Concordia University

COMP 6721 - Applied Artificial Intelligence (Winter 2022)

Project Assignment [Part I] - Report

Team Members

1. Elvin Rejimone (40193868) - Data Specialist, responsible for creating, pre-processing, loading & analyzing the datasets.
2. Santhosh Santhanam(40202625) - Training Specialist, responsible for setting up and training CNN.
3. Parth Ashokkumar Parekh (40161500) - Evaluation, responsible for analyzing, evaluating, and applying the generated model.

Table of Contents:

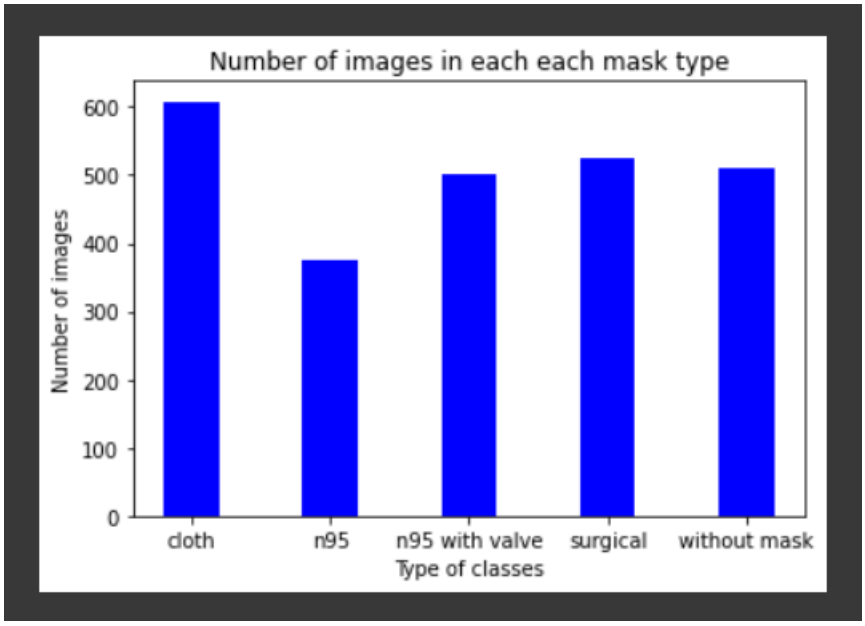
Dataset

The dataset consists of a collection of images from various sources for the following categories of masks:

1. Person **without** a Face mask
2. Person with a **Cloth** Mask
3. Person with a **Surgical** mask
4. Person with an **N95[FFP2/N95/KN95]** Mask
5. Person with an **N95[FFP2/N95/KN95] Mask with Valve**

In total we have collected **2519** images across the above-mentioned categories for testing and training. We divided the dataset with an 80:20 split for training and testing. Hence, we have **2015** images for training across all categories and **504** images for testing across all categories. The images have been normalized and resized to 32*32*3.

[Mask Dataset Link]




```
cloth: 608
n95: 375
n95 with valve: 501
surgical: 525
without mask: 510
```

Architecture

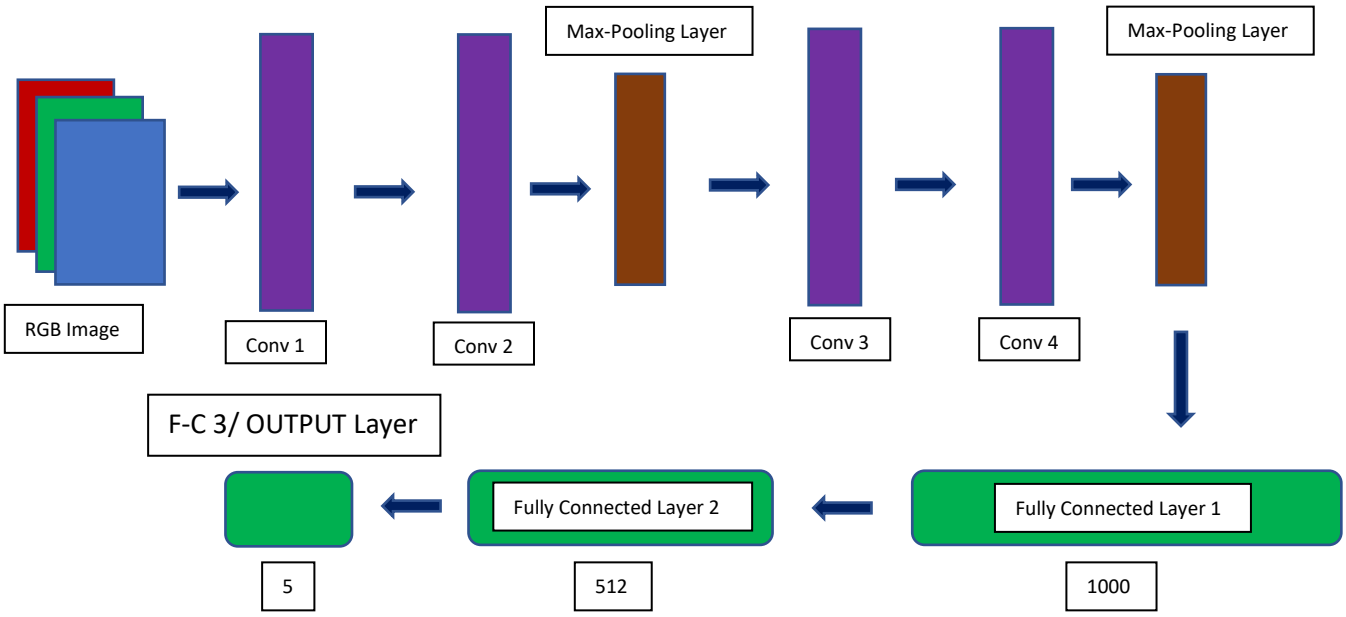
Layers:

