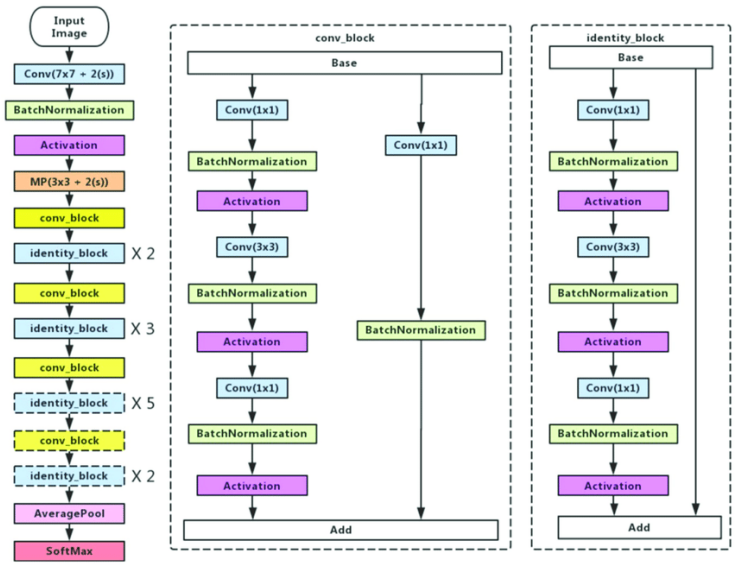
Castro Sanchez, Paula

Recognizing Masked Faces

ICPBL Project - Introduction to AI

ResNet50

For this project, we used ResNet50 architecture as a basis for our CNN. The diagram below illustrates its basic structure, having 48 convolution layers along with 1 MaxPool and 1 Average Pool layer:



It starts with a convolution with a kernel size of 7 * 7, using 64 different kernels with stride of size 2. Then, we see normalization, activation, and max-pooling layers (stride size of 2 and kernel size 3*3).

Then we find the first conv_block. It is a succession of 3 convolutions (1 * 1,64 ; 3 * 3,64 ; and 1 * 1,256 respectively), where each is followed by normalization and activation layers. To this, we add a skip connection, that uses a 1*1,64 kernel plus normalization on the input of the block. It appears in total 4 times across the network, giving us 12 layers.

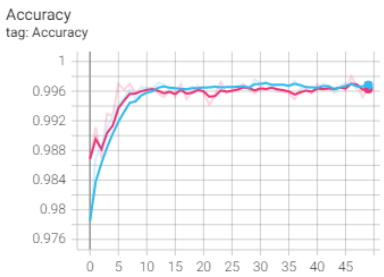
Next, we see the identity block. It also has 3 convolutions, of sizes 1 * 1,128 ; 3 * 3,128 and 1 * 1,512. Each convolution layer is again followed by normalization and activation layers. We have a skip connection too, adding to our result the input of the block. These appear 12 times in total, giving us 36 layers total. The exact order of succession between conv and identity blocks can be seen in the diagram above. Finally, we apply average pooling and a fully connected layer containing 1 node. The output value is a floating number. We will only classify a face image as having a mask if the output is positive.

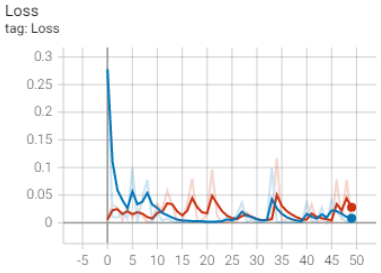
Our Data and Results

We had to create our own synthetic data for the masked faces, as no relevant dataset was available. This was done with 'MaskTheFace' software, and some examples can be visualized below:

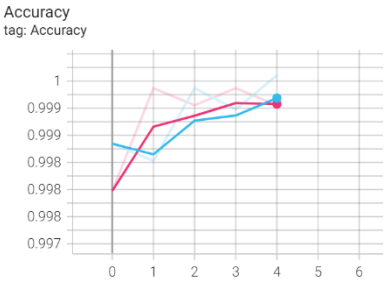


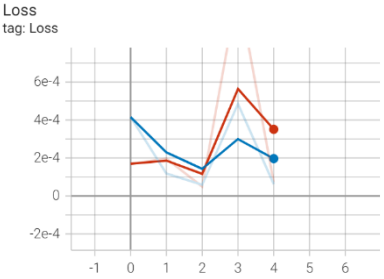

Once our dataset was ready, we were ready to start the training process. For benchmarking this research, we first used the setting suggested in the Report Guidelines. It produced the following plots, where the test stats are depicted in blue, and validation ones in pink/red:





Then, we set out to study how changing different parameters would affect these curves. We started out by changing the weight initialization, using those of the original ResNet50 (except for the final layer, which we had changed). Based on the success of transfer learning on many other tasks, we expected some good results from this model, where it would take very few iterations to achieve maximal performance. This was indeed the case:



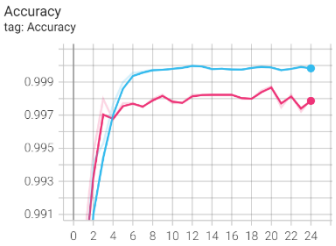


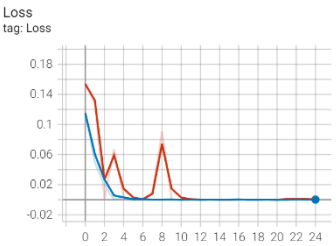
As we can see, in the first epoch the stats were outstanding, even better than those we obtained in the previous model over 50 iterations. Near the 5th iteration already, performance was close to perfect.

These results are more impacting when seeing the actual statistics for the first epoch:

```
Epoch: 0 , Train Loss: 0.00042, Train Accuracy: 99.87%, Validation Loss: 0.00017, Validation Accuracy: 99.80%
```

With this, we can affirm that setting good initialization can definitely help our CNNs converge faster and train better. Now, we wanted to look at the effect of not using data augmentation. Our expectations were that the model would quickly overfit, and we would therefore see the val_acc drop. Let's see the results:

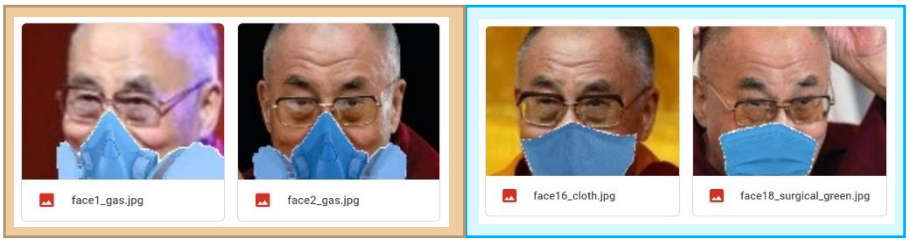




As we can see, now the validation accuracy doesn´t follow the training accuracy, as happened in the previous cases. This is possibly due to the overfitting.

However, we still observe that the validation accuracy, in all the studies we´ve conducted, is extremely high, and it is sometimes even greater than the training accuracy. One of the reasons is that the size of our testing set is very small. However, if this were to happen in a dataset with many more testing

examples, we should look into it. Secondly, after going through the training and validation sets, we can easily realize they are very related to each other. Below, training (left) and validation (right) samples:



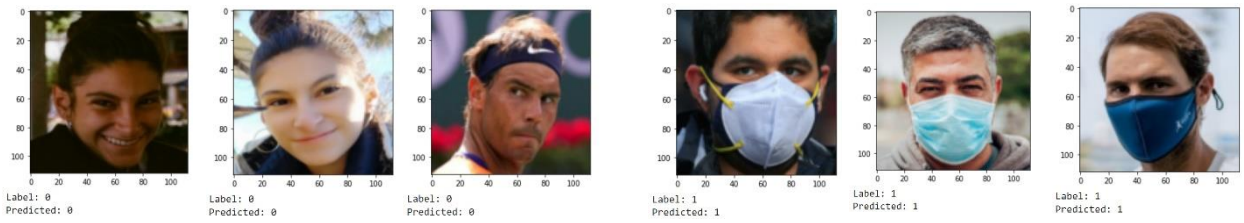
Through these ablation studies, we learn the importance of initialization and data augmentation for the optimal performance of our models.

Results and Limitations

We list here some of the limitations of our model:

- Our dataset is very ‘clean’, where the pictures have already been cropped and aligned. Therefore, the generalizability of the model to real-world scenarios may be limited.
- Currently, the model has some usage delay, which may make it un-deployable for real-time evaluation at kiosks. It is a simple problem, for which we are training a very complex network (starting acc extremely high), so maybe if we used a simpler architecture it would be more deployable.
- We have augmented the images ourselves to put masks on them. For starters, this may mean that some of the images are a bit ‘off’. Also, nowadays masks are seen as another fashion item, and we find various designs and patterns. The network may have trouble identifying these.

We then proceeded to test on custom images. On the first execution, we tested the model on the original images. The accuracy was 67% (correctly classifies 4 out of 6 images). Then, we cropped and centered the faces in the images to resemble the training data. Now, the accuracy was 100%:



This very clearly illustrates one of the limitations mentioned in our model.

The code for reproducing these results, as well as that related to the ablation studies, can be found in the training and evaluation scripts respectively.

All in all, this project has really helped me deepen my understanding of neural networks, especially of CNNs. I have really enjoyed it 😊.