COMP 6721 Applied Artificial Intelligence (Winter 2022)

Project Assignment, Part II

AI Face Mask Detector

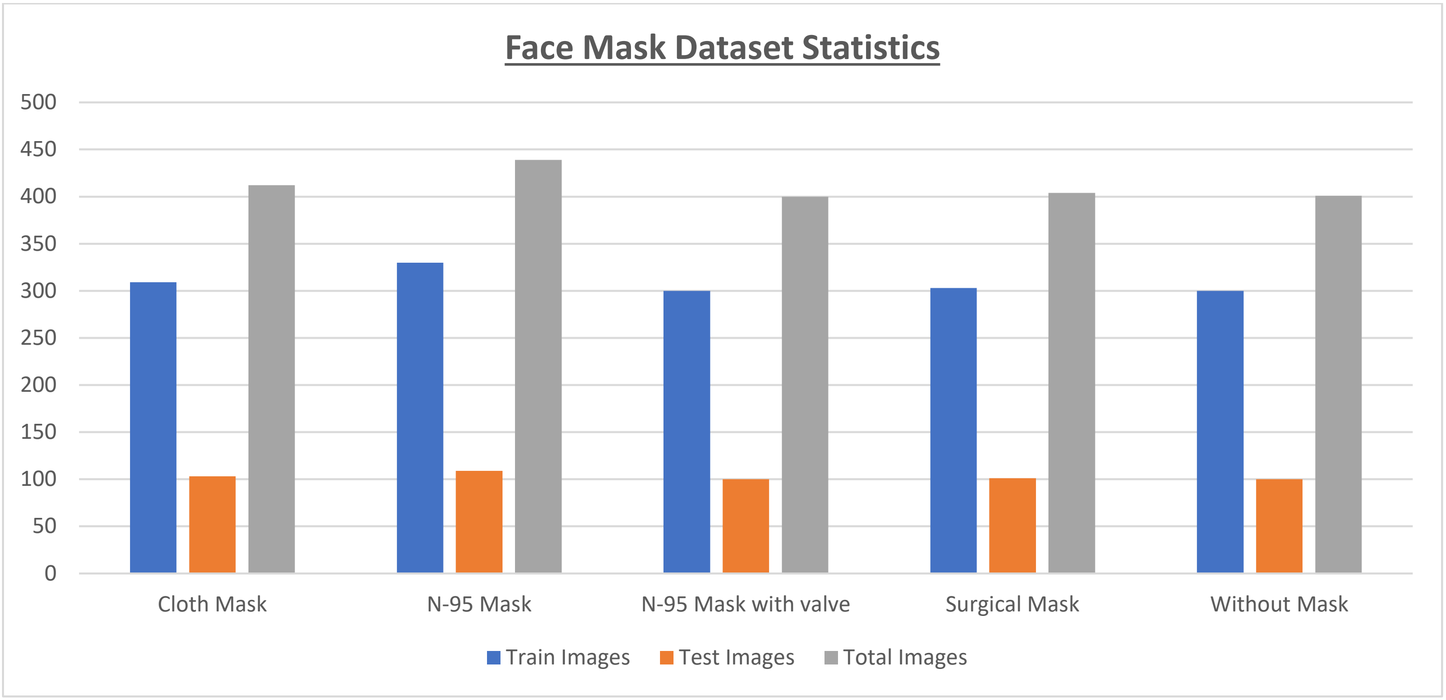
Group name: **NS_13**

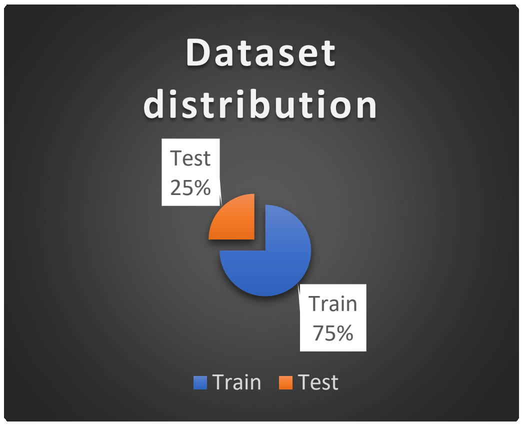
[1]

Role Assignments

Role	**Group Member**	**Student id**
Data Specialist	Shubham Bhanderi	40156448
Training Specialist	Parthiv Akbari	40155566
Evaluation Specialist	Jigar Borad	40155320

[1] Above images are taken from GitHub and Google Images.





