CONCORDIA UNIVERSITY

COMP 6721 Applied AI Project by OB_2

Professor: *Dr. René Witte*

Student name: *Maryam Valipour (40224474) - Data Specialist,*
Kary Sutariya (40193909) - Training Specialist,
Kanika Aggarwal (40195981) - Evaluation Specialist

Project Repository of AI

1. DATA SET

Data for first phase of project has been collected mainly from google images and a Kaggle dataset [1] made by author RAHUL MANGALAMPALLI, which are publicly available. IMAGE DOWNLOADER software is used to collect the images from google. The images of class 'Mask worn incorrectly' are mostly gathered from Kaggle [1]. Most of images are taken from google [2] or Kaggle dataset. It is also worth mentioning that efforts have been made to collect data from social media websites, but not more than ten appropriate images were found using these platforms [4]. In the second phase of project, a few images have been removed and added. Significant changes can be seen for "Mask worn incorrectly" as now most of images for this category has been replaced by high resolution synthetic images [3].

Moreover, the new size of this new data set is 222 MB, which is 70MB higher compared to the old data set. In total, it contains 2090 images for all the categories. These images are divided into five classes, and the samples of each class are shown in Fig. 1 below.

Figure 1: Samples of each class of Data set

Each of the image comes in different sizes, resolutions, and formats (JPEG and PNG). The size and average resolution for each class are shown in Table 1. After collecting the required data for the project, data is pre-processed in order to be used in the neural network model for training.

Class	No. of images	Size	Resolution
Cloth mask	411	49.9 MB	461x422
Mask worn incorrectly	400	23.1 MB	395x425
N95 mask	423	24.6 MB	328x290
No face mask	461	23.9 MB	240x227
Surgical mask	430	29.8 MB	436x369

Table 1: Common Attributes of the Old Dataset

1.1. New Balanced Dataset.

Using the Kaggle dataset mentioned in the last section 1, the dataset was made balanced regarding gender (female and male) and age (young and old) subclass. A lot of effort has been made to make the number of images equal in all five categories across the subclasses (age and gender).

Table 2 depicts number of images, size and resolution for all five classes for the new and balanced dataset. Total size of the new data set is 152 MB.

Class	No. of images	Size	Resolution
Cloth mask	401	49.5 MB	699x595
Mask worn incorrectly	420	123 MB	1024x1024
N95 mask	461	35.1 MB	593x446
No face mask	402	1.86 MB	200x200
Surgical mask	406	12.8 MB	196x233

Table 2: Common Attributes of the New Dataset

The number of images for each class and category are depicted in Tables 3 and 4 . In the test set, 42.77% and 58% of the images are male and female, respectively. Moreover, The young and old people contributed to 58.19% and 41.81% of the images in the test set, respectively.

Category	Cloth mask	Mask worn incorrectly	N95 mask	No face mask	Surgical mask	Total	Percentage
Male	45	47	41	51	38	222	42.77%
Female	55	57	73	49	63	297	57.23%

Table 3: Number of images of Male and Female for training

Category	Cloth mask	Mask worn incorrectly	N95 mask	No face mask	Surgical mask	Total	Percentage
Young	52	59	70	63	58	302	58.19%
Old	48	45	44	37	43	217	41.81%

Table 4: Number of images of Young and Old for training

1.2. Data Preprocessing.

Firstly, the images and their respective categories are loaded using the PIL library into the program. After that, they are split into train, validation, and test categories using the Scikit-Learn library. Before preprocessing, the number of images in the train, validation, and test dataset are 1431, 159, and 535, respectively. Whereas in second phase, before preprocessing, 1570 images had for training and 520 images for testing whilst after precosseing, there were 5 less images for training and 1 image have been lost for testing. Overall, the number of images for training and testing was 2084.

In this next step, the torchvision.transforms.compose is used to compose several transforms together, which is used on the existing images. All transformation functions accept PIL images, and they can be passed to the transform functions. In this part, some images were removed from the datasets because they only had one channel and could not be resized to (3,224,224). The different transformations [5] used on the images are described below.

1. **Resize**: Since neural networks expect all the image data to be the same size, the images from all datasets should be resized into a particular and shared size. For the purposes of this project, we resize all the images into the size (3, 224, 224), where 3 is the number of channels, and 224 is the width and height of the images.

2. **Randomrotation**: This function rotates images 10 degrees from their angle.

3. **Randomhorizontalflip**: This function flips the images horizontally with a 50 percent probability. The reason for using this type of transformation is that if all the images are from the same angle, the network will be biased to this particular angle, and the network would not generalize well to the unseen data. So, this transformation will cause the network to become less biased, and the training process will be improved.

4. **ToTensor**: This will transform the images into tensors, which has to be done in order for the network to work properly.

5. **Normalization**: This helps to improve the performance of the model. So for this, the data should be normalized using the mean and standard deviation (std) of the images. For the purposes of this project, the mean and standard deviation of the ImageNet dataset [6] is used, which works well enough.

For the training dataset, all the above-mentioned transformations are used. However, for the validation and test dataset, only resize, ToTensor, and Normalization are used. The reason behind this is that for training, overfitting should be avoided, so other kinds of transformations are used to reduce the bias from the dataset, and for the validation and test dataset, the performance of the model on the real dataset should be measured.

After applying these transformations to the images, the number of the images for train, validation, and test datasets were changed to 1368, 147, and 506 images, respectively. For the labels, a number should be assigned to each class of the dataset, so that the network would understand the output classes. This is done using the Labelencoder from the Scikit-Learn library. The categories and their corresponging labels are listed in Table 5. below.

In the next step, the created datasets will be passed to the DataLoader module in the Pytorch library.

Classification	Label
Cloth mask	0
Mask worn incorrectly	1
N95 mask	2
No face mask	3
Surgical mask	4

Table 5: Data Preparation

Then, the data sets are shuffled and divided into batches of 64, so that each batch would contain 64 images. A sample of 64 batches of images are shown in Fig. 2.

Figure 2: Sample of the Preprocessed data set

1.2.1. Cross Validation.

For this task, annotation files of the training data set have been made for the k-fold validation process. In the ’annotation.csv’ file, information on the location of images, age, and gender have been manually written for the test phase of the model. As a random seed has been the seed for splitting the data into train and test, the same test data set will be produced every time. Therefore, the images included in the CSV file are always the same.

Moreover, in the train test split, the variable stratify is used. So, the number of images across all five categories in train and test are the same. There are around 300 and 100 images from each category in the train and test dataset, respectively.

It is also worth mentioning that, the mean and standard deviation of the train images are also calculated and used in the transformations, instead of the ImageNet [6] standards that was used in the last phase of the project.

More information on this process can be found in Section 5.

2. CNN ARCHITECTURE

In this study, approximately 75% of the data set contributes to the training set and validation set, and the rest to the testing set. The input images are pre-processed and augmented using the steps described in the last section. The model architecture proposed is composed of three phases, data pre-processing, CNN Model training, and Model Evaluation, which is shown in Fig. 3.

The Models are designed in python using PyTorch library [7]. Three different CNN based models are proposed to classify the images into the 5 mentioned categories. Three different model architectures are proposed in this project, and they all share a few hyperparamters, including learning rate, number of epochs for training, batch size, optimizer, and Loss function, which their values are presented in Table 6.

Below, some functions that are used in architecture of the models are described:

1. **Convolution Layers**: Each convolution layer applies a convolution operation to the input and passes the result to the next layer.

2. **Activation Functions**: They are placed at the end or among the networks, which ultimately decide whether to fire a neuron or not. The choice of activation function at hidden layers as well as at the output layer is significant as it controls the quality of training the models. The LeakyReLu activation function is based on ReLU, and it has a small slope for negative values instead of a flat slope [8]. Therefore, LeakyReLU is used in hidden layers as it can help avoid the vanishing gradient problem and improve computational performance. Additionally, at the end of the network, a SoftMax activation function is used to calculate the probabilities distribution of each class to finally be able to predict the corresponding class of each image.

