COMP 6721 Applied Artificial Intelligence
Project Assignment, Part I

AI Face Mask Detector

Department of Computer Science and Software Engineering

Instructor: Dr. René Witte

Team number: NS_01
Team members:
Seyed Pouria Zahraei (40115175)
Siva Naga Sai Vemula (40151579)
Amirali Khedmati (40137238)

Winter 2021

Part 1

Dataset

The preprocessing of the data is an important part for each machine learning project. In this project we gathered 1592 images. These images have been separated into three different classes. 600 images for Not Person, 483 images for With Mask and 509 images for Without Mask. All images have png format.
We gathered these sample images from Kaggle website [1],[2]. For instance, for the Not Person images, we used the photo of cats and some images of people with masks and without masks. Below you can find the statistics of our dataset.

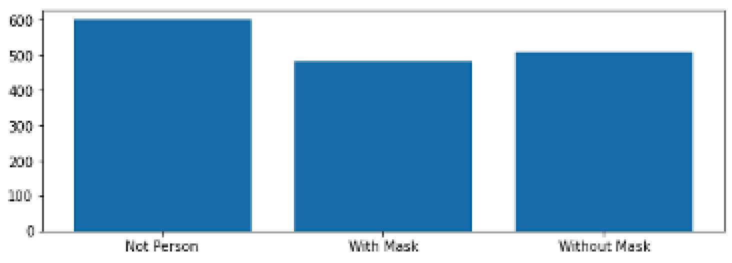


Figure 1: statistics of dataset

At first we resize the images to 32*32 and transformed them into Tensors of normalized range [-1,1].[3]
We split our dataset into training data and testing data. The training data is 75% of the full data set and the testing data is 25% of the original data.
Since we are working on 1592 images, working on this amount of data can be memory consuming and it will take some much time. In order to solve this problem, we used the DataLoader [4] from Pytorch. Dataloader takes *batch_size* for indicating the number of samples contained in each generated batch, *num_workers* for indicating number of processes that generate batches in parallel.

CNN Architecture:

We have used 3 convolution blocks(block = (convolution + batchNorm + RELU + maxpool) to train images of different classes-{with /without mask, not a person} of a dataset. In general each block applies a filter(kernel), padding(avoid losing edge layers), and max pooling to learn features. Please find the detailed steps below.

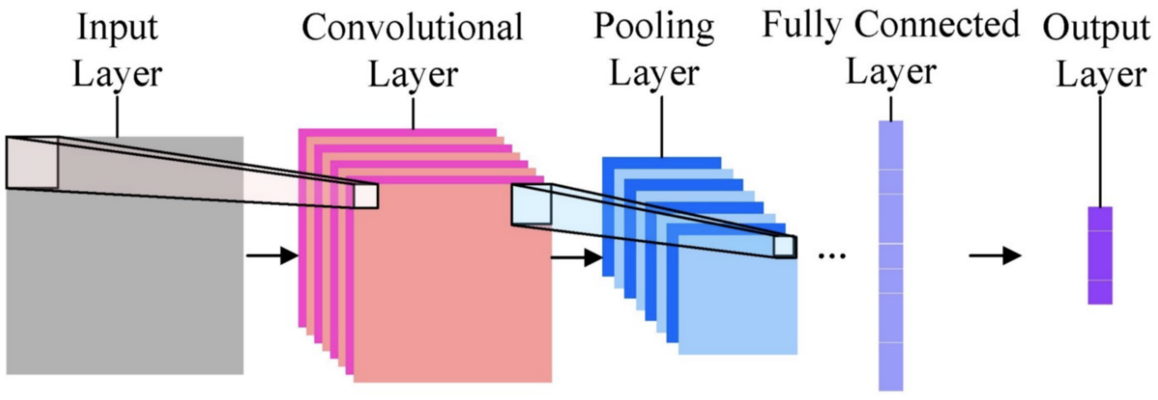


Figure 2: CNN Architecture (src)

1. Given the image size = 32 x 32 of 3 input colored channels, i.e represented in a 3D matrix.
2. Provide **filter** size 3 x 3 and apply **padding = 2**, this makes input matrix 32 x 32 and we have 32 such filters, thus makes matrix of 32 x 32 x 32
3. Next convolution(dot product) is performed using the above filter with **stride=1** creates output matrix.
4. Improve the learning rate by applying batch normalisation, so the mean becomes 0 and variance becomes 1; which improves the learning rate of the model.
5. Then max pooling is applied on the result of step 4; Thus completes a single convolution block.
6. Repeat steps 2 to 5 for 3 times (3 blocks with different params depicted in below table).
7. So, the final data gets transformed to a single numpy array and then fed as an input to a fully-connected layer; which provides the result.