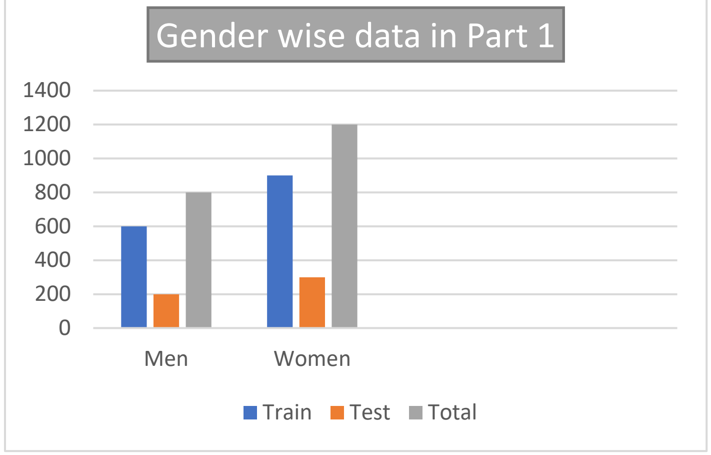
Dataset References:

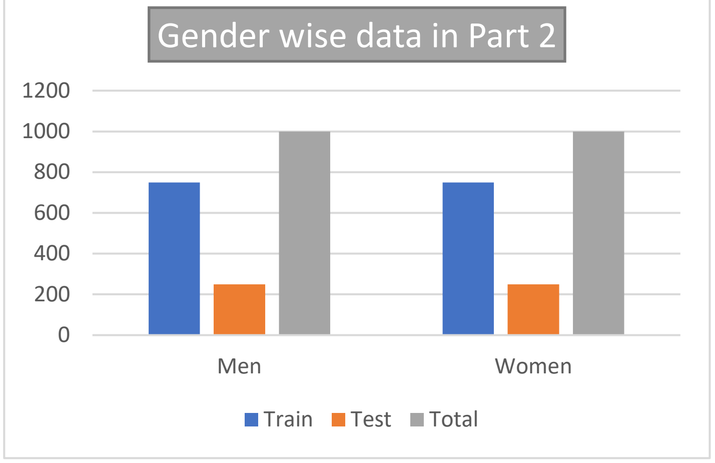
- Author, Larxel(2020) Face Mask Detection (version 1) from Kaggle
- Author, Omkar Gurav(2020) Face Mask Detection Dataset (version 1) from Kaggle
- Deb, C. (2022). Face Mask Detection (Version v1.0.0) [Computer software]. https://github.com/chandrikadeb7/Face-Mask-Detection
- Google Images
- https://humansintheloop.org/resources/datasets/
- GitHub
- Author, Prithwiraj Mitra (2020) COVID Face Mask Detection Dataset (Version 1) from Kaggle
- Author, Ashish Jangra (2020) Face Mask Detection ~12K Images Dataset (Version 1) from Kaggle

For the preprocessing of the data images, Pytorch transform function is used to resize all input images. Particularly in this project we transformed all images to 50 x 50 size[2]. For dividing the dataset randomly, Pytorch random split function is used to split randomly data from each class. As per diagram, it shows that data is split to the ratio of 75% train and 25% test images. There are total 2098 images across all class.

[2] https://towardsdatascience.com/how-to-train-an-image-classifier-in-pytorch-and-use-it-to-perform-basic-inference-on-single-images-99465a1e9bf5