We have implemented 4 types of layers in our CNN Model:

1. Convolutional Layers
2. Batch Normalization Layers
3. Pooling Layers
4. Fully Connected Layers

Model:

To classify the image into the five categories (No Mask, Surgical Mask, Cloth Mask, N95 Mask with and N95 with valve), we trained a model which consists of 4 Convolutional layers, Batch Normalization layers, 2 Max Pooling layers, 3 fully connected layers. The model is designed in such a way that after every 2 convolutional layers there is one Max Pooling layer with a filter of size 2 and stride of 2. After each max pooling layer, the number of features are halved. The size of the filter in each of the Convolutional layers is 3 with the stride of 1 and no padding. The activation function for all the layers is the Rectified linear unit (ReLU). After convolutions, the features are flattened and they are fed to the 3 fully connected layers of size 4096, which is followed by another fully connected layer of size 1000 and 512 after that. This 512 units fully connected layer is then connected to the final output layer of 5 units. Refer to the following image for the architecture.



CNN Model Architecture

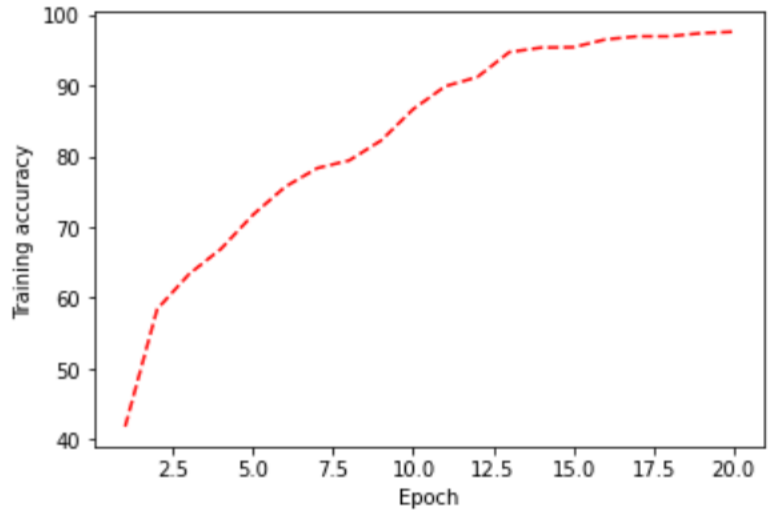
- Convolutional layer 1 – 32 Filters
- Convolutional layer 2 – 32 Filters
- Convolutional layer 3 – 64 Filters
- Convolutional layer 4 – 64 Filters
- Fully Connected layer 1 – 4096 to 1000
- Fully Connected layer 2 – 1000 to 512
- Fully Connected layer 3/Output Layer – 512 to 5

Training

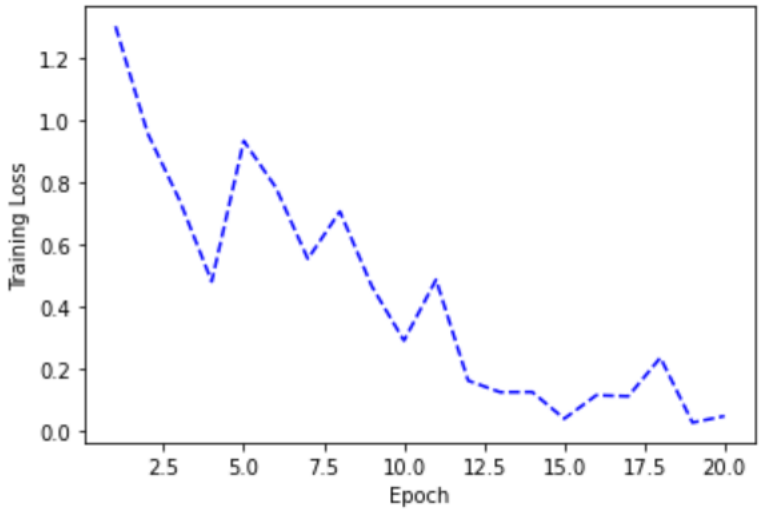
The size of our input images are 32*32*3. We used the Adam Optimizer, our optimizer, and CrossEntropyLoss as our loss function. Our model was trained for 20 epochs on 80% of the entire Dataset in batches of size 64. The training and test split was a random split to avoid imbalance of classes during the training phase. The learning rate was set to 0.001. After 20 epochs the training accuracy was around 74%.

```
----------------------------------------------------------------
        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
          Dropout-15                [-1, 4096]               0
           Linear-16                [-1, 1000]       4,097,000
             ReLU-17                [-1, 1000]               0
           Linear-18                 [-1, 512]         512,512
             ReLU-19                 [-1, 512]               0
          Dropout-20                 [-1, 512]               0
           Linear-21                   [-1, 5]           2,565
================================================================
Total params: 4,678,029
Trainable params: 4,678,029
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.01
Forward/backward pass size (MB): 2.40
Params size (MB): 17.85
Estimated Total Size (MB): 20.26
----------------------------------------------------------------
```

```
Text(0, 0.5, 'Training accuracy')
```




```
Text(0, 0.5, 'Training Loss')
```



