Concordia University

project phase 2 report

COMP6721

AI Face Mask Detector

Author:	*Author ID:*	*Specialization:*
Somayeh Ghahary	40106359	Training - Bias
Mehrnoosh Amjadi	40091264	Evaluation - K-Fold
Bikash Jaiswal	40115186	Data - Bias

Team Name: FL-08

Submitted to: Dr. René Witte

April 23, 2021

1 Phase 1

1.1 Dataset

In this project, we use two different image datasets: Face Mask Dataset and Cifar-10 dataset [4]. Face Mask Dataset consists of images with mask and non-mask and located in directories Mask and Non-Mask respectively. These directories act as two classes necessary for our project. We use a Cifar-10 dataset for third class i.e "Non-Human". Since, Cifar-10 dataset are grouped in 10 different class folders, we take 600 samples randomly from each class folder and store them in a single folder. We categorise the output folder as Non-Human class for our experiment. The statistics and structure about the final dataset use for our Experiment are mentioned in Table 1.

We perform pre-processing step in training and testing data before feeding into Convolutional Neural Networks. The pre-processing step include resizing, normalising, centrecrop. All these data is converted to tensor data. Then we normalize these tensor data with mean = [0.485, 0.456, 0.406] and standard deviation = [0.229, 0.224, 0.225]. We resize the each data to size of 256*256.

Info	**Face Mask Dataset**	**Cifar-10 Datasets**
Name	Face Mask ~12K Images Dataset	CIFAR-10 PNGs in folders
Author	Ashish Jangra	Swaroop Kumar and [4]
Source	https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset	https://www.kaggle.com/swaroopkml/cifar10-pngs-in-folders
Licence	No license specified, the work may be protected by copyright.	No license specified, the work may be protected by copyright.
Total Size:	12000 images	60000 images
Image per class	With Mask: 6000 Images Without Mask: 6000 Images	10 classes, with 600 images per class
Training Size	With Mask: 5000 Images Without Mask: 5000 Images	NonHuman: 5000 Images
Testing Size	With Mask: 483 Images Without Mask: 509 Images	NonHuman: 500 Images

Table 1: Statistics and structure about the final dataset

1.2 CNN Architecture

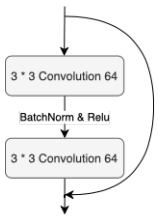

Figure 1: building blocks

In this project, we design a deep neural network to train, test the dataset and evaluate the results. To design our model we consider common issues in deep network such as overfitting and vanishing gradients and try to solve these problems efficiently. In this regard, to overcome the over-fitting problem which occurs in a network with lots of parameters, we decide to have a deeper network with more layers and we apply dropout strategy. For resolving vanishing gradient, we apply the Relu function as activation function. Also we use batch normalisation as another solution for vanishing gradients. Another efficient procedure is the idea of Residual network, so we supply the residual connections straight to earlier layers.

Generally we get the idea of our project from famous Resnet Algorithm [3] and [5]. According to [3], with Residual networks, deeper network has less complexity and shows improvement at analysing. This model consists of building blocks of convolution layers, which is shown in figure 1. These blocks are stacked together while the main idea is that "identity shortcut connection" changes the output of the block. The final architecture of the model, figure 2 has 15 layers. Although expanding the depth of this network is easy due to the property of the block, we limit our self to 15 layers. To speedup the process of testing and training we need GPU, so we decide to implement our model in the google collaboration [2], actually we access to limited hardware.

For training phase, the model iterates over 4 epochs and in each epoch, it trains model with all the batches in the train dataset. Besides we train famous AlexNet model [4], from torch library [5] with our dataset. Then we test the dataset on this trained models and compare the final results of our model and AlexNet.

1.3 Evaluation

In this phase, we evaluate the performance of our model from various aspects including accuracy, precision, recall, and f1 score. We also use the confusion matrix to visualize the performance of two models. To better understand the performance, we compare our model with one of the well-known models in the field of deep learning, namely the ALexNet model. By comparing these two models and the results obtained from them, we find that our model performs very close to the AlexNet model. This shows that the structure of our model is at least as good as the structure of the AlexNet model.

