COMP 6721

Applied Artificial Intelligence

Project Report (Phase-2)

Submitted to: Prof. Dr. René Witte

Team Id: NS-05

Gaurang Dobariya (40184707) – Evaluation Specialist
Vickykumar Patel (40185238) – Training Specialist
Parth Navsariwala (40178800) – Data Specialist

Table of Content

Table of Images

1. Dataset:

Collection of data called Dataset. In AI Based Face Mask Detection our main aim is to classify whether person wearing a certain type of mask which fall in one of five category or not person wearing no mask at all.

For this project, we need to collect data for **5 different categories** namely Person "**without a mask**", Person with a "**cloth mask**", Person with a "**Surgical mask**" , Person with a "**N95 mask**" and Person with "**N95 mask with valve**".

Cloth Mask

N95 Mask

N95 with Valve Mask

Surgical Mask

Without Mask

1.1 General Examples of Five Categories of Mask

The most difficult part of collecting enough data is ensuring that all training and testing photos for each class only contain data for that class. The majority of public available dataset contain two datasets, dataset where people wearing mask and where people not wearing mask.

However, a closer examination reveals that the dataset containing person waring mask includes a variety of face masks, including N95, N95 with valve, Cloth Mask, Surgical Mask and so on are examples. As a result, spend a significant amount of time gathering the relevant information and We used both publicly available data and data that we created ourselves Another crucial point to remember is that the photographs should only show one face, hence we didn't offer any training or testing images with many faces. Only a single face of a person wearing a (particular type of) mask or not must be portrayed in the image. Finally, we discovered that the background has a significant impact on our model's performance. To put it another way, photos with particular objects or environments in the background had a negative impact on our model training. As a result, we had to ensure that background noise was eliminated, which we accomplished by cropping our photographs such that only the face (along with some minor background components) is visible.

To build **our data set** we collect all images from **mainly 3 sources:**

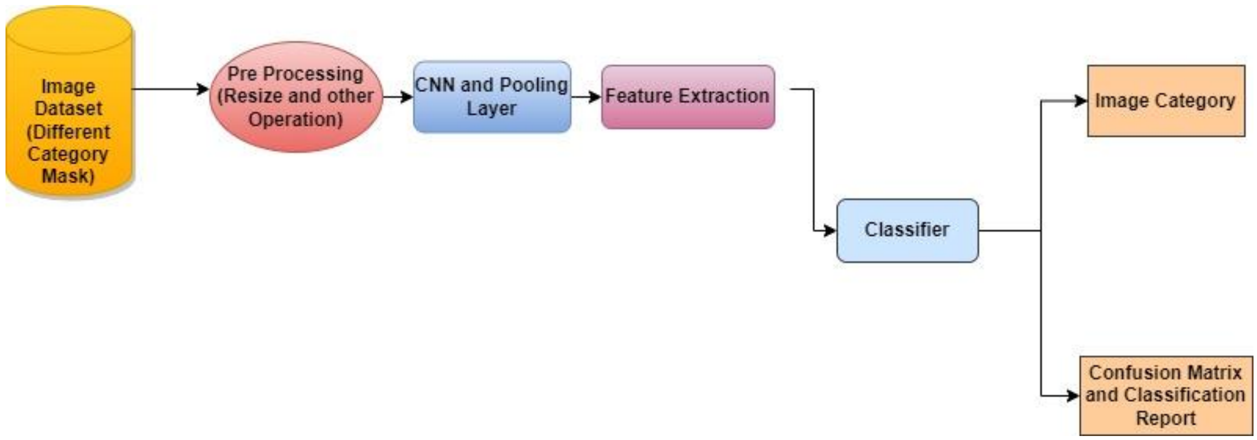
a. **Kaggle Dataset**
   There are many available datasets for mask detection on Kaggle we refer all the dataset and then we filter all dataset one by one and download images as per our categories.
b. **Google Images**
   To create our own data set we research on google using different key words which are as follow: **'Person wearing N95 face mask,' 'Person wearing cloth face mask', 'silk cloth face mask,' or 'cotton cloth face mask', 'Person without face mask' , 'Person wearing N95 face mask with valve'.** We've already begun planning the second phase of our project.

Size of each category or label in the dataset

Cloth Mask	401
N95 Mask	404
N95 Mask with Valve	401
Surgical Mask	647
Without Mask	999

So, total **we have 2852 Images** that divide in different categories as per above mentioned table.

Training Set	75%	2139
Testing Set	25%	713
Total	100%	2852



1.2 Project Overview

2. CNN Architecture:

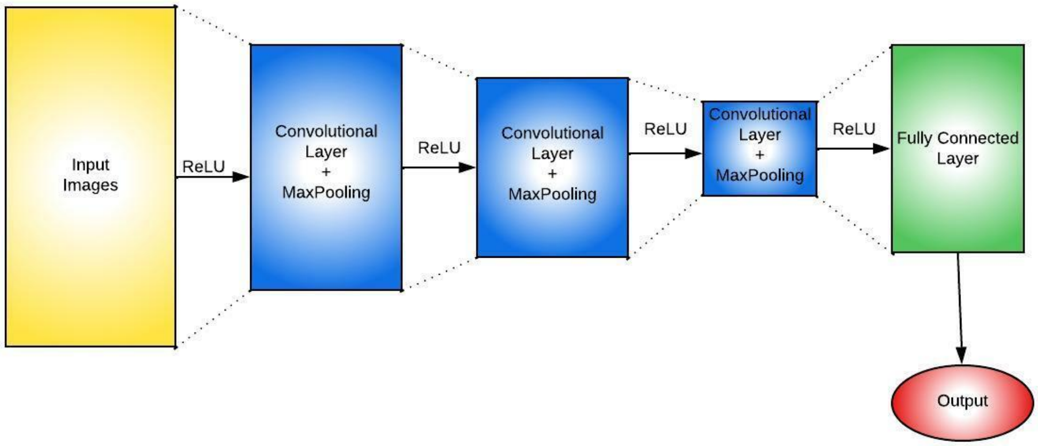
Convolutional Neural Network (CNN) are special kind of multi-layer neural networks, designed to extract key features from an image like visual patterns with very low pre-processing of data.

