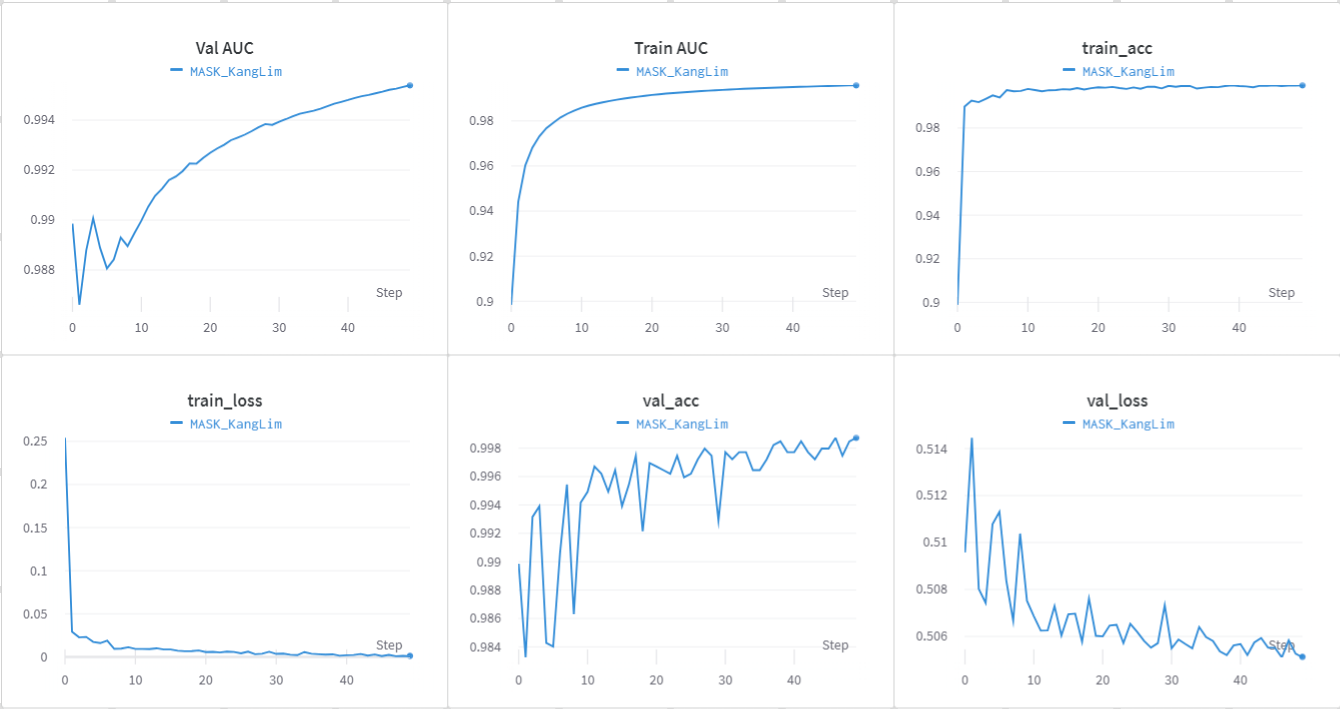
1. **Discussions**
   1. **2 ablation studies**
      1. **Setting model pretrain to True**

At epoch 50,

* train\_acc: 0.99815, train\_loss: 0.00568, train\_auc: 0.99114
* val\_acc: 0.99722, val\_loss: 0.50624, val\_auc: 0.97848

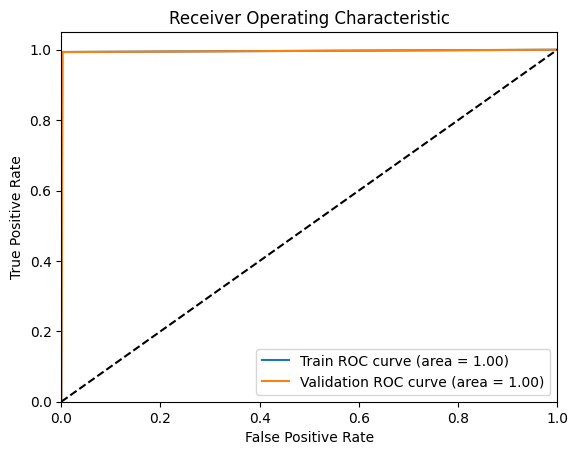


* + 1. **Setting model pretrain to True and also implementing learning rate scheduler(Changing hyperparameters)**