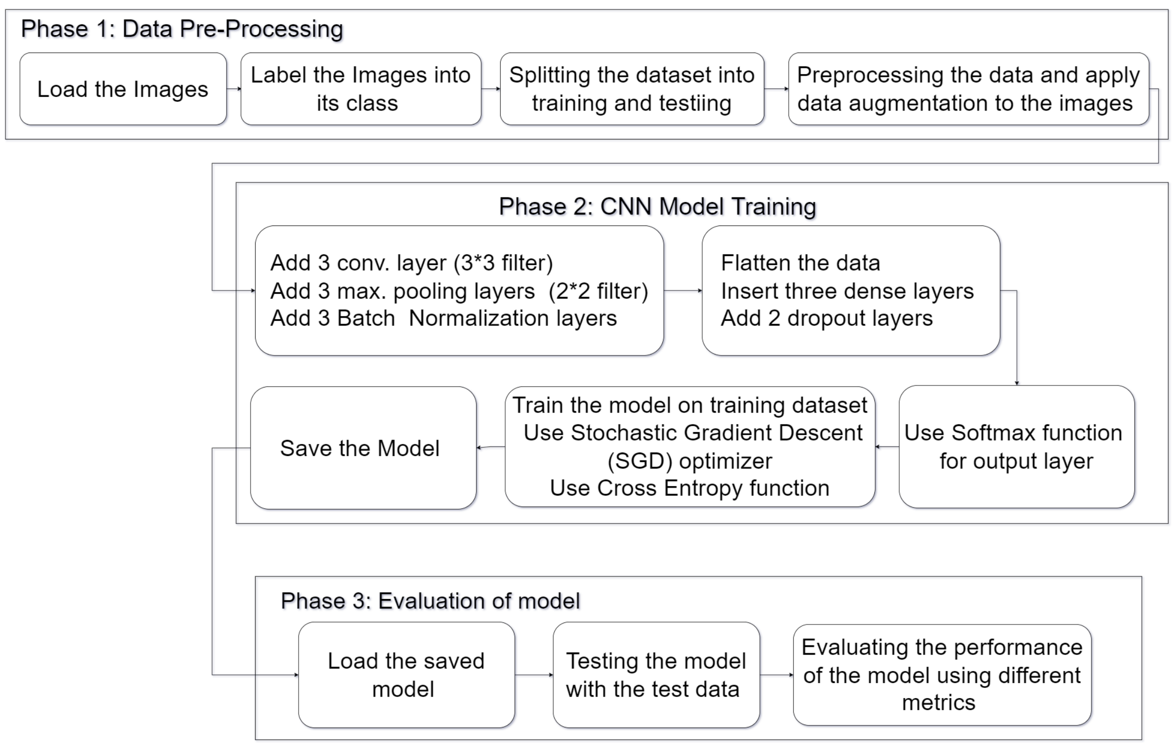


Figure 3: The proposed architecture

3. **Pooling Layers**: Pooling layers decrease the size of the feature map [9]. Thus, the number of trainable parameters is reduced, resulting in rapid calculations without losing the essential features. It also helps to reduce the risk of overfitting in large networks.

Parameter	**Values**
Learning Rate	0.001
Epochs	100
Batch Size	64
Optimizer	SGD
Loss Function	Cross Entropy

Table 6: Hyper Parameters used in Training

4. **Dropout Layers**: The Dropout layer is added to randomly skip some connections to force the network to learn from other parameters. It also helps to avoid overfitting [10].

5. **Flatten Layer**: The entire filter maps are then flattened to provide features to the classifier.

2.1. Models.

In the first model, 3 Conv blocks and one FC block are used in the model design. Each Conv block shares the same pattern. They are composed of one Conv2D layer, with 3*3 filter and 'same' padding, one LeakyRelu activation function layer, one batch normalization layer, and one Max-pooling layer with filter size 2 x 2 and stride 2. Additionally, the FC block contains four fully connected layers, Relu layers, and two Dropout layers with a probability of 0.5. The final layer of the neural network has only five outputs, which helps to classify the images into 5 classes with their corresponding values. The summary of the first model layers and their output size is presented in Table 7.

The second model includes 2 Conv blocks and one FC block (The blocks used are the same as in the first model). The layers and the respective output sizes of the second model are also presented in Table 8.

In the third model, two Conv block and one FC block are used. The Conv block used is the same as previous models with one difference, which is that max-pooling layers are omitted from the Conv block. The FC block used is the same as previous models. Similarly, the layers and the respective output sizes of the third model are presented in Table 9.

3. EVALUATION

In this multiclass classification problem, images are classified into five different categories i.e., Cloth mask, No face mask, Surgical mask, N95 mask and mask worn incorrectly. The models are trained using previously described hyper parameters. They are trained and validated for 100 epochs on the train and validation datasets, respectively.

Layer(type)	Output Shape	Structure
Conv2D	16 * 224 * 224	Filters = 16, Filter Size = 3 * 3, Stride = 1
MaxPooling	16 * 112 * 112	Filters = 16, Filter Size = 2 * 2 , Stride = 2
Conv2D	32 * 112 * 112	Filters = 32 , Filter Size = 3 * 3 , Stride = 1
MaxPooling	32 * 56 * 56	Filters = 32, Filter Size = 2 * 2 , Stride = 2
Conv2D	64 * 56 * 56	Filters = 64 , Filter Size = 3 * 3, Stride = 1
MaxPooling	64 * 28 * 28	Filters = 64, Filter Size = 2 * 2
FC	50176 * 128	-
FC	128 * 64	-
FC	64 * 32	-
FC	32 * 5	-
Classification Layer	5	Softmax

Table 7: Model 1 Summary

Layer(type)	Output Shape	Structure
Conv2D	16 * 224 * 224	Filters = 16, Filter Size = 3 * 3, Stride = 1
MaxPooling	16 * 112 * 112	Filters = 16, Filter Size = 2 * 2 , Stride = 2
Conv2D	32 * 112 * 112	Filters = 32 , Filter Size = 3 * 3 , Stride = 1
MaxPooling	32 * 56 * 56	Filters = 32, Filter Size = 2 * 2 , Stride = 2
FC	100352 * 128	-
FC	128 * 64	-
FC	64 * 32	-
FC	32 *5	-
Classification Layer	5	Softmax

Table 8: Model 2 Summary

Layer(type)	Output Shape	Structure
Conv2D	16 * 224 * 224	Filters = 16, Filter Size = 3 * 3, Stride = 1
Conv2D	32 * 224 * 224	Filters = 32, Filter Size = 3 * 3, Stride = 1
FC	1605632 * 128	-
FC	128 * 64	-
FC	64 * 32	-
FC	32 * 5	-
Classification Layer	5	Softmax

Table 9: Model 3 Summary

The loss and accuracy of both training and validation phases for the three proposed models are summarized in Tables 7, 8, and 9.

Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
10	1.3724	42.94	1.3788	52.27
20	1.1823	54.06	1.1765	57.59
30	1.0360	63.3	1.1040	57.33
40	0.9037	67.68	1.0456	59.75
50	0.7896	71.78	0.9499	65.12
60	0.6490	77.91	0.8001	65.30
70	0.5565	82.03	0.8057	69.29
80	0.4453	86.61	0.8691	71.96
90	0.3688	91.25	0.8144	71.44
100	0.2835	93.04	0.8221	80.101

Table 7: Loss and Accuracy for Train and Validation of Model 1

Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
10	1.3834	40.93	1.4312	42.59
20	1.1047	58.96	1.2771	59.90
30	0.8690	70.64	1.0783	61.69
40	0.7039	76.63	0.9149	66.12
50	0.5441	84.64	0.9528	72.45
60	0.4140	89.99	0.8904	72.56
70	0.3181	91.62	0.7766	69.43
80	0.2549	95.08	0.8359	67.61
90	0.2142	95.65	0.7594	67.35
100	0.1818	95.54	0.7656	71.93

Table 8: Loss and Accuracy for Train and Validation of Model 2

Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
10	0.9316	68.73	1.2111	57.78
20	0.5919	84.02	1.1704	56.06
30	0.3742	90.32	1.1210	62.01
40	0.2577	93.61	1.1452	63.90
50	0.1812	95.31	1.0878	62.04
60	0.1446	96.47	1.0546	59.02
70	0.1172	97.66	1.1973	61.75
80	0.0958	97.68	1.1846	60.22
90	0.0916	97.75	1.1008	60.71
100	0.0943	96.97	1.1543	64.28

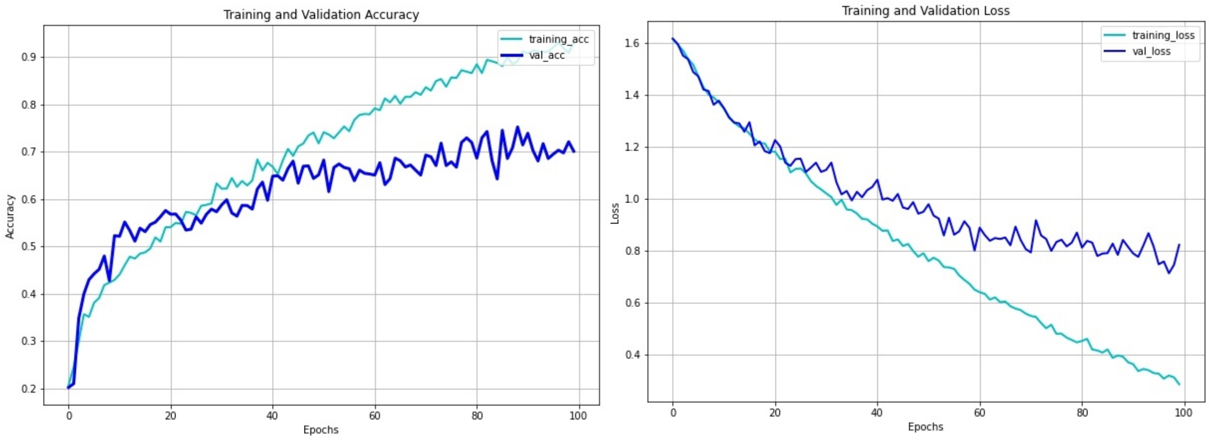
Table 9: Loss and Accuracy for Train and Validation of Model 3

Additionally, the loss and accuracy of the three proposed models in the training and validation process are also shown in Figures 4, 5, and 6.