Finding Bias in project



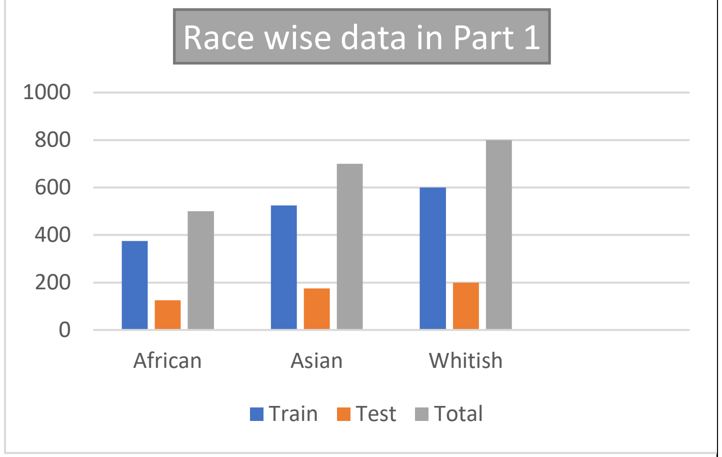


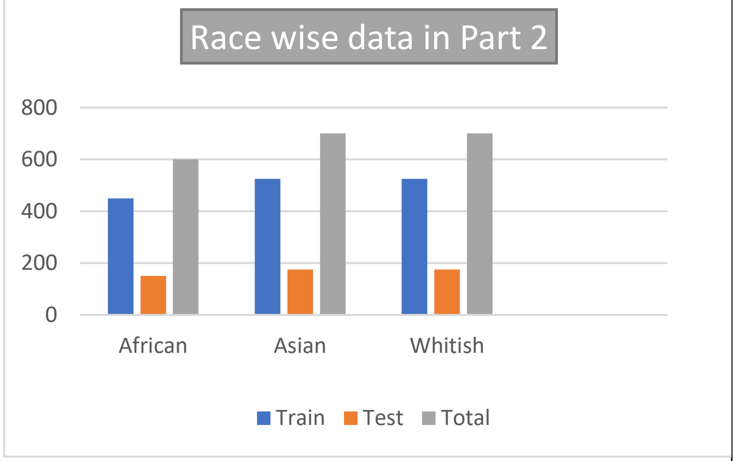
Gender wise performance in part 1

Gender	**Accuracy**
Male	66%
Female	63%

Gender wise performance in part 2

Gender	**Accuracy**
Male	69%
Female	70%





Race wise performance in part 1

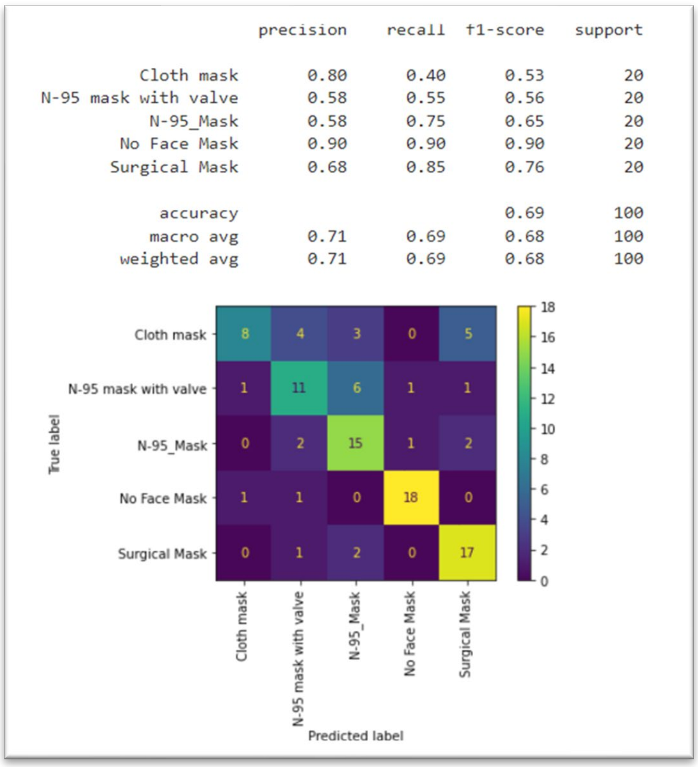
Race	**Accuracy**
African	52%
Asian	61%
Whitish	59%

Race wise performance in part 2

Race	**Accuracy**
African	67%
Asian	65%
Whitish	75%

Classification report and Confusion matrix for different sub-classes of part 2

Men

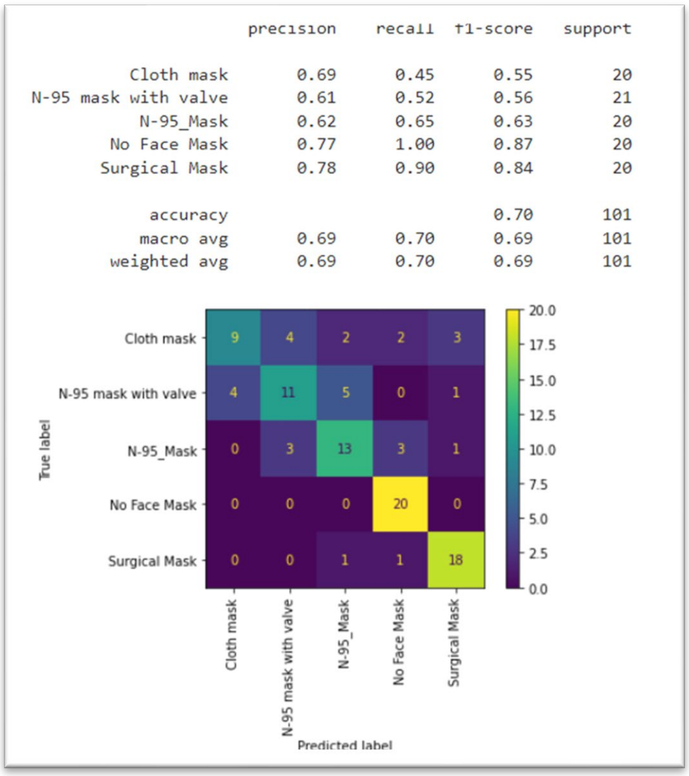


```
                    precision    recall  f1-score   support

         Cloth mask      0.80      0.40      0.53        20
N-95 mask with valve      0.58      0.55      0.56        20
           N-95_Mask      0.58      0.75      0.65        20
        No Face Mask      0.90      0.90      0.90        20
        Surgical Mask      0.68      0.85      0.76        20

            accuracy                          0.69       100
           macro avg      0.71      0.69      0.68       100
        weighted avg      0.71      0.69      0.68       100
```