We used CNN to train our model for specific categories. The dimensions of all the photographs we had were not in the right size because we had gathered data from various sources. Some were too big, while others were too small. We had to apply some **pre-processing** to make all of these **photographs look the same.**

We transformed all of the photos in the dataset to the **same size of (224 * 224)** while loading them. We also modify the pixel intensity by normalizing the data using standard deviation and image means. We also shuffled the data before passing it on to the training phase to add some randomization.

There are mainly **3 layers** in our CNN model which are as follow:

a. **Convolutional layers:** Convolutional layers can be used to extract specific features from the image. Each convolutional layer learns specific features of image.
b. **Pooling layers:** These layers are used after convolutional layers to reduce the size of feature map generated by the convolutional layer so that the model can generalize on other data apart from training data and also to reduce the complexity of computation.
c. **Fully connected layers:** These layers use the features extracted by the convolutional layers and use the extracted features for decision making.



2.1 CNN Model

To extract features from the dataset, we used a very simple CNN architecture with only 3 convolutional layers. First, an **image of size (224 * 224)** is passed to the first convolution 2D layer with **kernel size (3 * 3)** and stride and **padding equal to 1**. The output channel for this layer is 64. The data is then normalized and passed as input to the **ReLU activation function**. From here, the output of the ReLU activation function serves as an input to MaxPooling with **stride 2 and kernel size (2 * 2).** This output is passed to the next convolutional 2D layer. Here, the input channel is 64, and all other parameters are the same as for the first convolution 2D layer. The next step is the same as before. That is, normalization, the **ReLU** activation function, and MaxPooling. The output is then input to a third convolutional 2D layer with 128 input channels and 256 output channels. Again, all steps such as normalization, ReLU activation function, MaxPooling, etc. are performed. **Adam, a gradient-based stochastic optimization, is used in this project.**

Then, we used a fully connected dense layer that layers use the features extracted by the convolutional layers and use the extracted features for decision making.

```
Epoch: 1, training loss: 2.1886074542999268, training accuracy: 0.5961486101150513
Epoch: 2, training loss: 0.70000159740448, training accuracy: 0.7761102318763733
Epoch: 3, training loss: 0.4456046521663666, training accuracy: 0.8399122953414917
Epoch: 4, training loss: 0.37804123759269714, training accuracy: 0.8693119883537292
Epoch: 5, training loss: 0.28087034821510315, training accuracy: 0.902617871761322
Epoch: 6, training loss: 0.18871432542800903, training accuracy: 0.9351699352264404
Epoch: 7, training loss: 0.13879568874835968, training accuracy: 0.9539930820465088
Epoch: 8, training loss: 0.15275199711322784, training accuracy: 0.95280522108078
Epoch: 9, training loss: 0.11402993649244308, training accuracy: 0.9613715410232544
Epoch: 10, training loss: 0.08977413177490234, training accuracy: 0.9717881679534912
```

2.2 Epoch Output

```python
# Customized convolution neural network class which is again inherited from the torch
# neural network class. this class contains all the filters and other operations
# which are performed on images. The class contains forward method which is used to
# feedforward the neural network and the backpropogation.
class COMP_6721_CNN(nn.Module):

    def __init__(self):
        super(COMP_6721_CNN, self).__init__()
        self.cnn_layers = nn.Sequential(
            # convolution layer 1
            nn.Conv2d(3, 64, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),

            # convolution layer 2
            nn.Conv2d(64, 128, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),

            # convolution layer 3
            nn.Conv2d(128, 256, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(256),
            nn.ReLU(inplace=True),
            nn.MaxPool2d(kernel_size=2, stride=2),
        )

        self.linear_layers = nn.Sequential(
            nn.Linear(200704, 5)
        )

    # forward pass to readjust weights
    def forward(self, x):
        x = self.cnn_layers(x)
        x = x.view(x.size(0), -1)
#         print(x.size())
        x = self.linear_layers(x)
        return x
```

2.3 CNN Code

3. Bias Detection and Elimination:

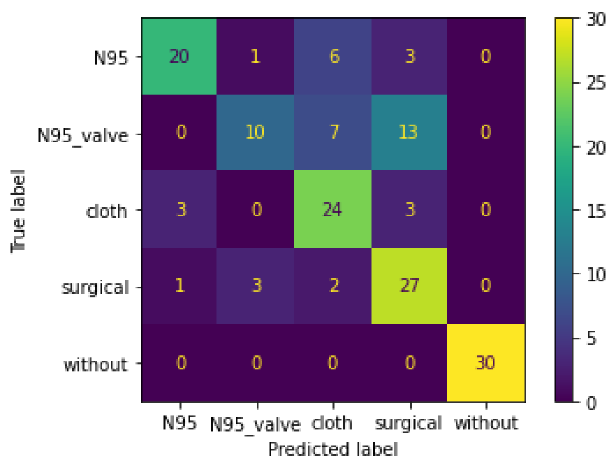
a phenomenon that occurs when an AI algorithm produces results that are systemically prejudiced due to erroneous assumptions in the machine learning process. There are several examples of age biased and gender biased in real life application. For this project our model was **biased towards female** when we predicted each gender-based test data and in Age based testing model is **bias towards young group**. To handle this situation we start cleaning our dataset and remove some irrelevant and blur images. Also we add good images in our dataset and after training our bias was minimized. Below mentioned confusion metrics and classification report shows the result of before and after eliminating bias of gender and age.

- **Gender Bias:**
  For gender biased testing we took two subgroups: **Male and Female** and we took around 400 images in total and mask classification.