Metric	Our Defined Model	AlexNet Model
Accuracy	95.84	97.72
Precision Micro	95.84	97.72
Precision Macro	96.15	97.75
Recall Micro	95.84	97.72
Recall Macro	95.74	97.71
F1 Score Micro	95.84	97.72
F1 Score Macro	95.80	97.70

Table 2: Evaluation Table

As there are three different classes in the dataset, we consider micro and macro averaging for recall, precision, and f1 score. As shown in the table 2, we obtain very high accuracy, precision, recall, and f1 score from testing phase. Also, it can be seen in the confusion matrix that the model predicts the category of images well. However, our model has a little difficulty in recognizing "with masked" images. Hence, we can conclude that our data are well balanced and our model designed well.

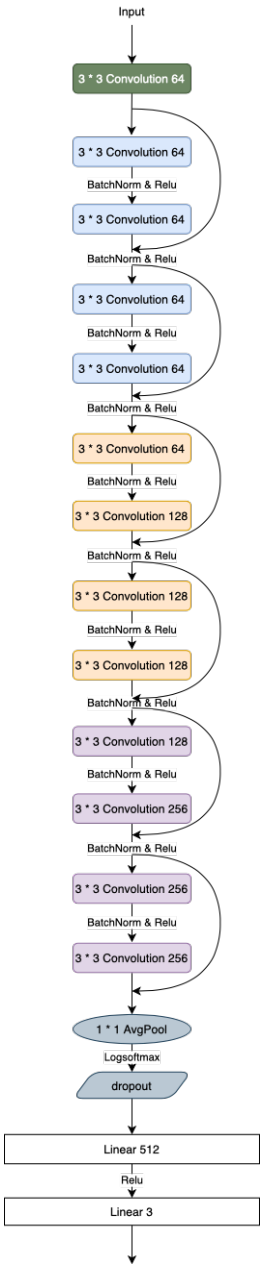

Figure 2: model architecture

In this project, we combined two datasets to have 3 different classes. The Cifar-10 dataset contains non-human images with low resolution, but the other dataset includes high-resolution "with mask" and "without mask" images. Although our model predicts non-human classes well, we can further improve this part and find a dataset for this class with the high-resolution images in the next step to make sure some important features are not missing during training due to low resolution.

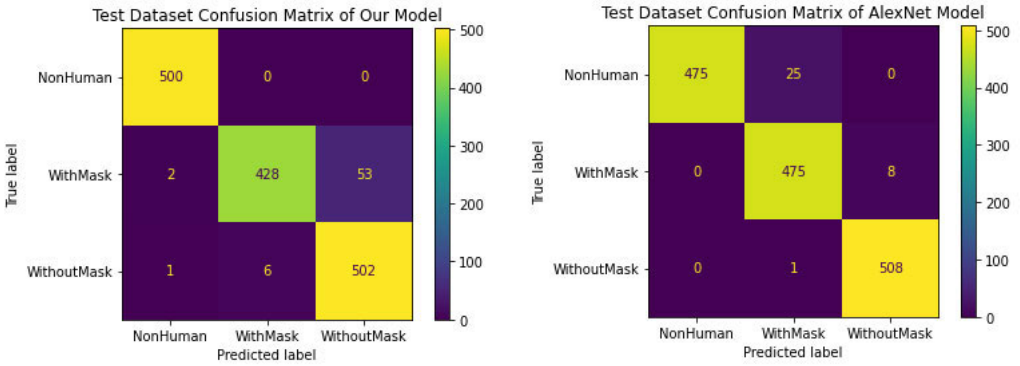

Figure 3: Confusion Matrix of two Models

In the second phase of the project, we can increase the size of the dataset and find appropriate dataset containing non-human images with high resolution. Also we can utilize cross validation technique to make sure that data is balanced across training and test sets. Currently, we consider 4 epochs in this phase. However, we can increase the number of epochs so the training set has more opportunity to update model parameters and minimises errors. Also, we investigate our time to find out if our model exposes any kind of biases, including age, race, etc., and remove them.