For testing, 500 different images have been selected which are distributed amongst these five classes. The accuracy of the first, second, and third model for the test dataset is 72.33, 70.75, and 65.91, respectively.

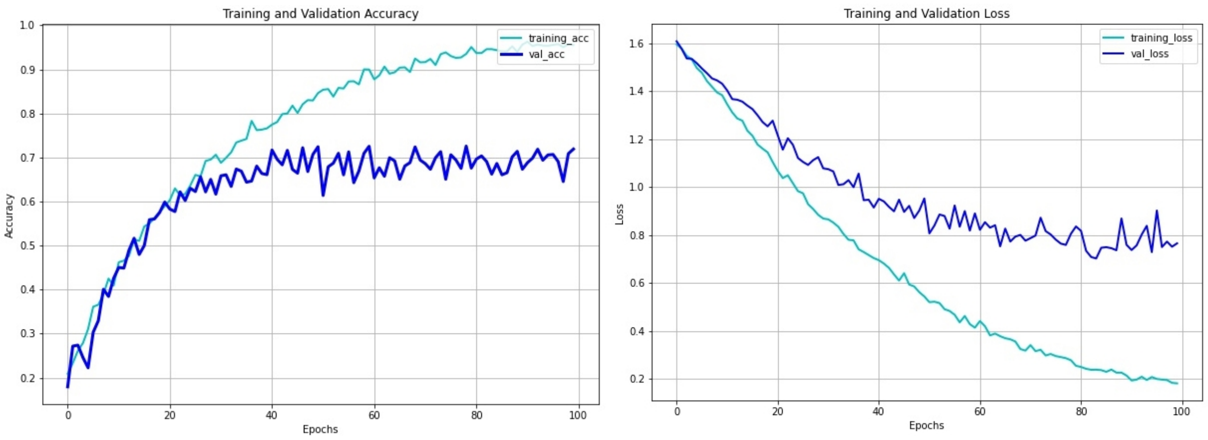
It is known that by adding more convolutional layers and therefore adding more parameters to the network, the performance of the model should improve. However, this has a limit, and from a certain point, adding more convolutional layers will only result in overfitting, and the performance of the model on unseen data would decrease.

The difference between the first and the second model is in the number of convolutional layers used in their architectures, which is three and two convolutional layers, respectively. The first model is doing a little better on the test dataset than the second model; however, the difference is not much noticeable. This means that from this point on, by adding more convolutional layers, the performance might not improve and could result in overfitting.



(a) Training and Validation accuracy (b) Training and Validation loss

Figure 4: Accuracy and Loss test results during Model 1 training

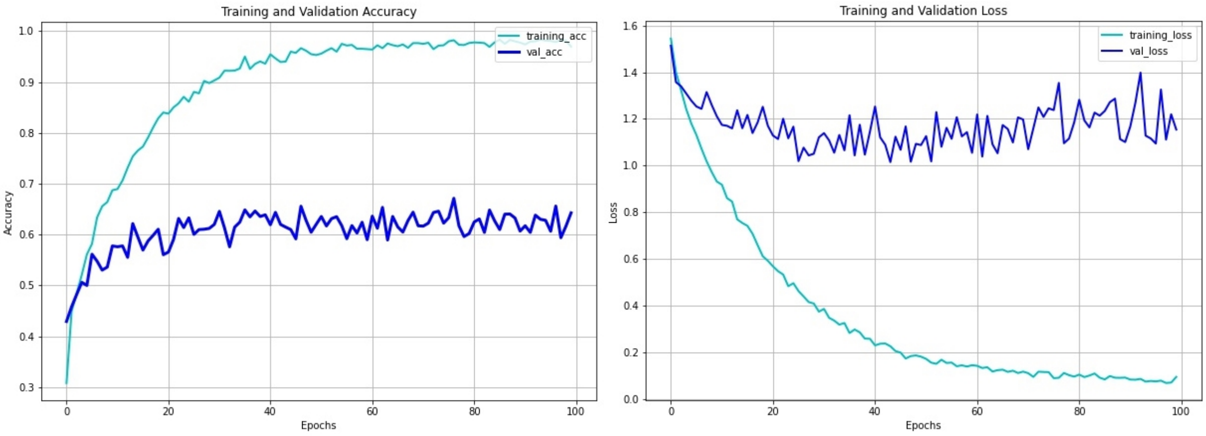


(a) Training and Validation accuracy (b) Training and Validation loss

Figure 5: Accuracy and Loss test results during Model 2 training

As can be seen in the above tables and plots, in the first few epochs, the third model is performing much better than the first and second model. However, at the end, it is performing worse than them. It can be seen that the model is quickly overfitting to the training images and the training accuracy is quickly rising. The reason is that in the third model, there is no max-pooling layer used in the model architecture. Therefore, the output size of the convolutional neural network is very large (1605632) as described in Table 9. So, it can be seen that the model is overfitting. As was mentioned before, the use of max-pooling layers helps avoid overfitting by reducing the number of parameters.

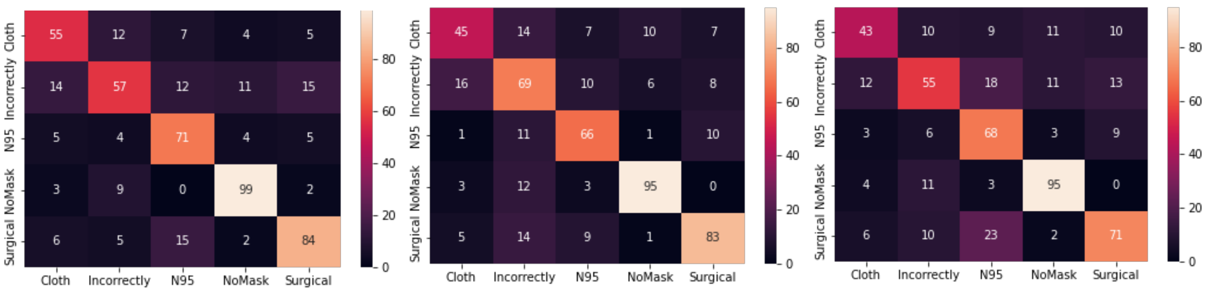
As can be seen, the third model has the lowest accuracy in the test dataset. Therefore, in this project, omitting the max-pooling layers in the third model had greatly impacted the model performance and the test accuracy.



(a) Training and Validation accuracy (b) Training and Validation loss

Figure 6: Accuracy and Loss test results during Model 3 training

The three confusion matrix have been generated in Fig 7. for the three proposed models. The confusion matrix makes easy to show whether the model is performing correctly i.e., which classes the model is predicting correctly and which the model is predicting incorrectly.
The final results of the proposed models are presented in the classification report as shown in Fig 8. Various metrics have been used to evaluate the models i.e., accuracy, precision, recall, and f1 score [11].



(a) Model 1 (b) Model 2 (c) Model 3

Figure 7: Confusion Matrix

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \tag{1}$$

$$Precision = \frac{Tp}{Tp + Fp} \tag{2}$$

$$Recall = \frac{Tp}{Tp + Fn} \tag{3}$$

$$F1 = (\frac{2 * Precision * Recall}{Precision + Recall}) \tag{4}$$

In the above equations, Tp represents: True positive, Tn: True negative, Fp: False positive, and Fn: False negative.

In True positive, model accurately predicted the positive class, whereas in false positive the model incorrectly predicted the positive class. In True negative, model accurately predicted the negative class, whereas in false negatives the model incorrectly predicted the negative class.