➢ **Female Bias:**
  For the female biased testing we almost used 153 Images and after generating report we came to know that our model has a problem in predicting N95 mask and N95 valve mask images.

	precision	recall	f1-score	support
N95	0.83	0.67	0.74	30
N95_valve	0.71	0.33	0.45	30
cloth	0.62	0.80	0.70	30
surgical	0.59	0.82	0.68	33
without	1.00	1.00	1.00	30
accuracy			0.73	153
macro avg	0.75	0.72	0.71	153
weighted avg	0.75	0.73	0.71	153

3.1 Classification Report Female Based before Bias

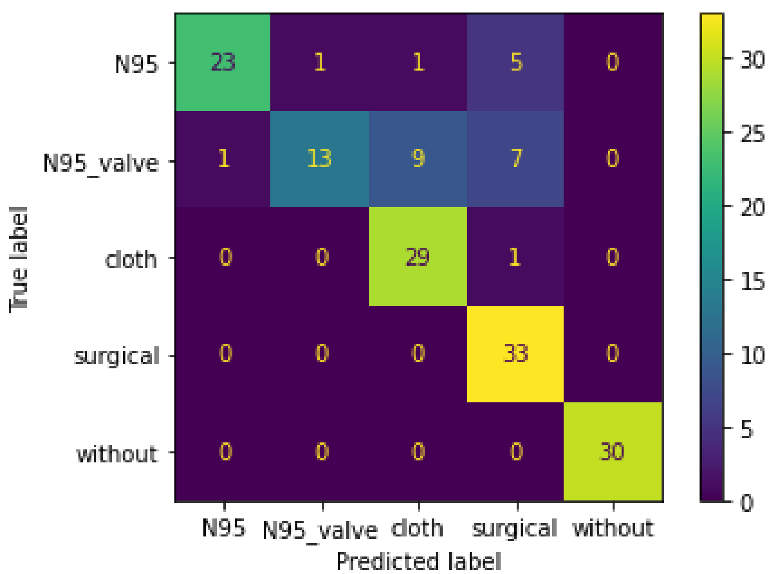


3.2 Confusion Matrix Female Based before Bias

For eliminating bias we added some clear and appropriate images of N95 and valve mask in our dataset for female and retrained our model and then we get some good accuracy that is around 84% and also bias is eliminate.

	precision	recall	f1-score	support
N95	0.96	0.77	0.85	30
N95_valve	0.93	0.43	0.59	30
cloth	0.74	0.97	0.84	30
surgical	0.72	1.00	0.84	33
without	1.00	1.00	1.00	30
accuracy			0.84	153
macro avg	0.87	0.83	0.82	153
weighted avg	0.87	0.84	0.82	153

3.3 Classification Report Female Based after Bias



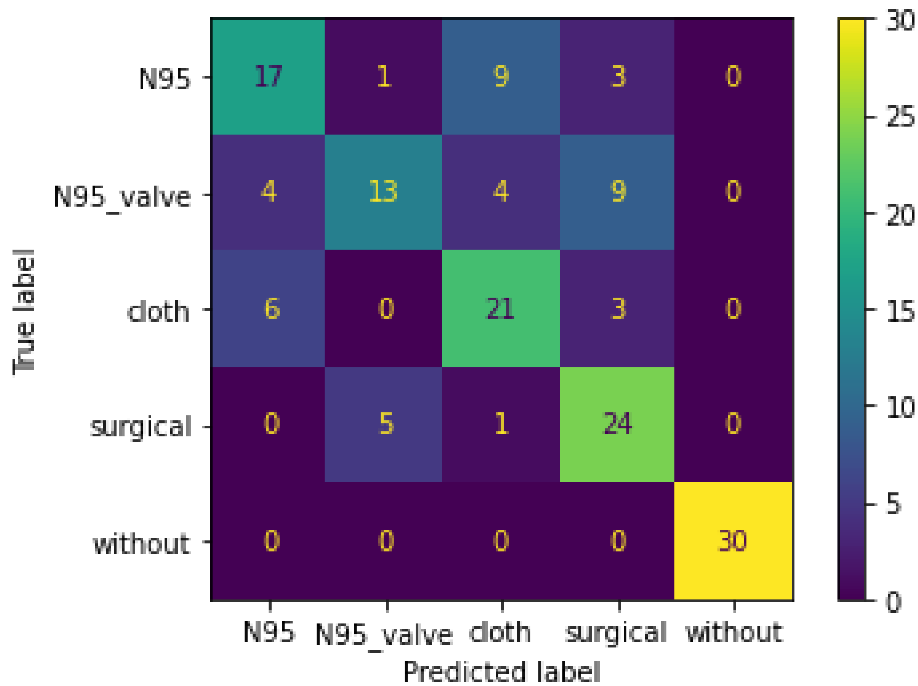
3.4 Confusion Matrix Female Based after Bias

- **Male Bias:**

  For the male biased testing we almost used 150 Images and after generating report we came to know that our model has a problem in predicting N95 and valve mask images.

	precision	recall	f1-score	support
N95	0.63	0.57	0.60	30
N95_valve	0.68	0.43	0.53	30
cloth	0.60	0.70	0.65	30
surgical	0.62	0.80	0.70	30
without	1.00	1.00	1.00	30
accuracy			0.70	150
macro avg	0.71	0.70	0.69	150
weighted avg	0.71	0.70	0.69	150