****

At epoch 50,

* train\_acc: 0.99958, train\_loss: 0.00116, train\_auc: 0.99579
* val\_acc: 0.99873, val\_loss: 0.50513, val\_auc: 0.99539

****

From the ablation studies, it is noticeable that the overall performance in terms of accuracy, loss and AUC improves when setting the model pretrain to True and even more when also introducing the learning rate scheduler. In the case of AUC, its trend was smoother when setting the model pretrain to true, but the figure was smaller than when setting it to false. This is because of a sudden deep downward spike in val\_acc, and we can see that even if that sudden spike negatively affected validation AUC, the number of spikes decreased conspicuously. In the second ablation study, the validation loss converges to a smaller number in a gradually stable state when compared to the original model and the first ablation study where it oscillates.

* + 1. **Limitations of current model**

The model performed successful learning of the given train and validation images, but the model learned the images “so perfectly’ that it had difficulties analyzing the test images consisting of pictures of our acquaintances below. The model showed relative inaccuracies in classifying images with rotation, high contrast, low brightness, and/or obstructions of facial features by, e.g., fingers partially covering the facial area.

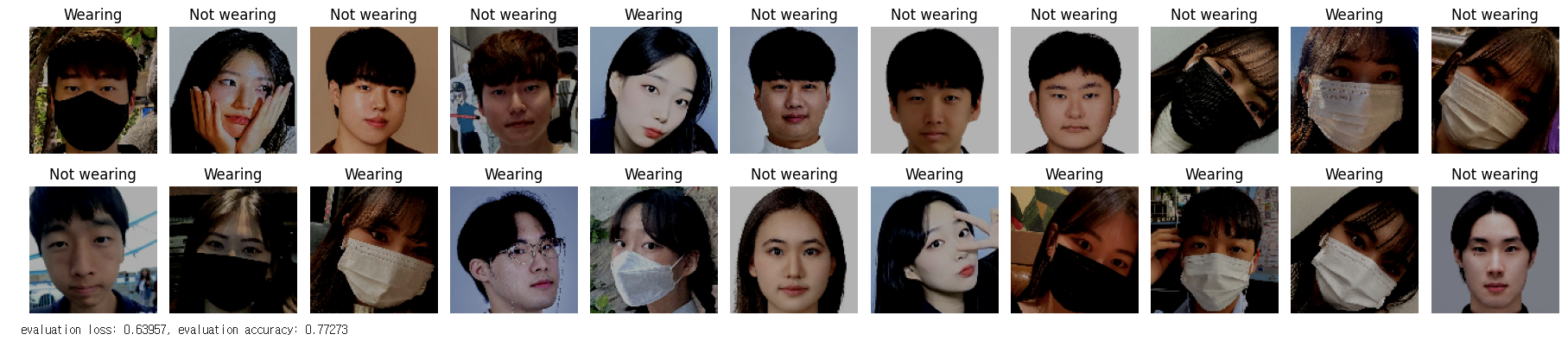
1. **Classification results on some of our acquaintances’ photos**
   1. **lr = 0.01, no lr scheduler, pretrained = false**

****

When giving 22 images (11 masked and 11 unmasked),

eval\_loss: 0.61138, eval\_acc: 0.81818

* 1. **lr = 0.01, no lr scheduler, pretrained = True**

****

When giving 22 images (11 masked and 11 unmasked),

eval\_loss: 0.63957, eval\_acc: 0.77273

* 1. **lr = 0.01, learning rate scheduler: LambdaLR, pretrained = True**

****

When giving 22 images (11 masked and 11 unmasked),

eval\_loss: 0.61138, eval\_acc: 0.81818

**References**

<https://junstar92.tistory.com/121>

<https://justkode.kr/deep-learning/pytorch-save/>

<https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html#sklearn.metrics.roc_curve>