Classification Report:

	precision	recall	f1-score	support
0.0	0.66	0.66	0.66	83
1.0	0.66	0.52	0.58	109
2.0	0.68	0.80	0.73	89
3.0	0.82	0.88	0.85	113
4.0	0.76	0.75	0.75	112
accuracy			0.72	506
macro avg	0.72	0.72	0.72	506
weighted avg	0.72	0.72	0.72	506

(a) Model 1

Classification Report:

	precision	recall	f1-score	support
0.0	0.64	0.54	0.59	83
1.0	0.57	0.63	0.60	109
2.0	0.69	0.74	0.72	89
3.0	0.84	0.84	0.84	113
4.0	0.77	0.74	0.75	112
accuracy			0.71	506
macro avg	0.70	0.70	0.70	506
weighted avg	0.71	0.71	0.71	506

(b) Model 2

Classification Report:

	precision	recall	f1-score	support
0.0	0.63	0.52	0.57	83
1.0	0.60	0.50	0.55	109
2.0	0.56	0.76	0.65	89
3.0	0.78	0.84	0.81	113
4.0	0.69	0.63	0.66	112
accuracy			0.66	506
macro avg	0.65	0.65	0.65	506
weighted avg	0.66	0.66	0.65	506

(c) Model 3

Figure 8: Classification Report

In conclusion, the main model has better performance in comparison to the two other models.

3.1. Improvements.

Since the task of classifying these images into the five categories is complex and complicated, a large amount of data is needed to train the model. Then, the model can perform better and yield better accuracy.

Moreover, Bias is another issue. For instance, in our dataset, most of the images are of young males, and is a little biased towards gender, age, and race. A balanced dataset will help to improve the model for real-time applications.

As the number of images are low, data augmentation and several oversampling techniques such as SMOTE (Synthetic Minority Oversampling Technique) and Fair SMOTE should be done to increase the number of training images. Using these techniques can reduce bias as well.

Additionally, different and more complex models such as ResNets, R-CNNs, Fast R-CNNs, Faster R-CNN, YOLO can be used to further improve the overall performance.

4. RESNET18 ARCHITECTURE - NEW MODEL

Since the dataset is quite huge, a much deeper network was needed for the model to reach a higher accuracy in classifying the images into the five categories. The Resnet models are usually used in image classification tasks, which were first proposed by Kaiming et al. [12] in 2015. Therefore, for the purposed of this project, resent 18 architecture was chosen.

Larger networks are more prone to the vanishing gradient problem. Resent leverage a technique called "residuals" in order to solve this problem, which basically adds the output from the previous layer to the current layer output, which is shown in the Fig. 9.

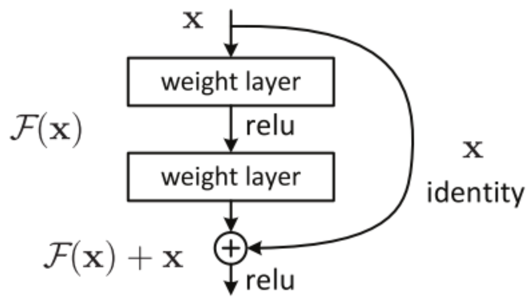


Figure 9: A Residual Block [12]

In the input layer, a Conv2d layer with stride two is used followed by a MaxPool2d, BatchNorm, and ReLU layers. In the next layers (1-4), resblokcs, consisting of Conv2d, BatchNorm, and ReLU layers, are used without MaxPool2d. In the final layer, global average pooling and a fully connected layer are stacked on top of each other to concatenate all the values into a one-dimensional list. The architecture of the model ResNet-18 is depicted in Fig. 10.

5. K-fold Cross-Validation (kFCV)

Cross-validation is a resampling procedure to evaluate the machine learning model on unseen data. The k-Fold Cross Validation [13] function split the training dataset into k folds with equal numbers and attach the remained samples to the last fold. Each time, one of the folds is used for validation, and the rest is used for training. The model is being iterated over ten folds with ten epochs running over each and containing a batch size of 64. The accuracy will be averaged over all the obtained accuracies in every fold. It is a popular method because it is simple to understand and generally results in a less biased and optimistic estimate of the model performance than other methods, such as a simple train/test split.

```
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 112, 112]           9,472
         MaxPool2d-2           [-1, 64, 56, 56]               0
       BatchNorm2d-3           [-1, 64, 56, 56]             128
              ReLU-4           [-1, 64, 56, 56]               0
            Conv2d-5           [-1, 64, 56, 56]          36,928
       BatchNorm2d-6           [-1, 64, 56, 56]             128
            Conv2d-7           [-1, 64, 56, 56]          36,928
       BatchNorm2d-8           [-1, 64, 56, 56]             128
          ResBlock-9           [-1, 64, 56, 56]               0
           Conv2d-10           [-1, 64, 56, 56]          36,928
      BatchNorm2d-11           [-1, 64, 56, 56]             128
           Conv2d-12           [-1, 64, 56, 56]          36,928
      BatchNorm2d-13           [-1, 64, 56, 56]             128
         ResBlock-14           [-1, 64, 56, 56]               0
           Conv2d-15          [-1, 128, 28, 28]           8,320
      BatchNorm2d-16          [-1, 128, 28, 28]             256
           Conv2d-17          [-1, 128, 28, 28]          73,856
      BatchNorm2d-18          [-1, 128, 28, 28]             256
           Conv2d-19          [-1, 128, 28, 28]         147,584
      BatchNorm2d-20          [-1, 128, 28, 28]             256
         ResBlock-21          [-1, 128, 28, 28]               0
           Conv2d-22          [-1, 128, 28, 28]         147,584
      BatchNorm2d-23          [-1, 128, 28, 28]             256
           Conv2d-24          [-1, 128, 28, 28]         147,584
      BatchNorm2d-25          [-1, 128, 28, 28]             256
         ResBlock-26          [-1, 128, 28, 28]               0
           Conv2d-27          [-1, 256, 14, 14]          33,024
      BatchNorm2d-28          [-1, 256, 14, 14]             512
           Conv2d-29          [-1, 256, 14, 14]         295,168
      BatchNorm2d-30          [-1, 256, 14, 14]             512
           Conv2d-31          [-1, 256, 14, 14]         590,080
      BatchNorm2d-32          [-1, 256, 14, 14]             512
         ResBlock-33          [-1, 256, 14, 14]               0
           Conv2d-34          [-1, 256, 14, 14]         590,080
      BatchNorm2d-35          [-1, 256, 14, 14]             512
           Conv2d-36          [-1, 256, 14, 14]         590,080
      BatchNorm2d-37          [-1, 256, 14, 14]             512
         ResBlock-38          [-1, 256, 14, 14]               0
           Conv2d-39            [-1, 512, 7, 7]         131,584
      BatchNorm2d-40            [-1, 512, 7, 7]           1,024
           Conv2d-41            [-1, 512, 7, 7]       1,180,160
      BatchNorm2d-42            [-1, 512, 7, 7]           1,024
           Conv2d-43            [-1, 512, 7, 7]       2,359,808
      BatchNorm2d-44            [-1, 512, 7, 7]           1,024
         ResBlock-45            [-1, 512, 7, 7]               0
           Conv2d-46            [-1, 512, 7, 7]       2,359,808
      BatchNorm2d-47            [-1, 512, 7, 7]           1,024
           Conv2d-48            [-1, 512, 7, 7]       2,359,808
      BatchNorm2d-49            [-1, 512, 7, 7]           1,024
         ResBlock-50            [-1, 512, 7, 7]               0
AdaptiveAvgPool2d-51            [-1, 512, 1, 1]               0
           Linear-52                    [-1, 5]           2,565
================================================================
Total params: 11,183,877
Trainable params: 11,183,877
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.57
Forward/backward pass size (MB): 42.11
Params size (MB): 42.66
Estimated Total Size (MB): 85.35
----------------------------------------------------------------
```

Figure 10: Summary of ResNet-18 - New Model

5.1. Classification Report for each of the 10 fold.

For this section, the original (old) model architecture proposed (the model with the best performance out of the three proposed models) in the first phase of the project, which is depicted in Table 7, is used for comparison with the new proposed Resnet18 model.

Both models are evaluated using k-fold cross-validation. The old model was trained for 100 epochs in each fold. However, the new model was trained for 50 epochs in each fold. The reason is that the Resnet18 model is significanely large and takes a lot more time to run. In total, it took approximately 20 hours to run for 50 epochs per fold.

Table 10 and 11 show the Train and Validation loss and accuracy value for the old and new model across each fold, respectively.