3.5 Classification Report Male Based before Bias

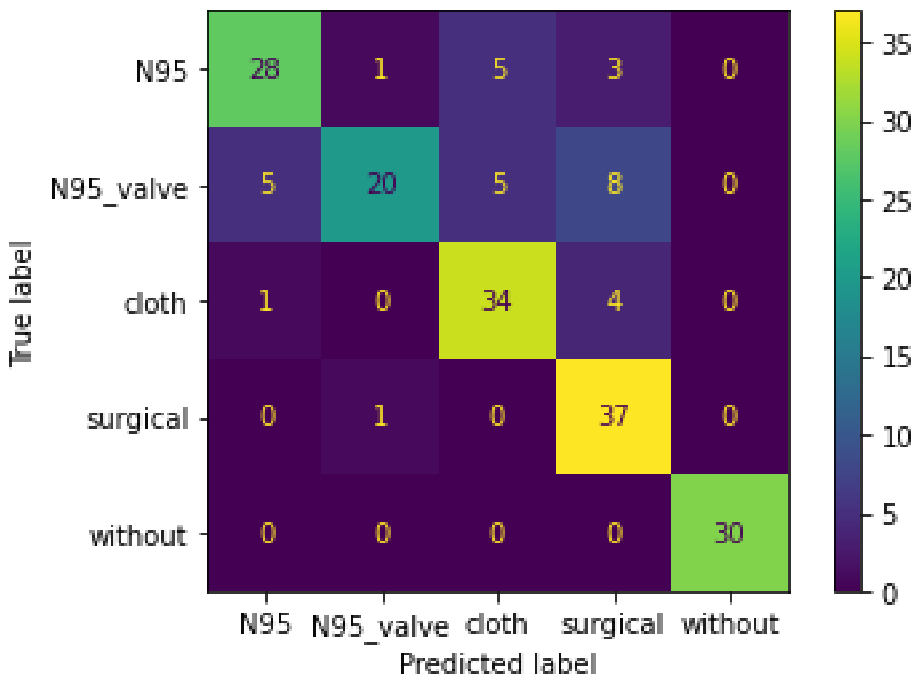


3.6 Classification Report Male Based before Bias

For eliminating bias we added some clear and appropriate images of N95 and valve mask in our dataset for female and retrained our model and then we get some good accuracy that is around 82% and also bias is eliminate.

	precision	recall	f1-score	support
N95	0.82	0.76	0.79	37
N95_valve	0.91	0.53	0.67	38
cloth	0.77	0.87	0.82	39
surgical	0.71	0.97	0.82	38
without	1.00	1.00	1.00	30
accuracy			0.82	182
macro avg	0.84	0.83	0.82	182
weighted avg	0.84	0.82	0.81	182

3.7 Classification Report Male Based after Bias



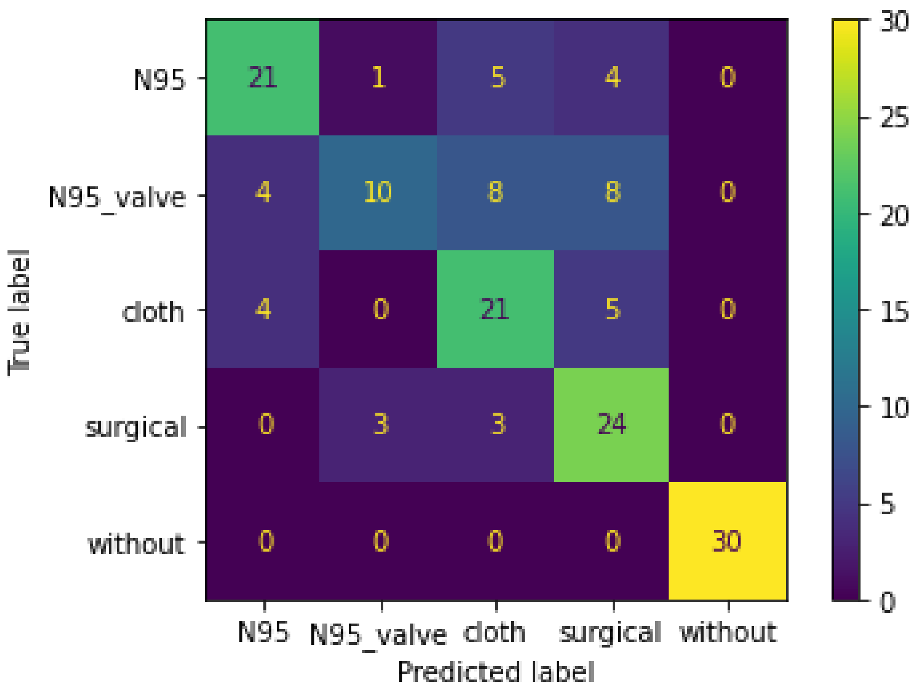
3.8 Classification Report Male Based after Bias

- **Age Bias:**
  For gender biased testing we took two subgroups: **Young and Old** and we took around 400 images in total and mask classification.

➢ **Young Bias:**
  For the young, biased testing we almost used 151 Images and after generating report we came to know that our model has a problem in predicting N95 mask and N95 valve mask images.

	precision	recall	f1-score	support
N95	0.72	0.68	0.70	31
N95_valve	0.71	0.33	0.45	30
cloth	0.57	0.70	0.63	30
surgical	0.59	0.80	0.68	30
without	1.00	1.00	1.00	30
accuracy			0.70	151
macro avg	0.72	0.70	0.69	151
weighted avg	0.72	0.70	0.69	151