2 Phase 2

2.1 Bias Detection & Elimination

Info	**Face Mask Dataset**	**Cifar-10 Datasets**
Name	Face Mask ~12K Images Dataset	CIFAR-10 PNGs in folders
Author	Ashish Jangra	Swaroop Kumar and [4]
Source	https://www.kaggle.com/ashishjangra27/face-mask-12k-images-dataset	https://www.kaggle.com/swaroopkml/cifar10-pngs-in-folders
Licence	No license specified, the work may be protected by copyright.	No license specified, the work may be protected by copyright.
Total Size:	1315 images	561 images
Training Size	With Mask: 506 Images Without Mask: 506 Images	NonHuman: 407 Images
Testing Size	With Mask: 154 Images Without Mask: 149 Images	NonHuman: 154 Images

Table 3: Statistics and structure about the phase 2 dataset

Since the depth of our network is high and the dataset is somehow large, we notice that executing k-fold cross-validation for 10 folds takes more than 10 hours. Beside, we use Google Colab [1] platform for executing the project, due to limitation in accessing to the GPU. This platform restricts our access after a while. Finally we decide to reduce depth of the model and decrease the size of

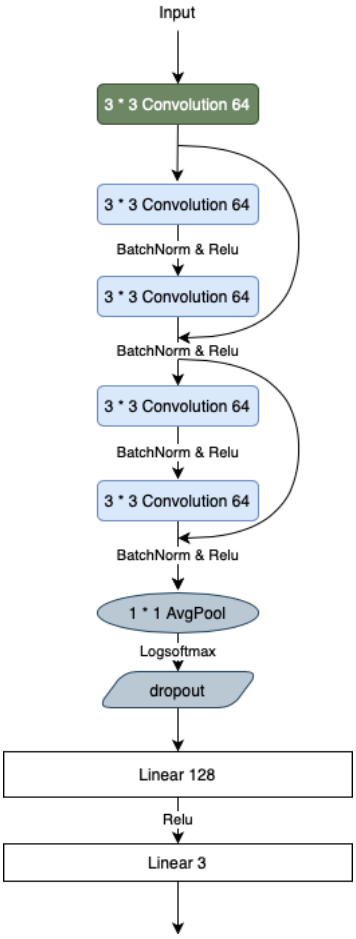

Figure 4: modified model architecture

the dataset. As a result, we predict that performance of the model would reduce. The statistics and structure about the phase 2 dataset are mentioned in table 3.

Figure 4 depicts the architecture of the model after modification. It has 7 layers, the network still keeps the idea of Residual networks and has residual connections. It should also be noted that for the second phase of the project, we increase the number of epochs by two due to getting better results.

Understanding algorithmic bias in AI-based systems has become an important issue. It is important to distinct out possible biases and lack of fairness in the performance of AI applications before deployment. The majority of the architectures report and analyze the effects of biases on their results. Many factors can cause biases, such as inequality of number of data in each category, the human factor since they are responsible to gather data, the quality of the data, etc. AI systems can take biases on various forms like gender, age and race biases.

In this phase, we focus on analyzing gender bias in our existing trained model. We annotate the testing data in two gender and perform bias analysis for the three classes (with mask/ without mask/ non-human). The inconsistency on accuracy and other evaluation factors for male and female

faces dataset are recorded in Table 4. It is clear that F1 score of male category is better than female category, so we see that the dataset is biased toward men images. The confusion matrix in figure 5 confirms this. We conclude that main reason for gender bias in our model is due to imbalance nature of training data with regards to male and female faces.