As we increase the number of epochs training accuracy increases and training loss decreases.

Data Evaluation

After we ran our training, we obtained a trained model which can be used to test other images of our dataset and analyze their results. The procedure of splitting the dataset for training and testing randomly have been automatically done with python code. This test dataset is then passed through our trained model and model's accuracy is calculated and displayed. The values of precision, recall, F1-Measure are being displayed using the scikit library which we are showing in the Classification Report. In addition to that- True Positives, False Positives, True negatives, and False Negatives are used to predict the metrics of a classification report. The number of correct and incorrect predictions are summarized with count values and broken down by each class and shown in the confusion matrix. Finally, we are passing a random image to our trained model and results show which label the image belongs to.

Displayed Outputs:

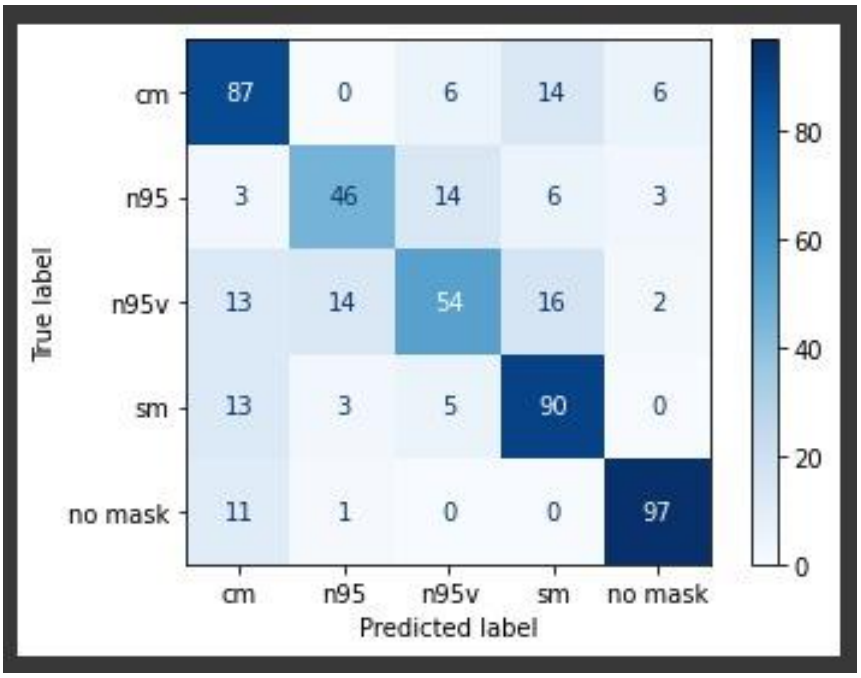
1. Accuracy:

```
Test Accuracy of the model on the test images: 74.20634920634922 %
```

2. Classification Report:

	precision	recall	f1-score	support
cm	0.69	0.77	0.72	113
n95	0.72	0.64	0.68	72
n95v	0.68	0.55	0.61	99
sm	0.71	0.81	0.76	111
no mask	0.90	0.89	0.89	109
accuracy			0.74	504
macro avg	0.74	0.73	0.73	504
weighted avg	0.74	0.74	0.74	504

3. Confusion Matrix



References:

Dataset reference:

1. https://www.kaggle.com/datasets/wobotintelligence/face-mask-detection-dataset

2. Face Mask Detection | Kaggle

3. COVID Face Mask Detection Dataset | Kaggle

4. Face Mask Detection Dataset | Kaggle

References for training the model:

1. PyTorch, Training a Classifier. URL : https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

2. https://medium.com/@Sanath_raj/using-pytorch-for-building-a-convolutional-neural-network-cnn-model-6e47ffb61b88

References for testing the model:

1. https://www.scikit-yb.org/en/latest/

2.https://scikitlearn.org/stable/modules/generated/sklearn.metrics.classification_report.html#sklearn.metrics.classification_report

3.https://scikitlearn.org/stable/modules/generated/sklearn.metrics.multilabel_confusion_matrix.html#sklearn.metrics.multilabel_confusion_matrix