3.9 Classification Report Young Based before Bias

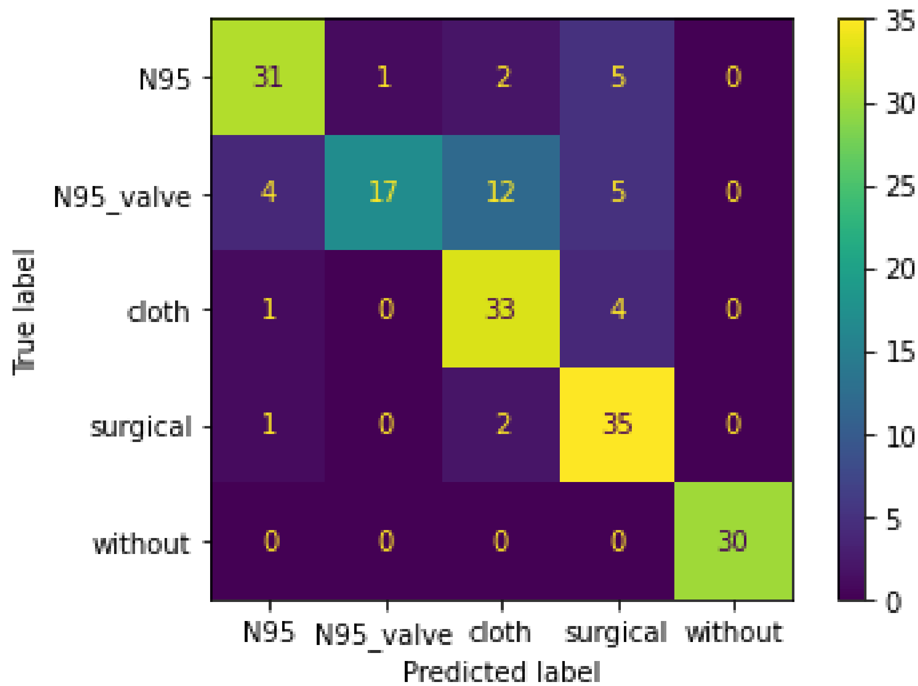


3.10 Confusion Matrix Young Based before Bias

For eliminating bias we added some clear and appropriate images of N95 and valve mask in our dataset for young and retrained our model and then we get some good accuracy that is around 80% and also bias is eliminate.

	precision	recall	f1-score	support
N95	0.84	0.79	0.82	39
N95_valve	0.94	0.45	0.61	38
cloth	0.67	0.87	0.76	38
surgical	0.71	0.92	0.80	38
without	1.00	1.00	1.00	30
accuracy			0.80	183
macro avg	0.83	0.81	0.80	183
weighted avg	0.83	0.80	0.79	183

3.11 Classification Report Young Based after Bias

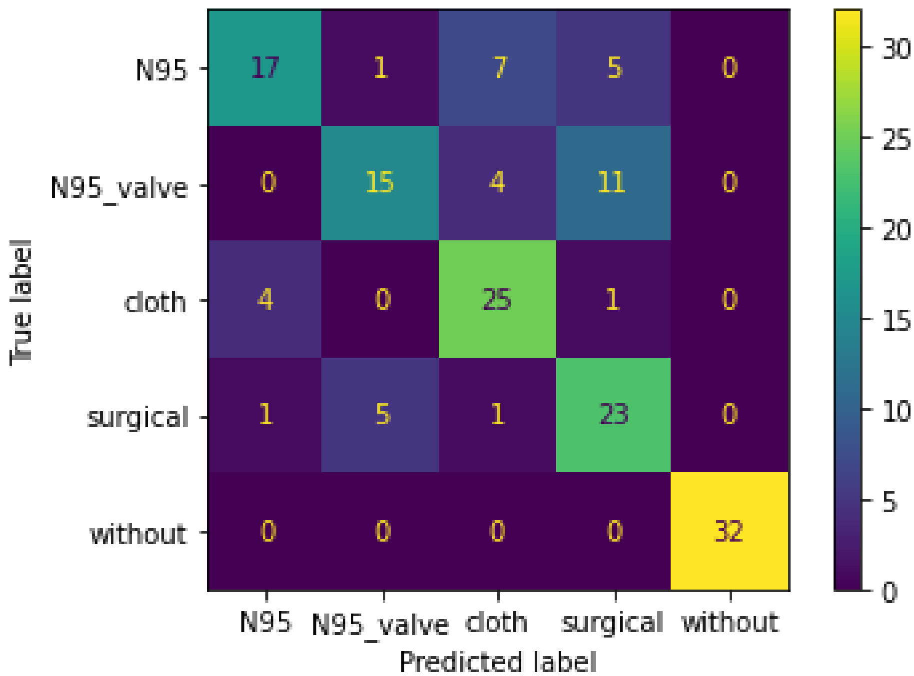


3.12 Confusion Matrix Young Based after Bias

- **Old Bias:**
  For the old, biased testing we almost used 152 Images and after generating report we came to know that our model has a problem in predicting N95 mask and N95 valve mask images.

	precision	recall	f1-score	support
N95	0.77	0.57	0.65	30
N95_valve	0.71	0.50	0.59	30
cloth	0.68	0.83	0.75	30
surgical	0.57	0.77	0.66	30
without	1.00	1.00	1.00	32
accuracy			0.74	152
macro avg	0.75	0.73	0.73	152
weighted avg	0.75	0.74	0.73	152

3.13 Classification Report Old Based before Bias

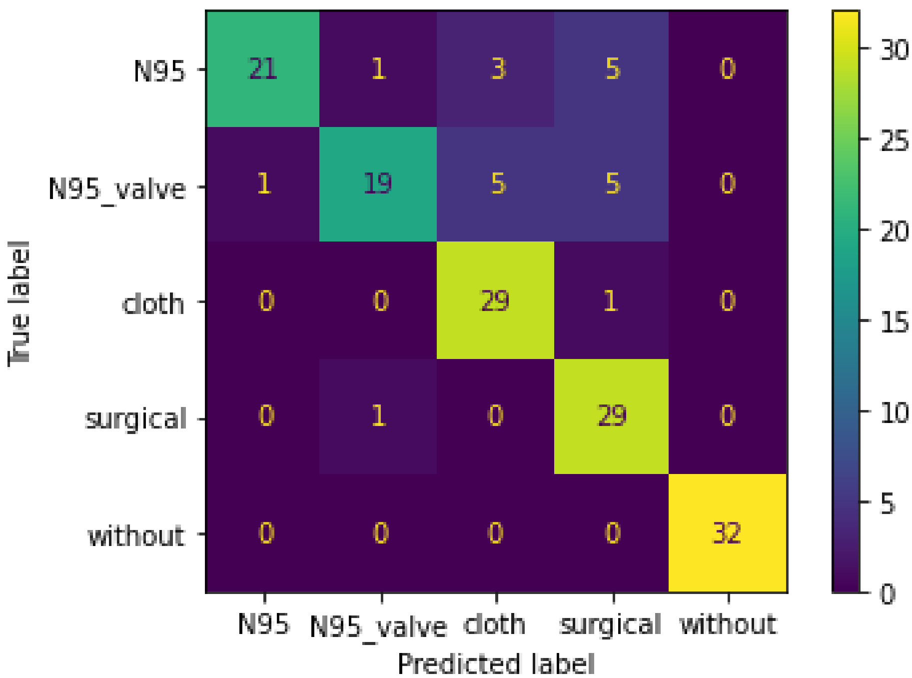


3.14 Confusion Matrix Old Based before Bias

For eliminating bias we added some clear and appropriate images of N95 and valve mask in our dataset for young and retrained our model and then we get some good accuracy that is around 86% and also bias is eliminate.