8. Finally train the images of each class with **20 epochs**(iterations). With more iterations improves the performance of the model and sometimes this may lead to overfitting as well.
9. Next we assign probability values for the trained network result using a softmax function to avoid the overflow errors.
10. Then the above probability values can be used to evaluate accuracy on the train data after every epoch and test data.

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 32, 32, 32]             896
       BatchNorm2d-2          [-1, 32, 32, 32]              64
         LeakyReLU-3          [-1, 32, 32, 32]               0
            Conv2d-4          [-1, 32, 32, 32]           9,248
       BatchNorm2d-5          [-1, 32, 32, 32]              64
         LeakyReLU-6          [-1, 32, 32, 32]               0
         MaxPool2d-7          [-1, 32, 16, 16]               0
            Conv2d-8          [-1, 64, 16, 16]          18,496
       BatchNorm2d-9          [-1, 64, 16, 16]             128
        LeakyReLU-10          [-1, 64, 16, 16]               0
           Conv2d-11          [-1, 64, 16, 16]          36,928
      BatchNorm2d-12          [-1, 64, 16, 16]             128
        LeakyReLU-13          [-1, 64, 16, 16]               0
        MaxPool2d-14            [-1, 64, 8, 8]               0
           Conv2d-15           [-1, 128, 8, 8]          73,856
      BatchNorm2d-16           [-1, 128, 8, 8]             256
        LeakyReLU-17           [-1, 128, 8, 8]               0
           Conv2d-18           [-1, 128, 8, 8]         147,584
      BatchNorm2d-19           [-1, 128, 8, 8]             256
        LeakyReLU-20           [-1, 128, 8, 8]               0
        MaxPool2d-21           [-1, 128, 4, 4]               0
================================================================
Total params: 287,904
Trainable params: 287,904
Non-trainable params: 0
```

```
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 2.73
Params size (MB): 1.10
Estimated Total Size (MB): 3.84
```

Evaluation

In this project we evaluated the model by using 398 images as test data,which was selected randomly and consisted of three classes(with mask, without mask, not a person). For the evaluation we used a confusion matrix to describe the performance of the classification model on the test data. And classification_report in order to show the accuracy, precision, recall and F1-score.

As you can observe in the Figure 3 we provided the confusion matrix, which indicates that this model predicted 5 times the class "Without Mask" as "With Mask" and 5 times "Not A Person" as "Without Mask" and 3 times predicted class "Without Mask" as "Not A Person".

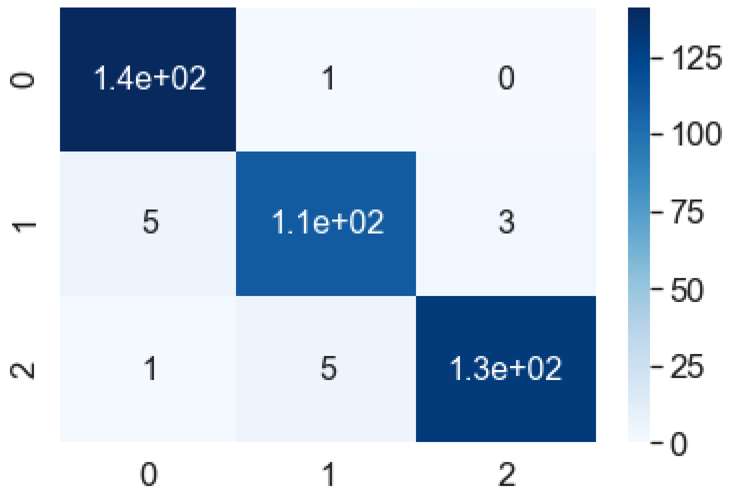
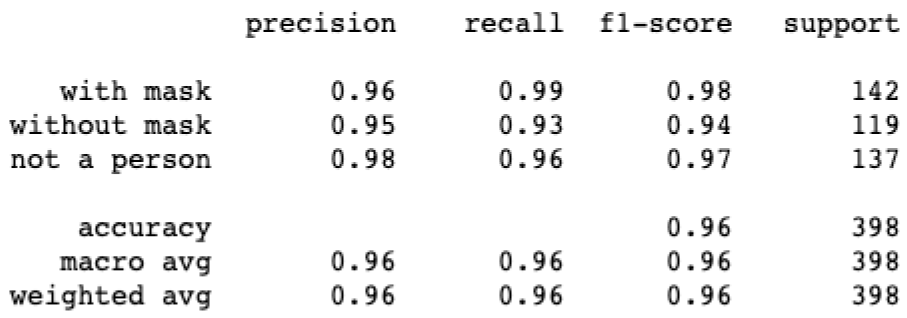


Figure 3: Confusion matrix

In Figure 4 we illustrated the classification report, which contains the accuracy of the model for the test data, also includes precision, recall and F1-score for all classes separately. In the table below we got the best precision of 98% for "Not A Person" and 96% for "With Mask" then 95% for "Without Mask". However if we take a look at the recall and f1-score, we can see that the better result was for the "With Mask" with 99% for recall and 98% for f1-score.

In terms of accuracy of the model, we received the accuracy of 96% for our testing data, which needs to get improved and we will discuss it in this section.