Folds	Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Fold 1	100	0.5696	81.10	0.9570	64.78
Fold 2	100	0.3506	90.10	0.9076	61.74
Fold 3	100	0.3575	90.92	0.9033	66.90
Fold 4	100	0.3006	92.78	0.8901	69.84
Fold 5	100	0.3578	86.08	1.0733	60.16
Fold 6	100	0.3837	88.71	0.9308	68.28
Fold 7	100	0.3061	92.54	0.7351	66.98
Fold 8	100	0.2991	92.61	0.9671	67.09
Fold 9	100	0.3973	85.58	1.7410	48.51
Fold 10	100	0.4391	85.44	0.9765	65.12

Table 10: Loss and Accuracy for Train and Validation - Old Model

Folds	Epochs	Train Loss	Train Accuracy	Val Loss	Val Accuracy
Fold 1	50	0.2201	94.89	0.4300	81.46
Fold 2	50	0.2642	91.19	0.3852	85.27
Fold 3	50	0.2412	93.47	0.5337	87.52
Fold 4	50	0.2308	94.18	0.4720	84.58
Fold 5	50	0.2442	93.61	0.3248	89.62
Fold 6	50	0.2779	94.57	0.3674	83.20
Fold 7	50	0.3400	88.79	0.3676	87.35
Fold 8	50	0.3280	89.27	0.4303	80.22
Fold 9	50	0.2871	94.29	0.4437	85.33
Fold 10	50	0.3592	88.93	0.5310	79.41

Table 11: Loss and Accuracy for Train and Validation - New Model

```
Fold1:
Testing Accuracy of the model on the test images
for fold1 57.03275529865125 %
Classification Report for the Whole dataset for fold1:
              precision    recall  f1-score   support

         0.0       0.71      0.37      0.49       100
         1.0       0.40      0.43      0.42       104
         2.0       0.69      0.61      0.65       114
         3.0       0.55      0.92      0.69       100
         4.0       0.60      0.51      0.56       101

    accuracy                           0.57       519
   macro avg       0.59      0.57      0.56       519
weighted avg       0.59      0.57      0.56       519
```

(a) Fold 1

```
Fold2:
Testing Accuracy of the model on the test images
for fold2 59.53757225433526 %
Classification Report for the Whole dataset for fold2:
              precision    recall  f1-score   support

         0.0       0.70      0.38      0.49       100
         1.0       0.41      0.44      0.42       104
         2.0       0.73      0.65      0.69       114
         3.0       0.61      0.95      0.75       100
         4.0       0.59      0.55      0.57       101

    accuracy                           0.60       519
   macro avg       0.61      0.60      0.58       519
weighted avg       0.61      0.60      0.59       519
```

(b) Fold 2

```
Fold3:
Testing Accuracy of the model on the test images
for fold3 57.22543352601156 %
Classification Report for the Whole dataset for fold3:
              precision    recall  f1-score   support

         0.0       0.71      0.34      0.46       100
         1.0       0.40      0.53      0.45       104
         2.0       0.63      0.66      0.64       114
         3.0       0.66      0.90      0.76       100
         4.0       0.55      0.43      0.48       101

    accuracy                           0.57       519
   macro avg       0.59      0.57      0.56       519
weighted avg       0.59      0.57      0.56       519
```

(c) Fold 3

```
Fold4:
Testing Accuracy of the model on the test images
for fold4 54.14258188824663 %
Classification Report for the Whole dataset for fold4:
              precision    recall  f1-score   support

         0.0       0.43      0.23      0.30       100
         1.0       0.36      0.39      0.38       104
         2.0       0.58      0.59      0.58       114
         3.0       0.67      0.95      0.79       100
         4.0       0.58      0.54      0.56       101

    accuracy                           0.54       519
   macro avg       0.52      0.54      0.52       519
weighted avg       0.52      0.54      0.52       519
```

(d) Fold 4

```
Fold5:
Testing Accuracy of the model on the test images
for fold5 56.84007707129094 %
Classification Report for the Whole dataset for fold5:
              precision    recall  f1-score   support

         0.0       0.64      0.37      0.47       100
         1.0       0.41      0.62      0.49       104
         2.0       0.57      0.57      0.57       114
         3.0       0.82      0.68      0.74       100
         4.0       0.57      0.60      0.59       101

    accuracy                           0.57       519
   macro avg       0.60      0.57      0.57       519
weighted avg       0.60      0.57      0.57       519
```

(e) Fold 5

```
Fold6:
Testing Accuracy of the model on the test images
for fold6 60.69364161849711 %
Classification Report for the Whole dataset for fold6:
              precision    recall  f1-score   support

         0.0       0.69      0.31      0.43       100
         1.0       0.49      0.62      0.54       104
         2.0       0.65      0.64      0.65       114
         3.0       0.66      0.96      0.78       100
         4.0       0.60      0.50      0.55       101

    accuracy                           0.61       519
   macro avg       0.62      0.61      0.59       519
weighted avg       0.62      0.61      0.59       519
```

(f) Fold 6

```
Fold7:
Testing Accuracy of the model on the test images
for fold7 53.5645472061657 %
Classification Report for the Whole dataset for fold7:
              precision    recall  f1-score   support

         0.0       0.48      0.15      0.23       100
         1.0       0.35      0.68      0.46       104
         2.0       0.70      0.49      0.58       114
         3.0       0.76      0.82      0.79       100
         4.0       0.56      0.53      0.55       101

    accuracy                           0.54       519
   macro avg       0.57      0.54      0.52       519
weighted avg       0.57      0.54      0.52       519
```

(g) Fold 7

```
Fold8:
Testing Accuracy of the model on the test images
for fold8 60.115606936416185 %
Classification Report for the Whole dataset for fold8:
              precision    recall  f1-score   support

         0.0       0.64      0.34      0.44       100
         1.0       0.43      0.49      0.46       104
         2.0       0.68      0.66      0.67       114
         3.0       0.63      0.97      0.77       100
         4.0       0.65      0.54      0.59       101

    accuracy                           0.60       519
   macro avg       0.61      0.60      0.59       519
weighted avg       0.61      0.60      0.59       519
```

(h) Fold 8

```
Fold9:
Testing Accuracy of the model on the test images
for fold9 40.847784200385355 %
Classification Report for the Whole dataset for fold9:
              precision    recall  f1-score   support

         0.0       0.34      0.43      0.38       100
         1.0       0.18      0.27      0.22       104
         2.0       0.40      0.02      0.03       114
         3.0       0.69      0.87      0.77       100
         4.0       0.48      0.51      0.50       101

    accuracy                           0.41       519
   macro avg       0.42      0.42      0.38       519
weighted avg       0.42      0.41      0.37       519
```

(i) Fold 9

```
Fold10:
Testing Accuracy of the model on the test images
for fold10 59.34489402697495 %
Classification Report for the Whole dataset for fold10:
              precision    recall  f1-score   support

         0.0       0.70      0.30      0.42       100
         1.0       0.45      0.52      0.48       104
         2.0       0.70      0.60      0.64       114
         3.0       0.61      0.94      0.74       100
         4.0       0.58      0.61      0.60       101

    accuracy                           0.59       519
   macro avg       0.61      0.59      0.58       519
weighted avg       0.61      0.59      0.58       519
```

(j) Fold 10

Figure 11: Classification Report for each of the 10 fold - Old Model

The classification reports for each 10 folds in the old and new model are shown in Fig. 11 and Fig. 12, respectively.

Folds	Accuracy	Weighted Average			
		Precision	Recall	f1-score	support
Fold 1	0.57	0.59	0.57	0.56	519
Fold 2	0.60	0.61	0.60	0.60	519
Fold 3	0.57	0.59	0.57	0.56	519
Fold 4	0.54	0.52	0.54	0.52	519
Fold 5	0.57	0.60	0.57	0.57	519
Fold 6	0.61	0.62	0.61	0.59	519
Fold 7	0.54	0.57	0.54	0.52	519
Fold 8	0.60	0.61	0.60	0.59	519
Fold 9	0.42	0.42	0.41	0.37	519
Fold 10	0.59	0.61	0.59	0.58	519

Table 12: Weighted Average for each of 10 fold - Old Model

```
Fold1:
Testing Accuracy of the model on the test images
for fold1 86.51252408477842 %
Classification Report for the Whole dataset for fold1:
              precision    recall  f1-score   support

         0.0       0.75      0.74      0.74       100
         1.0       0.99      0.98      0.99       104
         2.0       0.74      0.79      0.76       114
         3.0       0.93      0.99      0.96       100
         4.0       0.95      0.83      0.89       101

    accuracy                           0.87       519
   macro avg       0.87      0.87      0.87       519
weighted avg       0.87      0.87      0.87       519
```

(a) Fold 1

```
Fold2:
Testing Accuracy of the model on the test images
for fold2 85.74181117533719 %
Classification Report for the Whole dataset for fold2:
              precision    recall  f1-score   support

         0.0       0.79      0.62      0.70       100
         1.0       1.00      1.00      1.00       104
         2.0       0.73      0.82      0.77       114
         3.0       0.93      0.98      0.96       100
         4.0       0.84      0.86      0.85       101

    accuracy                           0.86       519
   macro avg       0.86      0.86      0.86       519
weighted avg       0.86      0.86      0.85       519
```