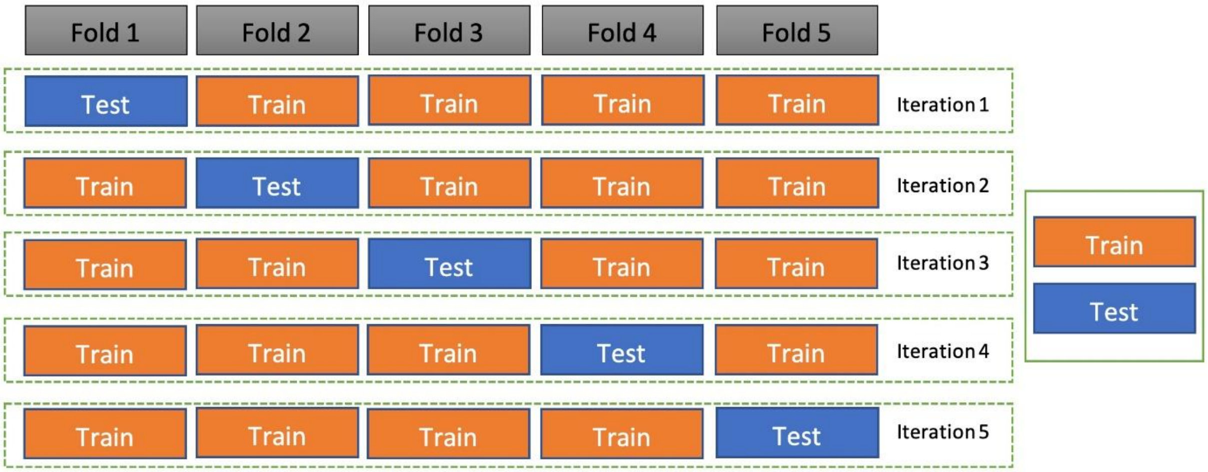
	precision	recall	f1-score	support
N95	0.95	0.70	0.81	30
N95_valve	0.90	0.63	0.75	30
cloth	0.78	0.97	0.87	30
surgical	0.72	0.97	0.83	30
without	1.00	1.00	1.00	32
accuracy			0.86	152
macro avg	0.87	0.85	0.85	152
weighted avg	0.88	0.86	0.85	152

3.15 Classification Report Old Based after Bias



3.16 Confusion Matrix Old Based after Bias

4 K-Fold Cross-Validation:



4.1 K-fold Cross-Validation

K-fold cross-validation is the most common technique for model evaluation and model selection in machine learning. The main idea behind K-Fold cross-validation is that each sample in our dataset has the opportunity of being tested. In K-Fold, we iterate over a dataset K times. In each round, we split dataset into K parts: one part is used for testing, and the remaining K-1 parts are merged into a training subset for model evaluation as shown in the figure above, which illustrates the process of 5-fold cross-validation:

The final score of the k fold is usually average of the all k-fold.

The advantages of k fold cross validation are as follows:

1. Computation time is reduced as we repeated the process only 10 times when the value of k is 10.
2. Reduced Bias
3. The variance of the resulting estimate is reduced as value of k increases

In our project we use 10 folds function so each using SKlearn library and each time dataset is divided into 9 parts for training and 1^{st} part for testing. In first fold 1^{st} part for testing and other 9^{th} part used for training. This process is continuing until all parts used for testing.

The main benefit of using this k-fold is that our model gets more variety of training data as well as testing data. This ultimately results in more accurate predictions for unknown data.

The other benefits of using K fold validation is it reduced the chances of overfitting compare to train test split because In train test split every time we use same data for train model over all epochs and in this approach there might be a chance to get result in overfitting and In k fold each time we use different images for training and testing.

- **Results of each individual fold, as well as the aggregate statistics, for precision, recall, F1, and accuracy:**

Folds	Precision	Recall	F-1 Score	Accuracy
Fold 1	91	92	91	93
Fold 2	97	97	97	98
Fold 3	98	98	98	98
Fold 4	98	98	98	99
Fold 5	100	100	100	100
Fold 6	100	100	100	100
Fold 7	100	99	99	100
Fold 8	99	99	99	99
Fold 9	99	99	99	99
Fold 10	100	100	100	100
Average	0.98	0.98	0.98	0.99

4.2 10-Fold Cross Validation on Old Dataset

Folds	Precision	Recall	F-1 Score	Accuracy
Fold 1	0.91	0.90	0.89	0.90
Fold 2	0.98	0.99	0.99	0.99
Fold 3	0.97	0.94	0.95	0.96
Fold 4	0.98	0.98	0.98	0.99
Fold 5	0.99	0.99	0.99	0.99
Fold 6	0.99	0.99	0.99	0.99
Fold 7	0.99	0.99	0.99	0.99
Fold 8	0.99	0.98	0.98	0.99
Fold 9	0.98	0.98	0.98	0.99
Fold 10	0.99	0.98	0.99	0.99
Average	0.97	0.97	0.98	0.98

4.3 10-Fold Cross Validation on Updated Dataset

The main difference between two data set is removal of underlying bias with respect to gender and age. From the above mentioned table we can say that the average precision, recall and F-1Score does not greatly between old and new dataset.