Women



```
                    precision    recall  f1-score   support

         Cloth mask      0.69      0.45      0.55        20
N-95 mask with valve      0.61      0.52      0.56        21
           N-95_Mask      0.62      0.65      0.63        20
        No Face Mask      0.77      1.00      0.87        20
        Surgical Mask      0.78      0.90      0.84        20

            accuracy                          0.70       101
           macro avg      0.69      0.70      0.69       101
        weighted avg      0.69      0.70      0.69       101
```

In this project, our task is to find out bias in model. we analyzed our model with 2 following categories. The first is Bias with gender and race.

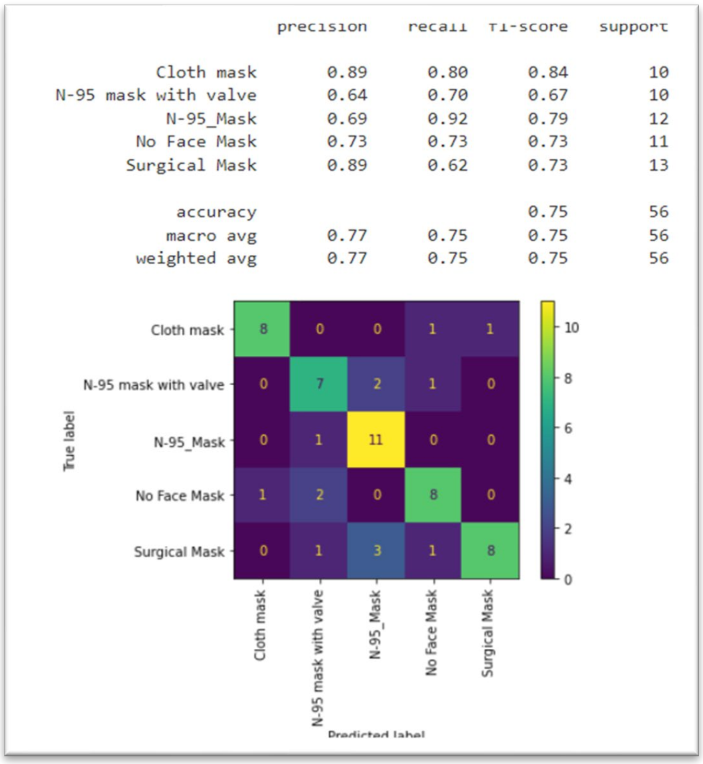
First of all, as you can see from the diagram and images, we figured out the bias between men and women images from the previously created database. Then we balanced the images for both classes to reduce the bias and improve on accuracy. After this process the accuracy recorded are 69% for men and 70% for women which previously was 66% and 63% respectively.

Secondly, as you can see from the images, we worked on mainly 3 different types of races. They are African, Asian, and white people. Part 1 model is used to evaluate the accuracy of test data of different race. But we found big difference in all of 3 accuracy and so we added some more clear photos across all races and remove some noisy data from dataset to increase accuracy of the model. At the end, In part 2, after doing filtering of dataset, we achieve 67%, 65% and 75 accuracies for African, Asian, and White people respectively.