(b) Fold 2

```
Fold3:
Testing Accuracy of the model on the test images
for fold3 86.1271676300578 %
Classification Report for the Whole dataset for fold3:
              precision    recall  f1-score   support

         0.0       0.75      0.74      0.74       100
         1.0       0.99      0.97      0.98       104
         2.0       0.75      0.81      0.78       114
         3.0       0.94      0.95      0.95       100
         4.0       0.89      0.84      0.87       101

    accuracy                           0.86       519
   macro avg       0.87      0.86      0.86       519
weighted avg       0.86      0.86      0.86       519
```

(c) Fold 3

```
Fold4:
Testing Accuracy of the model on the test images
for fold4 84.77842003853564 %
Classification Report for the Whole dataset for fold4:
              precision    recall  f1-score   support

         0.0       0.82      0.61      0.70       100
         1.0       0.98      0.98      0.98       104
         2.0       0.72      0.83      0.77       114
         3.0       0.93      0.96      0.95       100
         4.0       0.81      0.85      0.83       101

    accuracy                           0.85       519
   macro avg       0.85      0.85      0.85       519
weighted avg       0.85      0.85      0.85       519
```

(d) Fold 4

```
Fold5:
Testing Accuracy of the model on the test images
for fold5 84.58574181117534 %
Classification Report for the Whole dataset for fold5:
              precision    recall  f1-score   support

         0.0       0.70      0.61      0.65       100
         1.0       0.99      0.99      0.99       104
         2.0       0.69      0.80      0.74       114
         3.0       0.93      0.98      0.96       100
         4.0       0.95      0.85      0.90       101

    accuracy                           0.85       519
   macro avg       0.85      0.85      0.85       519
weighted avg       0.85      0.85      0.85       519
```

(e) Fold 5

```
Fold6:
Testing Accuracy of the model on the test images
for fold6 84.20038535645472 %
Classification Report for the Whole dataset for fold6:
              precision    recall  f1-score   support

         0.0       0.67      0.75      0.71       100
         1.0       1.00      0.99      1.00       104
         2.0       0.74      0.69      0.71       114
         3.0       0.94      0.96      0.95       100
         4.0       0.88      0.83      0.86       101

    accuracy                           0.84       519
   macro avg       0.85      0.85      0.85       519
weighted avg       0.85      0.84      0.84       519
```

(f) Fold 7

```
Fold7:
Testing Accuracy of the model on the test images
for fold7 85.9344894026975 %
Classification Report for the Whole dataset for fold7:
              precision    recall  f1-score   support

         0.0       0.82      0.62      0.70       100
         1.0       0.99      0.99      0.99       104
         2.0       0.71      0.83      0.77       114
         3.0       0.94      0.97      0.96       100
         4.0       0.87      0.88      0.88       101

    accuracy                           0.86       519
   macro avg       0.87      0.86      0.86       519
weighted avg       0.86      0.86      0.86       519
```

(g) Fold 8

```
Fold8:
Testing Accuracy of the model on the test images
for fold8 82.46628131021194 %
Classification Report for the Whole dataset for fold8:
              precision    recall  f1-score   support

         0.0       0.74      0.59      0.66       100
         1.0       0.98      0.99      0.99       104
         2.0       0.68      0.74      0.71       114
         3.0       0.91      0.95      0.93       100
         4.0       0.81      0.86      0.84       101

    accuracy                           0.82       519
   macro avg       0.83      0.83      0.82       519
weighted avg       0.82      0.82      0.82       519
```

(h) Fold 9

```
Fold9:
Testing Accuracy of the model on the test images
for fold9 83.62235067437379 %
Classification Report for the Whole dataset for fold9:
              precision    recall  f1-score   support

         0.0       0.74      0.57      0.64       100
         1.0       0.99      0.99      0.99       104
         2.0       0.69      0.82      0.75       114
         3.0       0.91      0.98      0.94       100
         4.0       0.88      0.81      0.85       101

    accuracy                           0.84       519
   macro avg       0.84      0.84      0.83       519
weighted avg       0.84      0.84      0.83       519
```

(i) Fold 10

```
Fold10:
Testing Accuracy of the model on the test images
for fold10 85.35645472061657 %
Classification Report for the Whole dataset for fold10:
              precision    recall  f1-score   support

         0.0       0.73      0.69      0.71       100
         1.0       0.99      0.99      0.99       104
         2.0       0.72      0.77      0.74       114
         3.0       0.96      0.97      0.97       100
         4.0       0.90      0.85      0.87       101

    accuracy                           0.85       519
   macro avg       0.86      0.85      0.86       519
weighted avg       0.85      0.85      0.85       519
```

(j) Fold 10

Figure 12: Classification Report for each of the 10 folds - New Model

Folds	Accuracy	Weighted Average			
		Precision	Recall	f1-score	support
Fold 1	0.87	0.87	0.87	0.87	519
Fold 2	0.86	0.86	0.86	0.85	519
Fold 3	0.86	0.86	0.86	0.86	519
Fold 4	0.85	0.85	0.85	0.85	519
Fold 5	0.85	0.85	0.85	0.85	519
Fold 6	0.84	0.85	0.84	0.84	519
Fold 7	0.86	0.86	0.86	0.86	519
Fold 8	0.82	0.82	0.82	0.82	519
Fold 9	0.84	0.84	0.84	0.83	519
Fold 10	0.85	0.85	0.85	0.85	519

Table 13: Weighted Average for each of 10 folds - New Model

Aggregated	Old Model				New Model			
	Precision	Recall	f1-score	Support	Precision	Recall	f1-score	Support
Macro Avg.	0.57	0.56	0.54	519	0.85	0.85	0.85	519
Weighted Avg.	0.57	0.55	0.54	519	0.85	0.85	0.85	519
Accuracy	0.55				0.85			

Table 14: Aggregate statistics across the 10 folds

To show more clearly, Table 12 and Table 13 depicts the weighted average for each of the 10 folds in old and new model.

5.2. Aggregate statistics across the 10 folds.

The aggregate statistics across 10 folds, including precision, recall, F1-score, and accuracy are reported in Table 14 for old and new model.

The aggregate accuracy of the old and new model are 57% and 0.85%, respectively. This shows that the new model is performing a lot better on the new dataset. Additionally, the values of the precision, recall, and F1-score are better by approximately 30% in the new model.

5.3. Comparison between Cross Validation and Train Test Split.

In this part, the model trained on the old dataset using training/test split, depicted in Table 7, is being evaluated on the new test dataset and compared with the old model with k-fold cross-validation.

The classification report of the old model with training/test and cross validation are depicted in Fig. 13.

```
Fold10:
Testing Accuracy of the model on the test images
for fold10 59.34489402697495 %
Classification Report for the Whole dataset for fold10:
              precision    recall  f1-score   support

         0.0       0.70      0.30      0.42       100
         1.0       0.45      0.52      0.48       104
         2.0       0.70      0.60      0.64       114
         3.0       0.61      0.94      0.74       100
         4.0       0.58      0.61      0.60       101

    accuracy                           0.59       519
   macro avg       0.61      0.59      0.58       519
weighted avg       0.61      0.59      0.58       519
```

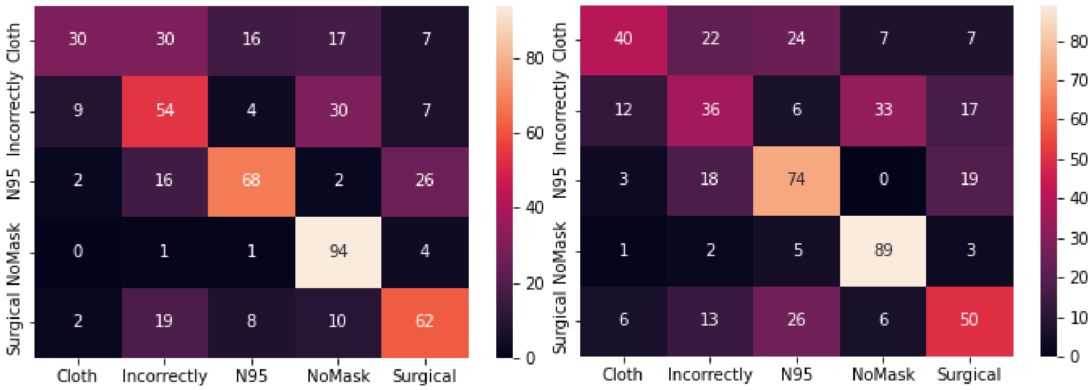
(a) Cross Validation

```
Testing Accuracy of the model on the test
snimages: 55.68400770712909 %
Classification Report for Whole dataset:
              precision    recall  f1-score   support

         0.0       0.65      0.40      0.49       100
         1.0       0.40      0.35      0.37       104
         2.0       0.55      0.65      0.59       114
         3.0       0.66      0.89      0.76       100
         4.0       0.52      0.50      0.51       101

    accuracy                           0.56       519
   macro avg       0.55      0.56      0.54       519
weighted avg       0.55      0.56      0.54       519
```

(b) Train Test Split

Figure 13: Classification Report - Old Model

The confusion matrix of the old model with training/test and cross validation are depicted in 14.



