Project Assignment Part 1

COMP 472 – AA

For Dr. René Witte

8th June 2022

40075793 Abdul Qadir Ali

40058876 Abdul Rahman Mirza

40079665 Ahnaf Habib Khan

40010809 Mohamed Allalou

Team Composition:

Title	Name	Tasks
Data Specialist	Abdul Qadir Ali	Creating, pre-processing, loading & analyzing the datasets.
Training Specialist	Abdul Rahman Mirza	Setting up and training the CNN.
Evaluation Specialist	Ahnaf Habib khan	Analyzing, evaluating, and applying the generated model.
Compliance Specialist	Mohamed Allalou	Overall planning, scope verification, and compliance.

Dataset:

To build our dataset, we collected our images from Kaggle repositories that we have referenced[1][2][3]. We have 1600 images (which include 300 for training and 100 for testing for each class), with a total size of 256 Mb. We separated them into 4 classes, each having 2 subclasses. For example, 400 images are in the N95 class. From this, 300 are N95-training, and 100 are N95-test.

We separated into 8 folders, 4 for training having 300 images each and 4 for testing having 100 images for each type of class required.

For preprocessing, we resized all the images to 64x64 by leveraging the native function 'resize' within the transforms class in the torchvision library, and for normalization we use (0.5) as a mean parameter.

The way we imported the images into our model is by creating 8 arrays; 1 for each subclass, and then applied transform.compost functions on those arrays.

These are arrays are made up of tuples of image and its corresponding class label. For example,

```
dirsTraining = [
            (root + '/' + 'Train_NoMask',0),
            (root + '/' + 'Train_SurgicalMask', 1),
            (root + '/' + 'Train_ClothMask', 2),
            (root + '/' + 'Train_N95Mask', 3)
        ]
datasetTraining = MasksDataSet(dirsTraining, transform = transform)
```

Similarly, we did the same thing for testing.

CNN Architecture:

For our CNN we used NeutralNetClassifier function, and we used just 1 epoch. We chose our epoch as 1 because when we increased the number of epochs our precision decreased. In fact, with one epoch, our precision was 80%, with 2 epochs, it lowered to 70%, and this linear trend continued. This is because increasing the number of epochs caused overfitting, and thus what was effectively happening is our model was not **learning** the data, but rather it was **memorizing** the data. Thus, we stayed with 1 epoch as this maximized precision.