Dataset	Accuracy	Precision micro	Precision macro	Recall micro	Recall macro	F1 micro	F1 macro
Male	74.34	74.22	80.53	74.22	74.22	74.22	74.26
Female	73.45	73.55	79.71	73.55	73.55	73.55	73.24

Table 4: Before balancing

Dataset	Accuracy	Precision micro	Precision macro	Recall micro	Recall macro	F1 micro	F1 macro
Male	78.32	78.22	81.04	78.22	78.22	78.22	77.93
Female	78.32	78.22	81.61	78.22	78.22	78.22	77.72

Table 5: After balancing

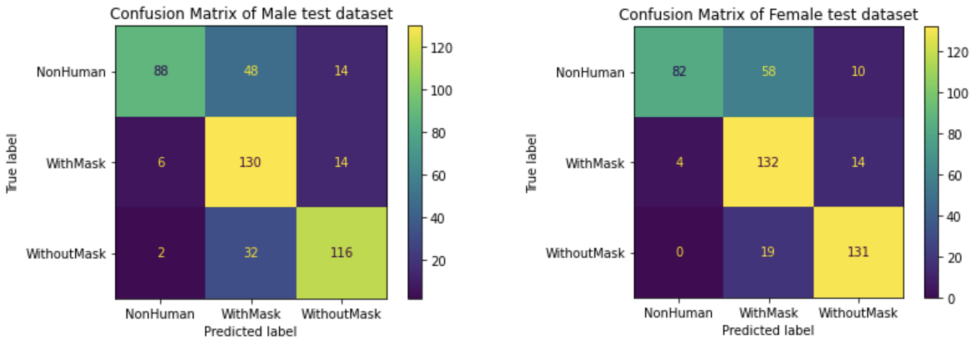

Figure 5: Confusion Matrix of male and female prediction before balancing of dataset

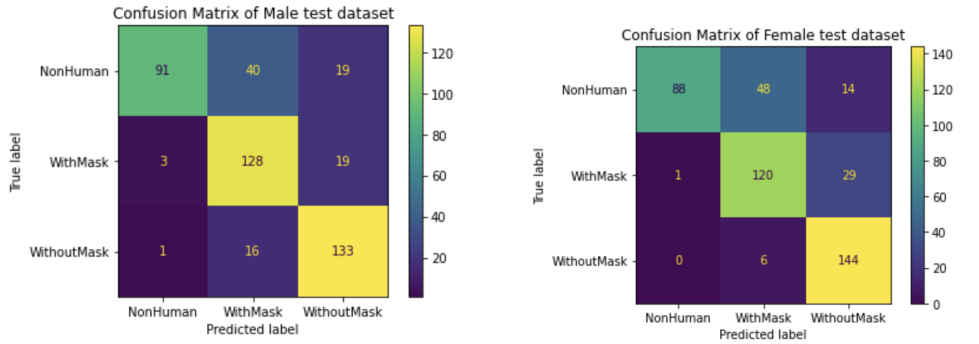

Figure 6: Confusion Matrix of male and female prediction after balancing of dataset

To solve the imbalance dataset issue, we decide to add more female images to train dataset. Then the model is retrained and tested with two male and female test datasets. Results are shown at table 5. According to similar scores, 77.93% macro F1 score for male and 77.72% macro F1 score for female, we deduct that bias is resolved. Also, confusion matrix in figure 6 shows better results after balancing. But we should consider that our train dataset still might be biased toward any other factors, such as race, age. In addition, our dataset might comporise noises including low quality photos, incomplete faces, image rotation, etc.

2.2 k-fold cross-validation

In this phase of the project, first of all, we re-execute phase 1 for small-scale dataset and modified model, for comparison purpuses, table 6 shows the scores. Then, we use the original (small) dataset and rebalanced dataset, to perform k-fold cross-validation with 10 folds on both of them. As it is shown in Figure 7, fold 8 with micro F1 score of 91.44% and macro F1 score of 91.48% has the best performance and fold 3 has the worst F1 score with approximately 79%. Also, we obtain total micro F1 score of 86.09% and macro F1 score of 86.47%, which shows our results are improved by 5% compared to the F1 score we obtain from phase 1. As K-Fold splits the entire dataset into k folds randomly, it leads to less biased model in comparison to fix train/test split, because every single data has the chance to appear in train or test datasets.