(a) Cross Validation (b) Train Test Split

Figure 14: Confusion Matrix - Old Model

As can be seen in the classification reports, the training/test split and cross-validation have an accuracy of 56% and 59%, respectively. Both models have quite the same accuracy. However, if you look at the precision, recall, and F1-score columns of the two models, the cross-validation model is performing far better in precision, recall, and F1-score metrics across all five categories. In the weighted average row, it is also clear that the cross-validation model is performing better in an imbalance of datasets.

In k-fold cross-validation, since every time a different part of the dataset is chosen for train and validation, the models are less biased towards the training dataset and can generalize better to unseen data. Therefore, the cross validation model is better than the training/test split model for different classes and across metrics that are more sensitive, such as precision, recall, and F1-score.

6. BIAS DETECTION AND ELIMINATION

For the Gender subclass, female and male are the attributes that have been analyzed for bias. While for age subclass, young and old attributes have been analyzed. To eliminate bias, the performance of two models, the old model (from part1) and the new model have been evaluated using the new dataset as described in Section 1. The previous dataset (part 1) was biased towards age and gender. To modify the new dataset into a balanced one, the number of images are made equal in all five categories across the subclasses (age and gender).

6.1. Performance evaluation metrics for each of the subclass.

The performance of the old model and the new model has been tested using the new dataset. The classification report for each of the subclasses and whole dataset has been generated for old model and new model respectively. The classification report of the old model displays the accuracy of 59%, 59%, 62% and 56% for female, male, young and old subclasses respectively.
The new model performs better than the old model on the new dataset. The new model correctly classifies the images into their class and subclass. The classification report of the new model displays the accuracy of 85%, 85%, 84%, and 88% for female, male, young and old subclasses, respectively. Thus, the accuracy of the new model is much better than the old model on the new dataset.

6.1.1. Old Model.

The final results of the old model on the new dataset are presented in the classification report as shown in Fig. 15. The figures show the precision, recall, f1- score, accuracy, and weighted average 5 for each subclass, respectively.

```
Classification Report for females in dataset:
              precision    recall  f1-score   support

         0.0       0.67      0.29      0.41        55
         1.0       0.41      0.46      0.43        57
         2.0       0.71      0.63      0.67        73
         3.0       0.59      0.98      0.74        49
         4.0       0.63      0.63      0.63        63

    accuracy                           0.59       297
   macro avg       0.60      0.60      0.57       297
weighted avg       0.61      0.59      0.58       297
```

(a) Female in dataset

```
Classification Report for male in dataset:
              precision    recall  f1-score   support

         0.0       0.67      0.29      0.41        55
         1.0       0.41      0.46      0.43        57
         2.0       0.71      0.63      0.67        73
         3.0       0.59      0.98      0.74        49
         4.0       0.63      0.63      0.63        63

    accuracy                           0.59       297
   macro avg       0.60      0.60      0.57       297
weighted avg       0.61      0.59      0.58       297
```

(b) Male in dataset

```
Classification Report for young in dataset:
              precision    recall  f1-score   support

         0.0       0.74      0.27      0.39        52
         1.0       0.47      0.54      0.50        59
         2.0       0.65      0.61      0.63        70
         3.0       0.67      0.97      0.79        63
         4.0       0.64      0.64      0.64        58

    accuracy                           0.62       302
   macro avg       0.63      0.61      0.59       302
weighted avg       0.63      0.62      0.60       302
```

(c) Young in dataset

```
Classification Report for old in dataset:
              precision    recall  f1-score   support

         0.0       0.67      0.33      0.44        48
         1.0       0.42      0.49      0.45        45
         2.0       0.81      0.57      0.67        44
         3.0       0.53      0.89      0.67        37
         4.0       0.52      0.58      0.55        43

    accuracy                           0.56       217
   macro avg       0.59      0.57      0.56       217
weighted avg       0.59      0.56      0.55       217
```

(d) Old in dataset

Figure 15: Classification Report for different subclasses - Old Model

6.1.2. New Model (Resnet18).

The final results of the new model on the new dataset are presented in the classification report as shown in Fig. 16. The figures show the precision, recall, f1- score, accuracy, and the weighted average for each subclass, respectively.

Classification Report for females in dataset:

	precision	recall	f1-score	support
0.0	0.67	0.64	0.65	55
1.0	0.98	0.98	0.98	57
2.0	0.76	0.81	0.78	73
3.0	0.96	0.98	0.97	49
4.0	0.92	0.87	0.89	63
accuracy			0.85	297
macro avg	0.86	0.86	0.86	297
weighted avg	0.85	0.85	0.85	297

(a) Female in dataset

Classification Report for male in dataset:

	precision	recall	f1-score	support
0.0	0.67	0.64	0.65	55
1.0	0.98	0.98	0.98	57
2.0	0.76	0.81	0.78	73
3.0	0.96	0.98	0.97	49
4.0	0.92	0.87	0.89	63
accuracy			0.85	297
macro avg	0.86	0.86	0.86	297
weighted avg	0.85	0.85	0.85	297

(b) Male in dataset

Classification Report for young in dataset:

	precision	recall	f1-score	support
0.0	0.63	0.60	0.61	52
1.0	0.98	0.98	0.98	59
2.0	0.69	0.77	0.73	70
3.0	0.98	0.98	0.98	63
4.0	0.91	0.83	0.86	58
accuracy			0.84	302
macro avg	0.84	0.83	0.84	302
weighted avg	0.84	0.84	0.84	302

(c) Young in dataset

Classification Report for old in dataset:

	precision	recall	f1-score	support
0.0	0.83	0.79	0.81	48
1.0	1.00	1.00	1.00	45
2.0	0.76	0.77	0.76	44
3.0	0.92	0.95	0.93	37
4.0	0.88	0.88	0.88	43
accuracy			0.88	217
macro avg	0.88	0.88	0.88	217
weighted avg	0.88	0.88	0.88	217

(d) Old in dataset

Figure 16: Classification Report for different subclasses - New Model

6.1.3. Whole Dataset.

The performance evaluation has been done on the whole dataset for the old model and new model. The final results of the new model and the old model across all five classes are presented in the classification report as shown in Fig. 17. The figures show the precision, recall, f1- score, accuracy, and the weighted average for each subclass, respectively. As we can see, the new model performs better with an accuracy of 85% than the old model with an accuracy of 59%.

Classification Report for the Whole dataset:

	precision	recall	f1-score	support
0.0	0.70	0.30	0.42	100
1.0	0.45	0.52	0.48	104
2.0	0.70	0.60	0.64	114
3.0	0.61	0.94	0.74	100
4.0	0.58	0.61	0.60	101
accuracy			0.59	519
macro avg	0.61	0.59	0.58	519
weighted avg	0.61	0.59	0.58	519

(a) Old Model

Classification Report for the Whole dataset:

	precision	recall	f1-score	support
0.0	0.73	0.69	0.71	100
1.0	0.99	0.99	0.99	104
2.0	0.72	0.77	0.74	114
3.0	0.96	0.97	0.97	100
4.0	0.90	0.85	0.87	101
accuracy			0.85	519
macro avg	0.86	0.85	0.86	519
weighted avg	0.85	0.85	0.85	519

(b) New Model

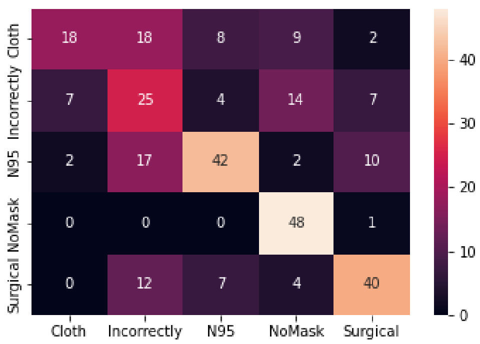
Figure 17: Classification Report for all 5 classes

6.2. Confusion Matrix for each of the subclasses .

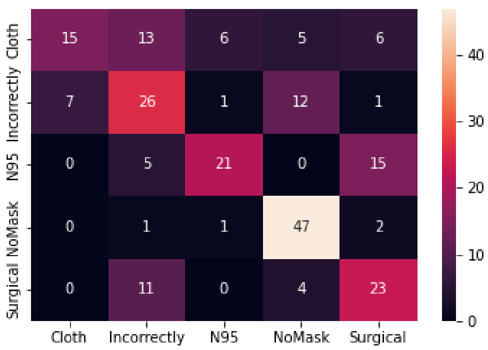
The confusion matrix for each of the subclasses and whole dataset has been generated for old model and new model respectively. The confusion matrix makes easy to show whether the model is performing correctly i.e., which classes the model is predicting correctly and which the model is predicting incorrectly.