For the different race, you can see following diagram of classification report and confusion matrix.[3]

[3] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.ConfusionMatrixDisplay.html

African People



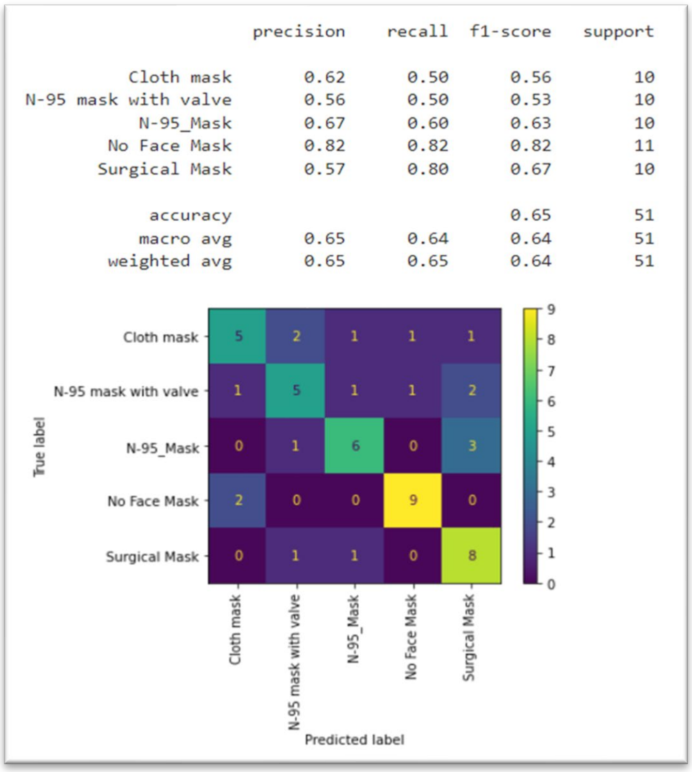
```
                     precision    recall  f1-score   support

          Cloth mask       0.89      0.80      0.84        10
N-95 mask with valve       0.64      0.70      0.67        10
            N-95_Mask       0.69      0.92      0.79        12
         No Face Mask       0.73      0.73      0.73        11
        Surgical Mask       0.89      0.62      0.73        13

             accuracy                          0.75        56
            macro avg       0.77      0.75      0.75        56
         weighted avg       0.77      0.75      0.75        56
```

True label \ Predicted label	Cloth mask	N-95 mask with valve	N-95_Mask	No Face Mask	Surgical Mask
Cloth mask	8	0	0	1	1
N-95 mask with valve	0	7	2	1	0
N-95_Mask	0	1	11	0	0
No Face Mask	1	2	0	8	0
Surgical Mask	0	1	3	1	8

Asian People



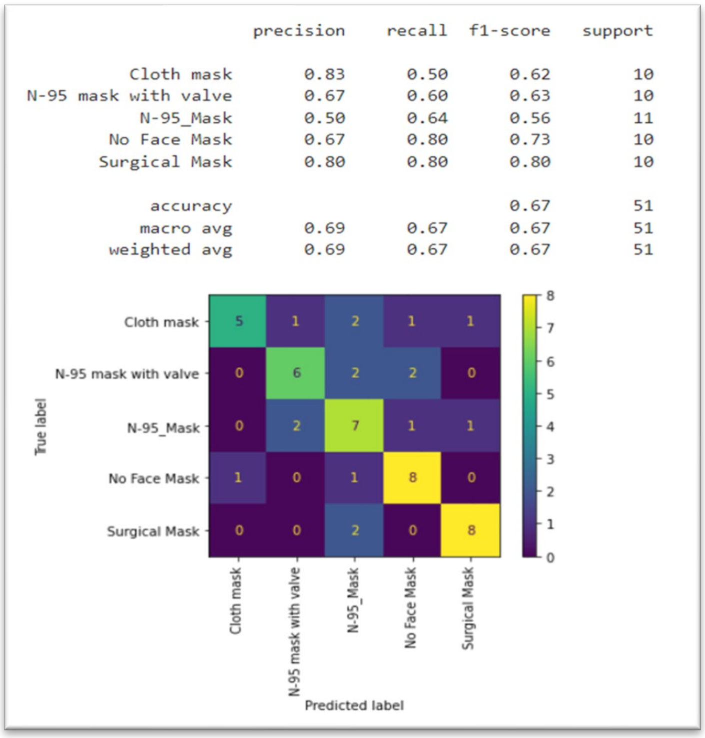
```
                     precision    recall  f1-score   support

          Cloth mask       0.62      0.50      0.56        10
N-95 mask with valve       0.56      0.50      0.53        10
            N-95_Mask       0.67      0.60      0.63        10
         No Face Mask       0.82      0.82      0.82        11
        Surgical Mask       0.57      0.80      0.67        10

             accuracy                          0.65        51
            macro avg       0.65      0.64      0.64        51
         weighted avg       0.65      0.65      0.64        51
```