5 Evaluation:

a. Part - I

After training the model, we need to check whether the model was trained well and for that we can evaluating various factors like accuracy, precision and f1score and support. **The model is trained for 10 epochs** and in each epoch, the **learning rate was 0.001**. We have calculated all this with factors with the help of **classification_report()** method. At last, we build the confusion matrix using **confusion_matrix()** method. The confusion matrix of the model can be useful to know the precision which in our case shows that many testing set images were misclassified due to imbalanced data. The confusion matrix of the model is useful for knowing the accuracy. In this case, **the training dataset is 94% accurate and the test dataset is 96% accurate.**

We also randomly fetched **100 different category images** and tried to predict those categories by a trained model. The file name (image type) is displayed on the left side, and the expected category is displayed on the right side.

Training Classification Report:

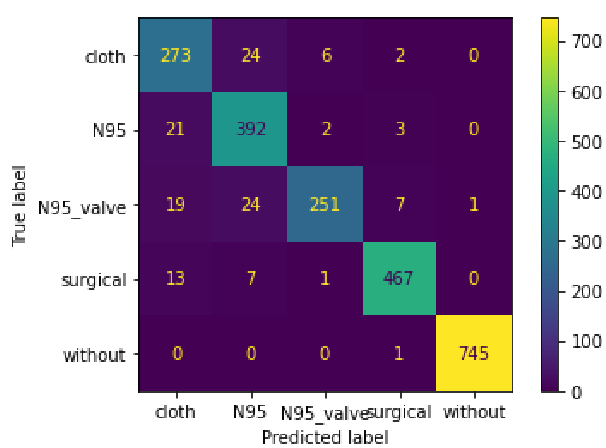
```
---- Loading saved Model ----
---- Generating Classification Report ----
Training Classification Report
              precision    recall  f1-score   support

           0       0.84      0.90      0.87       305
           1       0.88      0.94      0.91       418
           2       0.97      0.83      0.89       302
           3       0.97      0.96      0.96       488
           4       1.00      1.00      1.00       746

    accuracy                           0.94      2259
   macro avg       0.93      0.92      0.93      2259
weighted avg       0.94      0.94      0.94      2259
```

5.1 Training Classification Report

Training Confusion Matrix:



5.2 Training Classification Report

Testing Classification Report:

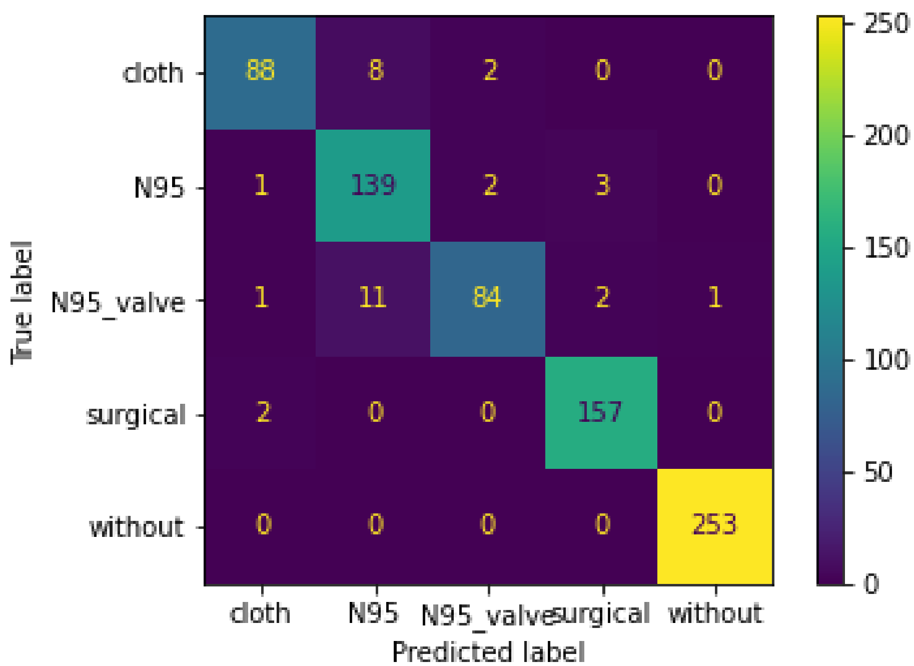
```
---- Generating Classification Report ----
Testing Classification Report
              precision    recall  f1-score   support

           0       0.96      0.90      0.93        98
           1       0.88      0.96      0.92       145
           2       0.95      0.85      0.90        99
           3       0.97      0.99      0.98       159
           4       1.00      1.00      1.00       253

    accuracy                           0.96       754
   macro avg       0.95      0.94      0.94       754
weighted avg       0.96      0.96      0.96       754
```

5.3 Testing Classification Report

Testing Confusion Matrix:



5.4 Testing Classification Report

b. Part - II

In the second part of project, we used k-fold cross validation to enhance model performance and analyzed the possible bias in the project part 1. Also, we eliminated all the reasons making our model to misclassify the data from the build 1.

We trained our model with 10-fold cross validation.