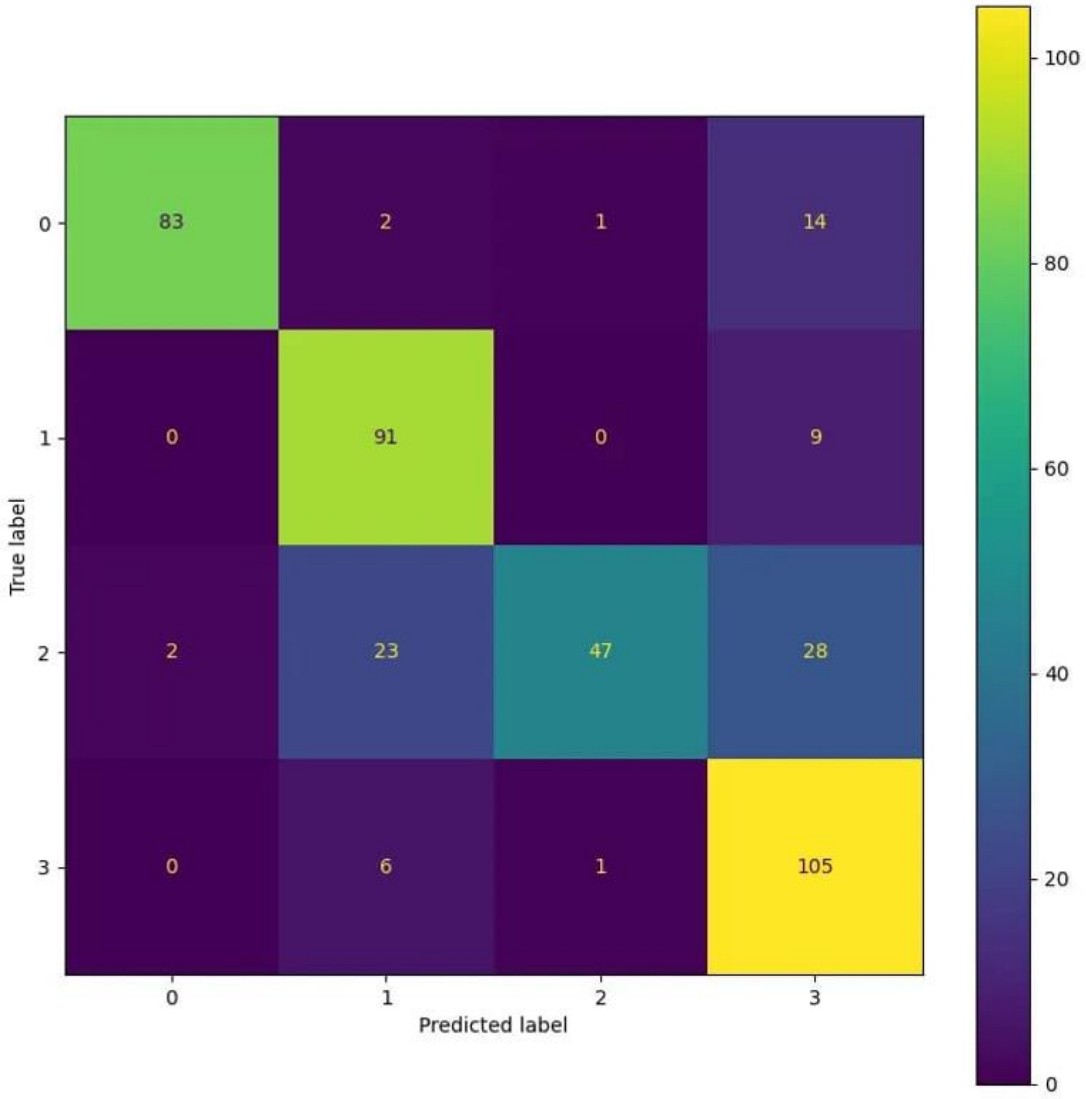
We kept the batch size 8. We trained our model using the NeutralNetClassifier function by using the **.fit** function supplied with datasetTraining and its corresponding label.

```
net = NeuralNetClassifier(

        CNN,

        max_epochs=1,

        iterator_train__num_workers=0,

        iterator_valid__num_workers=0,

        lr=1e-3,

        batch_size= batch_size,#8,

        optimizer=optim.Adam,

        criterion=nn.CrossEntropyLoss,

        device=DEVICE

    )



net.fit(datasetTraining, y=y_train)
```

After our model was trained we performed our predictions and evaluations.

Evaluation:

Confusion Matrix:



^ 0 is No Mask, 1 is Surgical Mask, 2 is Cloth Mask, and 3 is N95 Mask.

Precision, accuracy, recall, and F1 measure below:

```
Accuracy:     79.12621359223301
Recall:       79.12621359223301
Precision:    83.38354502172565
F1_Score:     78.29129953002672
```

Discussion:

As we can see, our model is successfully relatively precise. In the future, we would spend more time preparing our dataset. Perhaps we can analyze our algorithm and see what is going wrong that is causing increased epochs to lower our model's precision.

Regarding the data, we would spend a few hours scouring google images and downloading a variety of images. We can also automate the screengrabbing of frames from youtube videos. We would normalize them; ensure that they are relatively high resolution, and run them through a program such as Irfanview in order to make them all the same size without the distortion that is possibly caused by the torchvision library functions.

In truth, what we have found is that one's model is only as good as the data it is trained on. There is not much we can do to the architecture to improve a model that is trained on low quality data, and vice versa.

References:

All images were pulled from Kaggle repositories. Screenshots were taken from our own instance of the code running on our machines.

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

[2] - https://www.kaggle.com/datasets/coffee124/facemaskn95?select=000099.jpeg

[3] - https://www.kaggle.com/code/ayushimishra2809/face-mask-detection/data