True label \ Predicted label	Cloth mask	N-95 mask with valve	N-95_Mask	No Face Mask	Surgical Mask
Cloth mask	5	2	1	1	1
N-95 mask with valve	1	5	1	1	2
N-95_Mask	0	1	6	0	3
No Face Mask	2	0	0	9	0
Surgical Mask	0	1	1	0	8

White People



```
                     precision    recall  f1-score   support

          Cloth mask       0.83      0.50      0.62        10
N-95 mask with valve       0.67      0.60      0.63        10
            N-95_Mask       0.50      0.64      0.56        11
         No Face Mask       0.67      0.80      0.73        10
        Surgical Mask       0.80      0.80      0.80        10

             accuracy                          0.67        51
            macro avg       0.69      0.67      0.67        51
         weighted avg       0.69      0.67      0.67        51
```

True label \ Predicted label	Cloth mask	N-95 mask with valve	N-95_Mask	No Face Mask	Surgical Mask
Cloth mask	5	1	2	1	1
N-95 mask with valve	0	6	2	2	0
N-95_Mask	0	2	7	1	1
No Face Mask	1	0	1	8	0
Surgical Mask	0	0	2	0	8

K Fold cross Validation result analysis

- **Part 1 model results**

Fold	Precision	Recall	F1-measure	Accuracy
1	0.60	0.64	0.61	0.63
2	0.64	0.64	0.61	0.64
3	0.72	0.65	0.65	0.65
4	0.64	0.65	0.64	0.65
5	0.68	0.69	0.69	0.69
6	0.66	0.68	0.66	0.67
7	0.63	0.64	0.62	0.64
8	0.66	0.64	0.69	0.64
9	0.66	0.67	0.65	0.67
10	0.67	0.66	0.65	0.67

- **Part 2 model results**

Fold	Precision	Recall	F1-measure	Accuracy
1	0.70	0.68	0.69	0.70
2	0.61	0.61	0.59	0.63
3	0.63	0.59	0.57	0.61
4	0.57	0.55	0.55	0.56
5	0.64	0.63	0.62	0.63
6	0.69	0.69	0.66	0.72
7	0.64	0.64	0.63	0.63
8	0.63	0.63	0.59	0.65
9	0.62	0.62	0.62	0.63
10	0.62	0.63	0.60	0.63

K-folds technique is a popular and easy to understand it generally results in a less biased model compared to other methods because it ensures that every observation from the original data set has the chance of appearing in the training and test set this is one among the best approach if we have limited input data.

CNN Architecture

Layers:

```
(0): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(1): ReLU(inplace=True)
(2): MaxPool2d(kernel_size=3, stride=3, padding=0, dilation=1, ceil_mode=False)
(3): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(4): ReLU(inplace=True)
(5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(6): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(7): ReLU(inplace=True)
(8): MaxPool2d(kernel_size=3, stride=3, padding=0, dilation=1, ceil_mode=False)
(9): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(10): ReLU(inplace=True)
(11): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(12): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(13): ReLU(inplace=True)
(14): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
(15): Flatten(start_dim=1, end_dim=-1)
(16): Linear(in_features=256, out_features=1024, bias=True)
(17): Linear(in_features=1024, out_features=64, bias=True)
(18): Linear(in_features=64, out_features=5, bias=True)
(19): Softmax(dim=1)
```

Diagram[4]:

Layer (Type)	Input shape	Output shape
Conv2d_1 (Conv2D)	(-1,3,96,96)	(-1,16,96,96)
ReLU – Activation	(-1,16,96,96)	(-1,16,96,96)
MaxPool2D	(-1,16,96,96)	(-1,16,32,32)
Conv2d_2 (Conv2D)	(-1,16,32,32)	(-1,32,32,32)
ReLU – Activation	(-1,32,32,32)	(-1,32,32,32)
MaxPool2D	(-1,32,32,32)	(-1,32,16,16)
Conv2d_3 (Conv2D)	(-1,32,16,16)	(-1,64,16,16)
ReLU – Activation	(-1,64,16,16)	(-1,64,16,16)
MaxPool2D	(-1,64,16,16)	(-1,64,5,5)
Conv2d_3 (Conv2D)	(-1,64,5,5)	(-1,128,5,5)
ReLU – Activation	(-1,128,5,5)	(-1,128,5,5)
MaxPool2D	(-1,128,5,5)	(-1,128,2,2)

[4] https://github.com/pytorch/pytorch/issues/2001

Conv2d_3 (Conv2D)	(-1,128,2,2)	(-1,256,2,2)
ReLU – Activation	(-1,256,2,2)	(-1,256,2,2)
MaxPool2D	(-1,256,2,2)	(-1,256,1,1)
Flatten	(-1,256,1,1)	(-1,256)
Linear_1	(-1,256)	(-1,1024)
Linear_2	(-1,1024)	(-1,64)
Linear_3	(-1,64)	(-1,5)
Softmax (dim=1)	(-1,5)	(-1,5)

Table 1: CNN network workflow

*Format of input and output tensor is (batch size, channel , image height , image width)

Above table shows the output the generates from each layer after performing operations.