```
              precision    recall  f1-score   support

   with mask       0.96      0.99      0.98       142
without mask       0.95      0.93      0.94       119
not a person       0.98      0.96      0.97       137

    accuracy                           0.96       398
   macro avg       0.96      0.96      0.96       398
weighted avg       0.96      0.96      0.96       398
```

Figure 4: Classification report

As we mentioned above we trained the images with 20 epochs. Looking at the logs of our training accuracy and training loss we observed that after each epoch the training accuracy and training loss has been improved. In order to improve the performance of the model for the next phase we have different approaches.

The first approach, which could lead us to improve the accuracy is to improve some parameters such as, increase the number of epochs. The next approach to use cross validation in order to avoid overfitting, which is not that significant but it could be helpful.

Part 2

Bias

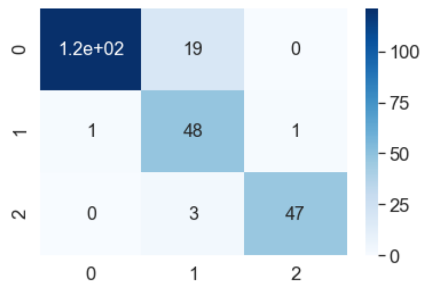
In this section we are going to analyze the gender attribute. In order to test our model we used 240 images for each Female and Male consisting of 140 Not Person, 50 With Mask and 50 Without Mask. After training our model using our old dataset and testing our model for each of these categories(Female, Male, All) we noticed that the accuracy of our model for each of them has dropped around 9%. Therefore, we decided to re-balance our dataset. This time for our training dataset we divided images equally between these categories.

Now our new dataset has 1800 images. 600 images for NotPerson, 600 images WithMask(300 Female, 300 Male), 600 WithMask(300 Female, 300 Male).

For the complete testing dataset we did not notice any changes. However, for the dataset that we chose for the bias attribute subset we noticed much lower testing accuracy. 90% accuracy for the Female dataset and 87% for the Male dataset. After re-balancing the dataset and training the model we were able to improve the testing accuracy for each of these categories. 97% for Female dataset and 98% for Male dataset.

Below you can observe the evaluation difference between our previews model(part 1) and the new model.

Female dataset:

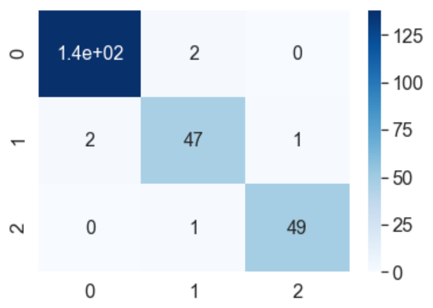


```
              precision    recall  f1-score   support

   with mask       0.99      0.86      0.92       140
without mask       0.69      0.96      0.80        50
not a person       0.98      0.94      0.96        50

    accuracy                           0.90       240
   macro avg       0.89      0.92      0.89       240
weighted avg       0.93      0.90      0.91       240
```

Figure 5: Old model(female dataset)



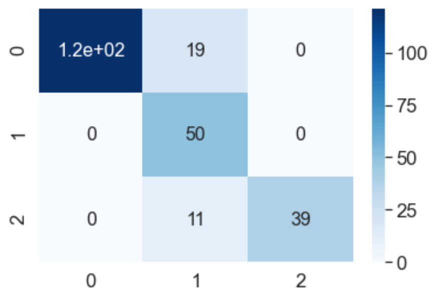
```
              precision    recall  f1-score   support

   with mask       0.99      0.99      0.99       140
without mask       0.94      0.94      0.94        50
not a person       0.98      0.98      0.98        50

    accuracy                           0.97       240
   macro avg       0.97      0.97      0.97       240
weighted avg       0.97      0.97      0.97       240
```

Figure 6: New model(female dataset)

Male dataset:

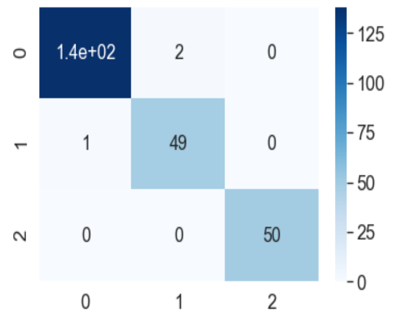