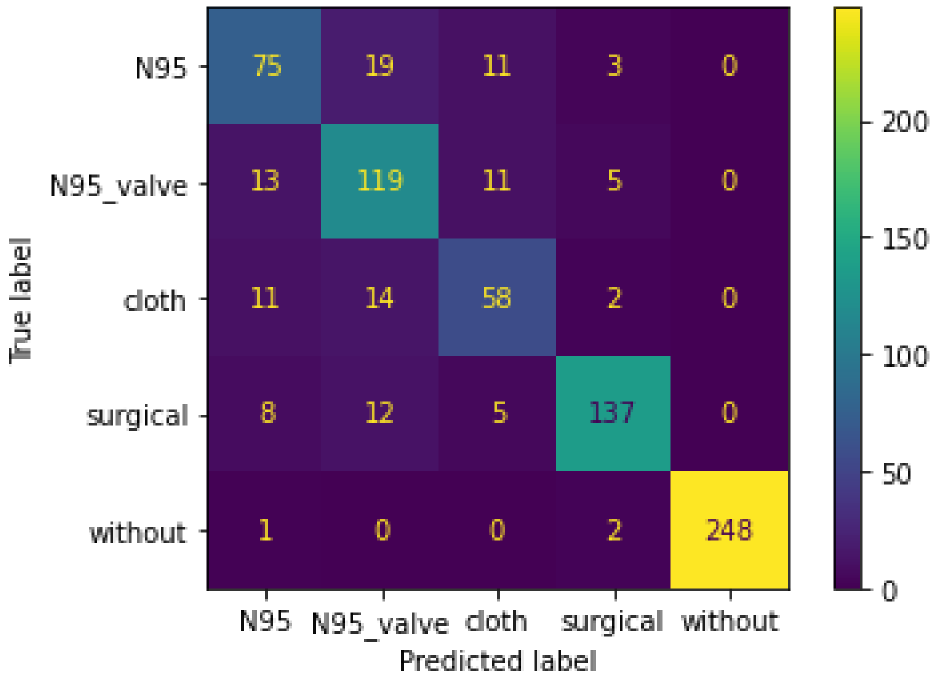
```
Accuracy : 1.0
Running Fold Num:  10
---- Training Model ----
Epoch: 1, training loss: 0.01881873607635498, training accuracy: 0.99609375
Epoch: 2, training loss: 0.07563132047653198, training accuracy: 0.9781249761581421
Epoch: 3, training loss: 0.19925455749034882, training accuracy: 0.9502962827682495
Epoch: 4, training loss: 0.385394811630249, training accuracy: 0.9112472534179688
Epoch: 5, training loss: 0.17672185599803925, training accuracy: 0.9546874761581421
```

As above mentioned, classification report we can clear see that our accuracy for training is **95%** and testing accuracy is **91%** which is more compared to previous train test split result.

	precision	recall	f1-score	support
N95	0.69	0.69	0.69	108
N95_valve	0.73	0.80	0.76	148
cloth	0.68	0.68	0.68	85
surgical	0.92	0.85	0.88	162
without	1.00	0.99	0.99	251
accuracy			0.84	754
macro avg	0.80	0.80	0.80	754
weighted avg	0.85	0.84	0.85	754

---- Confusion Matrix ----

5.5 K-Fold Before Bias Testing Classification Report



5.6 K-Fold Before Bias Testing Confusion Matrix

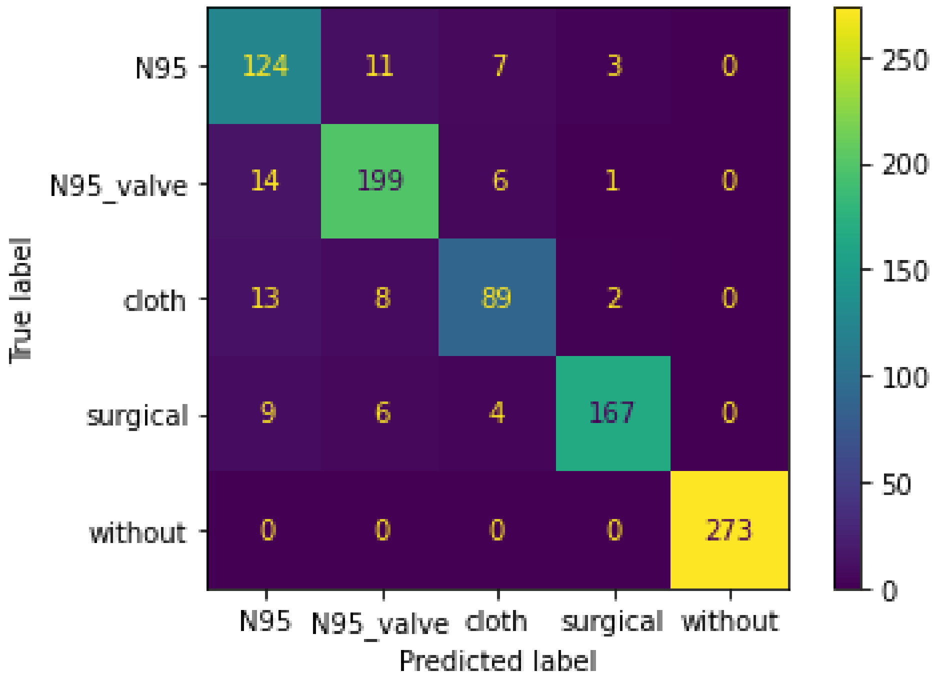
```
---- Confusion Matrix ----
---- Generating Classification Report ----
Testing Classification Report
              precision    recall  f1-score   support

         N95       0.78      0.86      0.81       145
   N95_valve       0.89      0.90      0.90       220
       cloth       0.84      0.79      0.82       112
    surgical       0.97      0.90      0.93       186
     without       1.00      1.00      1.00       273

    accuracy                           0.91       936
   macro avg       0.89      0.89      0.89       936
weighted avg       0.91      0.91      0.91       936

---- Confusion Matrix ----
```

5.7 K-Fold After Bias Testing Classification Report



5.8 K-Fold After Bias Testing Confusion Matrix

6 References:

1. Kaggle Dataset
2. CNN Model
3. Different Type of CNN Architecture
4. Without Mask
5. Surgical Mask
6. Surgical Mask
7. N95 Valve Mask
8. Cloth Mask
9. Cloth Mask
10. Cloth Mask Men
11. N95 Mask
12. N95 Mask
13. Confusion Matrix
14. Confusion Matrix Display
15. Neural Network Model
16. Image Transformation
17. Bias
18. K-Fold Cross-Validation
19. K-Fold