The above diagram of network layers shows how layers are implemented in CNN[5] network along with activation functions. Specifically, in this project, three convolutional layers and one linear layer are used. Diagram also refers to the input feature and output feature dimensions, when and how it changes while going to the next layer. ReLU activation function is used between each preceding layer to normalize the negative values. Maxpool2D is also used to down sample the image based on kernel size, padding and stride value. The equation for maxpool2D is as follows:[6]

$$\textbf{Convolution Output dimension} = [(I - F + 2 * P) / S] + 1 \times D$$

Ex: in 1st conv2d layer we have dimension, 100x12x50x50 so image dimension is 50x50. **I=50.** Kernel size in maxpooling2d is **F = 3**. Padding is **P= 0**. Stride is **S = 2**. D means output features of the previous layer that is **D = 12**.

Apply all values to equation, [(50-3 + 2*0)/2] + 1 x 12 = **12 x 24 x 24**

More importantly, when input image(tensor) goes from Convolution layer to Linear layer, we need to flatten the input tensor value. Hence, we multiply input channel with image width and heigh which you can see in table 1.

Ex: in above table before linear layer tensor is **(100,48,5,5).** After applying flatten, it converts input tensor to **(100,48*5*5) = (100,1200).**

[5] https://www.analyticsvidhya.com/blog/2019/10/building-image-classification-models-cnn-pytorch/
https://pyimagesearch.com/2021/07/19/pytorch-training-your-first-convolutional-neural-network-cnn/
https://medium.com/thecyphy/train-cnn-model-with-pytorch-21dafb918f48
https://medium.com/thecyphy/train-cnn-model-with-pytorch-21dafb918f48
[6] https://kvirajdatt.medium.com/calculating-output-dimensions-in-a-cnn-for-convolution-and-pooling-layers-with-keras-682960c73870

Training of CNN Model:[7]

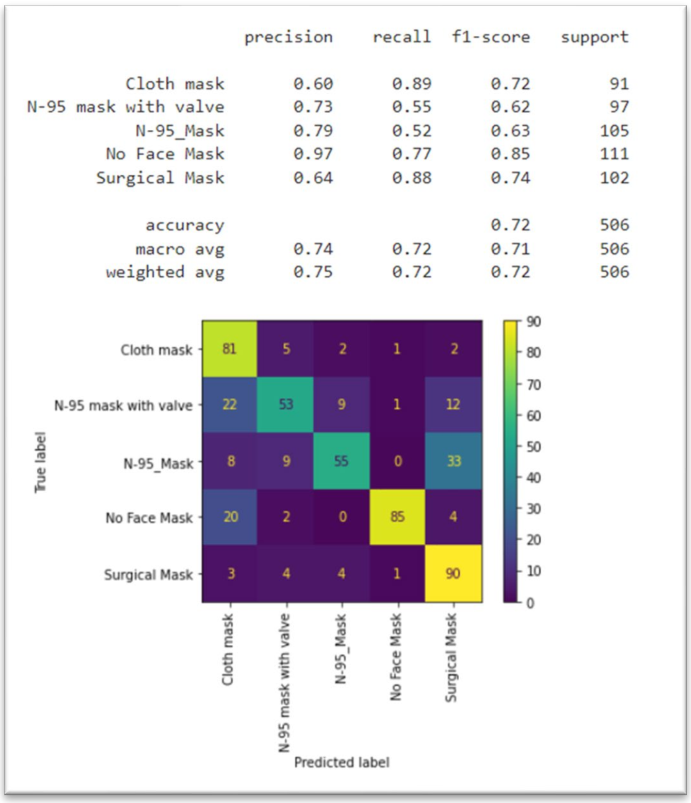
Training the CNN model basically consist two parts:

1. A forward pass, in which the input image throughout the neural network layers.
2. A backward pass, in which gradients are backpropagated and weights and bias are updated.