```
              precision    recall  f1-score   support

   with mask       1.00      0.86      0.93       140
without mask       0.62      1.00      0.77        50
not a person       1.00      0.78      0.88        50

    accuracy                           0.88       240
   macro avg       0.88      0.88      0.86       240
weighted avg       0.92      0.88      0.88       240
```

Figure 7: Old model(male dataset)



```
              precision    recall  f1-score   support

   with mask       0.99      0.99      0.99       140
without mask       0.96      0.98      0.97        50
not a person       1.00      1.00      1.00        50

    accuracy                           0.99       240
   macro avg       0.98      0.99      0.99       240
weighted avg       0.99      0.99      0.99       240
```

Figure 8: New model(male dataset)

k-fold cross-validation:

In this section we applied k-fold on the model (phase 1) with 10 folds on the training dataset, which splits into 9 train and 1 test fold on the training data and trains the model accordingly by providing the mean value of predictions as a final result. Based on this, the k-fold applied model improves predictions than usual learning applied on the model(phase 1). Thus it decreases the loss of the network and increases accuracy of the results.

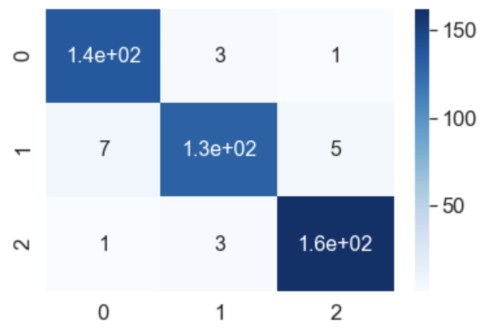
After training with K-fold with the same train dataset which was used for the default model(phase1, we noticed there's a decrease in prediction errors.

- In the model taken from phase errored predictions on test dataset are 20
- And after training with k-fold model the errored predictions on test dataset are 9

K Fold count no	**Agg accuracy(%)**	**Agg loss(%)**
1	89.31	0.43
2	99.481	0.04
3	99.67	0.0135
4	100	0.000656
5	100	8.47
6	100	0.00027
7	100	0.00015
8	100	4.5097
9	100	6.557
10	100	0.0016
	Overall Accuracy: 98%	

Accuracy of the model in phase 1 was 95 % and Now it changed to after applying k-fold 98%

Confusion matrix before applyng k-fold on the same test dataset above using model of phase 1

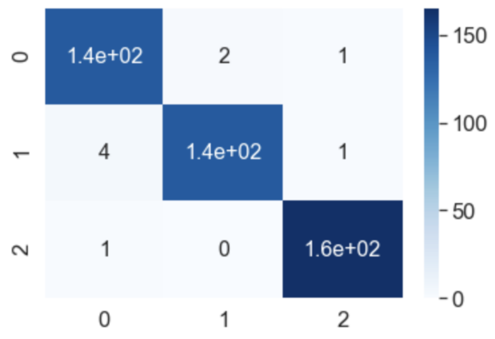


```
              precision    recall  f1-score   support

   with mask       0.94      0.97      0.96       141
without mask       0.96      0.92      0.94       143
not a person       0.96      0.98      0.97       166

    accuracy                           0.96       450
   macro avg       0.96      0.95      0.95       450
weighted avg       0.96      0.96      0.96       450
```

Confusion matrix after applying k-fold on the same test dataset



```
              precision    recall  f1-score   support

   with mask       0.97      0.98      0.97       141
without mask       0.99      0.97      0.98       143
not a person       0.99      0.99      0.99       166

    accuracy                           0.98       450
   macro avg       0.98      0.98      0.98       450
weighted avg       0.98      0.98      0.98       450
```

Reference

[1] https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset

[2] https://www.kaggle.com/prasunroy/natural-images

[3] https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

[4] https://pytorch.org/docs/stable/data.html\

[5]https://medium.com/@himadrisankarchatterjee/a-basic-introduction-to-convolutional-neural-network-8e39019b27c4