6.2.1. Old Model.

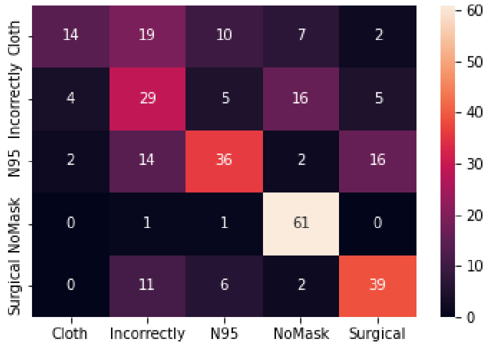
As we can see in Fig. 18, for class category: cloth mask, the old model performs poorly since it classifies less than 50% of the images correctly for all subclasses. Also, for the class: mask worn incorrectly in old category, the model misclassifies more images. However, the model performs better in classifying the images for class: no mask and all its subclasses.



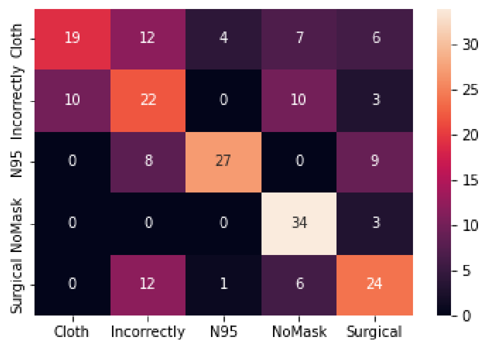
(a) Female in dataset



(b) Male in dataset



(c) Young in dataset

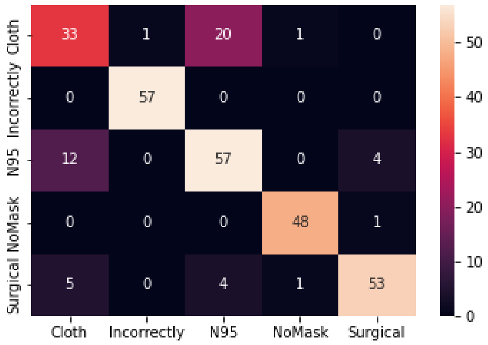


(d) Old in dataset

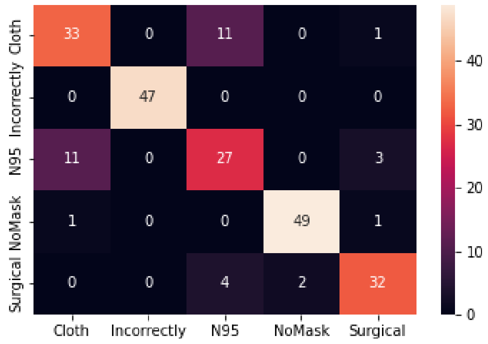
Figure 18: Confusion Matrix for different subclasses - Old Model

6.2.2. New Model (Resnet18).

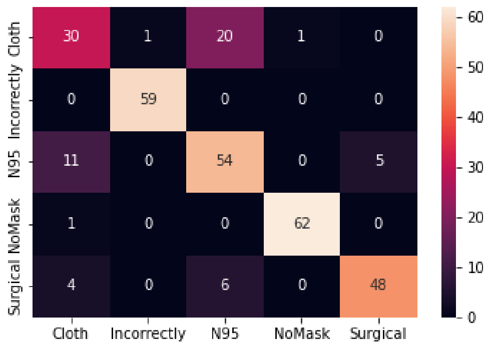
The new model correctly classifies the images into its classes and subclasses, as shown in Fig. 19. For instance, for all subclass, the model correctly classifies all the images into mask worn incorrectly class. Similarly, for subclasses female, male and young, the model accurately predicts the no mask, mask worn incorrectly, and surgical classes.



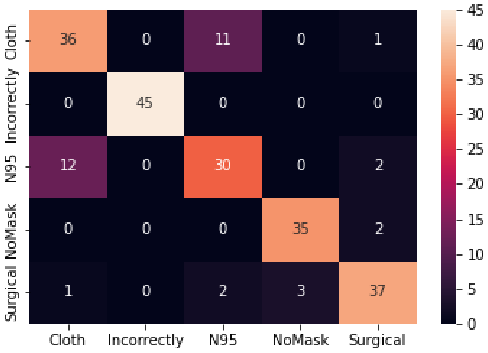
(a) Female in dataset



(b) Male in dataset



(c) Young in dataset

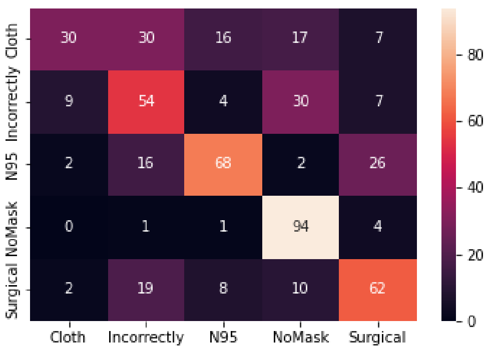


(d) Old in dataset

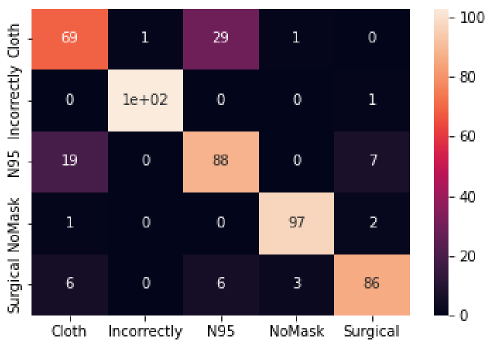
Figure 19: Confusion Matrix for different subclasses - New Model

6.2.3. Whole Dataset.

The confusion matrix for the old model and new model is depicted in Fig. 20. As we can see, the new model does classification better than the old model. For instance, for the class mask worn incorrectly, the new model classifies all the images correctly except one misclassified image. While the old model, it only classifies 50% of the images correctly. Thus, the new model gives better performance than the old model.



(a) Old Model



(b) New Model

Figure 20: Confusion Matrix for all 5 classes

6.3. Bias Analysis and comparison between old and new Model.

The performance of the old model and the new model has been tested on the new dataset. The classification report for each of the subclasses and the whole dataset has been generated for the old model and new model respectively. It shows the precision, recall, f1- score, accuracy, and the weighted average for each of the subclass respectively. The classification report of the old model displays the accuracy of 59%, 59%, 62%, and 56% for female, male, young and old subclasses respectively. While, the classification report of the new model displays the accuracy of 85%, 85%, 84%, and 88% for female, male, young and old subclasses respectively.

According to the above statistics, it is clearly shown that the new model performs better than the old model on the new dataset. In other words, the new model correctly classifies the images into its class and subclass. Thus, the new model performs better with an accuracy of 85% than the old model with an accuracy of 59%.

Moreover, the confusion matrix for the old model and new model also depicts that the new model does classification better than the old model. In Fig. 20, it can be seen that for the N95 mask class, the new model correctly classifies 88 images while for the old model the classification number is just 68. Similarly, for class masks worn incorrectly, the new model classifies all the images correctly except one misclassified image. While the old model, it only classifies 50% of the images correctly.

Thus, the new model performs a lot better than the old model over all subclass and the five categories.

REFERENCES

1. https://www.kaggle.com/datasets/rahulmangalampalli/mafa-data
2. https://www.google.com/search?q=worn+mask+incorrectly
3. https://github.com/cabani/MaskedFace-Net
4. https://www.facebook.com/search/photos/?q=mask%20worn%20incorrectly
5. https://towardsdatascience.com/improves-cnn-performance-by-applying-data-transformation
6. https://www.geeksforgeeks.org/how-to-normalize-images-in-pytorch
7. https://towardsdatascience.com/convolutional-neural-network-for-image-classification-with-implementation-on-python-using-pytorch
8. https://towardsdatascience.com/activation-functions-neural-networks
9. https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer
10. https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks
11. https://machinelearningmastery.com/precision-recall-and-f-measure-for-imbalanced-classification
12. Deep Residual Learning for Image Recognition
13. https://www.analyticsvidhya.com/blog/2022/02/k-fold-cross-validation-techniques