In model, we are processing **20 epochs** to train the model**. 75%** of dataset is used to train the model and rest of **25%** data is used to test the model. Adam optimizer is used to update the gradients value while training the model. All the tensor values are uploaded to single device that is CPU. Learning rate is set to 0.0005. CrossEntropyLoss function is used to calculate the loss during the training and used that value in backpropagation of the network. Things to consider while training model, if the gap between training and testing accuracy widens with subsequent epochs that means model is overfitting. To address this issue, we will add dropout layer in our network. So, we force layers to learn from other parameters too. One more thing, a large batch size reduces the time to train the model, but batch size should not be extremely large.

Evaluation:

For the evolution part of the project, classification report is calculated for test data which includes precision, recall and f1 score for each class separately. Report also includes the how many numbers of images are in each class for testing the model. Accuracy of the model is also shown in the classification report.



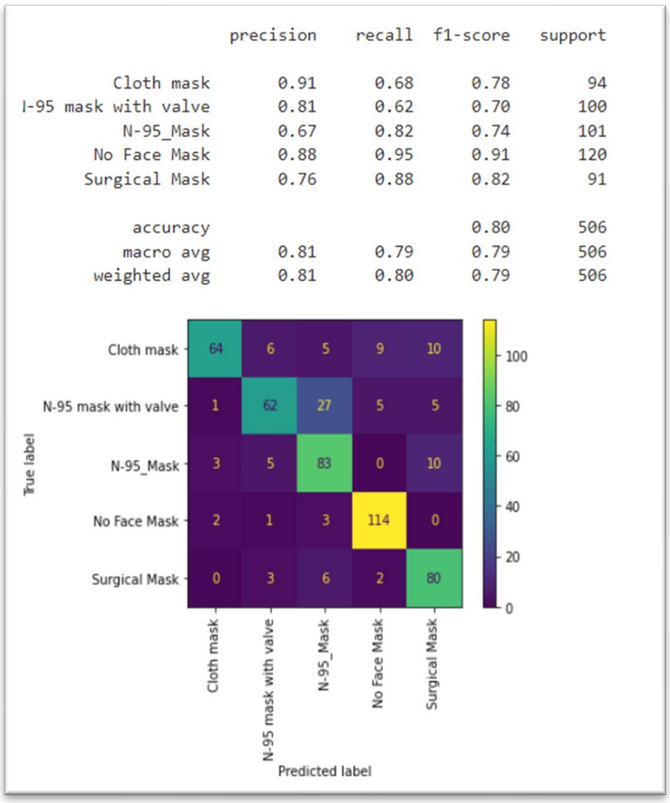
```
                     precision    recall  f1-score   support

         Cloth mask       0.60      0.89      0.72        91
N-95 mask with valve       0.73      0.55      0.62        97
          N-95_Mask       0.79      0.52      0.63       105
       No Face Mask       0.97      0.77      0.85       111
      Surgical Mask       0.64      0.88      0.74       102

           accuracy                           0.72       506
          macro avg       0.74      0.72      0.71       506
       weighted avg       0.75      0.72      0.72       506
```

	Cloth mask	N-95 mask with valve	N-95_Mask	No Face Mask	Surgical Mask
Cloth mask	81	5	2	1	2
N-95 mask with valve	22	53	9	1	12
N-95_Mask	8	9	55	0	33
No Face Mask	20	2	0	85	4
Surgical Mask	3	4	4	1	90

Figure 2: Test data evaluation part 1[8]



```
                     precision    recall  f1-score   support

         Cloth mask       0.91      0.68      0.78        94
I-95 mask with valve       0.81      0.62      0.70       100
          N-95_Mask       0.67      0.82      0.74       101
       No Face Mask       0.88      0.95      0.91       120
      Surgical Mask       0.76      0.88      0.82        91

           accuracy                           0.80       506
          macro avg       0.81      0.79      0.79       506
       weighted avg       0.81      0.80      0.79       506
```

	Cloth mask	N-95 mask with valve	N-95_Mask	No Face Mask	Surgical Mask
Cloth mask	64	6	5	9	10
N-95 mask with valve	1	62	27	5	5
N-95_Mask	3	5	83	0	10
No Face Mask	2	1	3	114	0
Surgical Mask	0	3	6	2	80

Figure 3: Test data evaluation part 2

[7] https://towardsdatascience.com/training-a-convolutional-neural-network-from-scratch-2235c2a25754

As per above 2 figures, we can evaluate that in part 2, we achieve more accuracy compared to part 1 and also, we have included 10 K-Fold cross validation results in previous pages. Then we have selected best model among those 10 models.

In part 2, specifically, we tried to remove bias from AI model and try to achieve more accuracy across all categories such as Gender, Race. we modify our dataset by adding some more photos of all different classes to rebalance the dataset so we can achieve reliable performance for various categories of data.

[8] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html