Metric	Our Defined Model	AlexNet Model
Accuracy	81.52	91.52
Precision Micro	81.40	91.46
Precision Macro	82.80	91.61
Recall Micro	81.40	91.46
Recall Macro	81.48	91.44
F1 Score Micro	81.40	91.46
F1 Score Macro	81.55	91.49

Table 6: Evaluation Table of phase 1 for the smaller dataset

In the second part of K-Fold, we perform cross-validation on rebalanced dataset. As you can see in Figure 8, the best performance is related to the fold one with total micro and macro f1 score of 91.6% approximately and the lowest one is obtained by fold 2 having about 84.55% f1 score. The aggregation result is near to 89% which shows that the result is improved by 3% compared to the previous part.

Eventually, we can conclude that if we utilize cross-validation and remove biases from our training dataset, we can improve our performance by 8% relative to phase 1 (using small dataset). In this project, we only consider gender as biases. However, if we consider other biases such as age and race, our results may be improved significantly. Moreover, by adding more data, we can have better performance.

References

[1] Google colaboratory.

[2] Ekaba Bisong. *Google Colaboratory*, pages 59–64. Apress, Berkeley, CA, 2019.

	Fold	Accuracy	Precision_Micro	Precision_Macro	Recall_Micro	Recall_Macro	F1_Score_Micro	F1_Score_Macro
0	Fold0	81.910	81.910	82.590	81.910	81.850	81.910	81.910
1	Fold1	82.980	82.980	87.070	82.980	82.310	82.980	83.360
2	Fold2	90.430	90.430	91.130	90.430	90.110	90.430	90.530
3	Fold3	79.260	79.260	83.860	79.260	78.650	79.260	79.770
4	Fold4	86.170	86.170	88.530	86.170	87.410	86.170	87.330
5	Fold5	86.170	86.170	88.920	86.170	85.460	86.170	86.330
6	Fold6	86.170	86.100	87.490	86.100	86.770	86.100	87.030
7	Fold7	87.770	87.700	89.260	87.700	88.330	87.700	88.020
8	Fold8	91.490	91.440	91.590	91.440	91.490	91.440	91.480
9	Fold9	88.830	88.770	89.350	88.770	88.830	88.770	88.930
10	Aggregation	86.118	86.093	87.979	86.093	86.121	86.093	86.469

Figure 7: Result of K-Fold cross-validation for the smaller dataset

	Fold	Accuracy	Precision_Micro	Precision_Macro	Recall_Micro	Recall_Macro	F1_Score_Micro	F1_Score_Macro
0	Fold0	89.060	89.060	90.320	89.060	89.210	89.060	89.300
1	Fold1	91.150	91.150	91.330	91.150	90.880	91.150	91.050
2	Fold2	83.850	83.850	87.850	83.850	83.700	83.850	84.550
3	Fold3	91.670	91.620	91.870	91.620	91.530	91.620	91.320
4	Fold4	89.580	89.530	89.730	89.530	89.880	89.530	89.770
5	Fold5	89.060	89.010	90.290	89.010	88.830	89.010	89.360
6	Fold6	84.380	84.290	85.650	84.290	85.410	84.290	84.950
7	Fold7	88.020	87.960	88.020	87.960	89.100	87.960	88.440
8	Fold8	91.670	91.620	91.740	91.620	91.920	91.620	91.680
9	Fold9	90.620	90.580	91.680	90.580	90.290	90.580	90.740
10	Aggregation	88.906	88.867	89.848	88.867	89.075	88.867	89.116

Figure 8: Result of K-Fold cross-validation for the rebalanced dataset

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[4] Alex Krizhevsky. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

[5] Sébastien Marcel and Yann Rodriguez. Torchvision the machine-vision package of torch. In *Proceedings of the 18th ACM International Conference on Multimedia*, MM '10, page 1485–1488, New York, NY, USA, 2010. Association for Computing Machinery.