# Applied AI Project: AI-based Face Mask Detector

Maxime Lucienne Gevers (27074808)
Data Specialist, AK_09

Meet Patel (40166409)
Evaluation Specialist, AK_09

Shubham Patel (40171270)
Training Specialist, AK_09

## Abstract

*This project creates an AI-based Face Mask Detector. We have build a CNN model without the use of existing models or libraries in order to understand the fundamental architecture of the model. Furthermore, we have created our own dataset of images to train, and test, our CNN model. In this paper, we first demonstrate and apply Exploratory Data Analysis on our data. Secondly, we explain in detail the architecture of our CNN model. Thirdly, we show the results and findings from the trained CNN model with an evaluation metrics. Lastly, we conclude and provide suggestions for improvement.*

## 1. Introduction

During the COVID-19 pandemic, multiple regulations have been put in place by governments in order to contain the transmission of the virus. See Ismail [4] for a more complete discussion. One of these regulations is the obligation to wear an (appropriate) face mask in public spaces. However, it has been proven difficult to verify if this regulation is being respected and it is even more complicated to verify whether the appropriate face mask is worn, as is shown by Tan et al. [11] and Cheng et al. [1].

In this paper, we propose a solution to this problem by facilitating an AI-based automated facial recognition system that can detect *whether* a person is wearing a face mask and *what kind* of face mask (either 'surgical', 'FFP2'-type, or 'community'). In short, this project aims to detect and distinguish whether a person is (i) not wearing a face mask, (ii) is wearing a "community" (cloth) face mask, (iii) a "surgical" (procedural) mask, or (iv) an "FFP2"-type face mask. We do not consider any other type of face masks, such as 'N95', nor images that simply do not represent a face.

In Section 2, we will explain how we obtained the data and apply exploratory data analysis (EDA). In Section 3, we will discuss the Convolutional Neural Network (CNN) model and explain the general CNN architecture, as how we implied the model in PyTorch [7]. Noteably, our implementation of the CNN model solely relies on standard libraries, as well as Skorch [10] and scikit learn [8]. We will motivate the specifics of the CNN architecture. Finally, in Section 4, we will demonstrate our results and discuss our findings.

## 2. Data

As previously described, our AI-based Face Mask Detector aims to detect whether a person is wearing a certain type of face mask, or not face mask at all. In order to properly classify a new image in one of the four categories, it is crucial that the CNN model is trained and tested on adequate data. We aimed to have at least 1,000 training images and 300 test images for each of the four categories. Furthermore, even though the project did not specify the need for a validation set, we did aim to also include validation. Based on common practice in machine learning, the inclusion of a validation set will most likely result in a better model.

The main challenge for gathering the adequate data is to make sure that all training and testing images for each class *actually* only contain data for that class. Most publicly available data contain two datasets, namely a dataset containing images where people are wearing a face mask and a dataset where people are not wearing a fask mask. However, close analysis shows that the dataset with people wearing a fask mask contains all sort of face masks, such as FFP2, FFP3, KN95, N95, cloth masks, etc. Hence, we spend considerable time to gather the necessary data and we used publicly available data, as well as creating our own dataset. Another important observation is that the images should *only* show one face, meaning that we did not include any training and testing images representing several faces. The image must only depicted a *single* face of a person wearing a (certain type of) mask or not. Lastly, we noticed that background has a great influence on the performance of our model. In other words, images that included certain objects or surroundings in the background of the images had a negative effect on our model training. Hence, we had to make sure that background noise would be cancelled out which we achieved by cropping our images such that only the face is visible in the image (with some minor background elements).

### 2.1. Publicly available datasets

We made use of several publicly available datasets found on Kaggle [5], Github, namely UTKFace [12] and MaskTheFace [6], on a website called humans in the in the loop [3], and we created our own dataset which we demonstrate in Section 2.2. It is important to notice that much of the images in the datasets are synthetic images. In other words, 'real' images with a single face are used to put a generated (i.e. artificial) mask on the face, meaning that the image is not a 'real' image of a person wearing a mask. To describe in more detail the datasets that we used, we will describe each dataset in full.

- website called Humans in the in the loop [3]

Humans in the Loop is an organisation that published an open-access dataset annotated as a contribution to the worldwide fight against COVID-19. In total, the dataset consists of 6,000 images from the public domain with an extreme attention to diversity, meaning that the dataset features people of different kind of ethnicities, ages, and regions. We used the images labeled with 'surgical' to update our data with images of people wearing a surgical mask. See Figure 2 as a general example of our data that is labeled 'surgical'.

Figure 1. A general example of our data that is labeled 'surgical'

- Kaggle [5] Face Mask Detection Dataset

This particular datadset consists of 7553 RGB images in 2 folders as 'withmask' and 'withoutmask', and are labelled accordingly. There are 3,725 of faces with mask and 3,828 of faces without a mask. Across both folders, the author has taken 1,776 images from Github [2] Prajna Bhandary's Github account, whereas the remaining 5,777 images are collected and filtered from Google search engine. The data labeled 'withmask' contained several different types of masks, so we filtered out a subsut of images of people wearing solely a surgical mask. We used this dataset to obtain our images that we labeled 'surgical'. See Figure 1 as a general example of our data that is labeled 'surgical'.

Figure 2. A general example of our data that is labeled 'surgical'

- MaskTheFace [6]

This is a Github repository with a computer vision-based script to mask faces in images. Thus, the generated data is synthetic data of images with people wearing a certain type of face mask. The script supports five different type of masks: (i) surgical, (ii) N95 or FFP2, (iii) KN95, (iv) cloth, and (v) gas mask. For our project, we solely used images with generated surgical mask and cloth mask. See Figure 3 as a general example of our data that is labeled 'FFP2' and 'cloth'.

Figure 3. A general example of our data that is labeled 'FFP2' and 'cloth'

- UTKFace [12]

We used the UTKFace dataset in order to obtain the faces without mask and also to use these images as input

for the MaskTheFace script. The advantage of the UTK-Face dataset is that this large-scale face dataset has tried to remove certain biases in the dataset. More concretely, the dataset consists of over 20,000 face images with annotations of (i) age (range from 0 to 116 years old), (ii) gender, and (iii) ethnicity. Furthermore, the images cover large variation in pose, facial expression, illumination, occlusion, resolution, etc. See Figure 4 as a general example of our data that is labeled 'without'.

Figure 4. A general example of our data that is labeled 'without' mask

## 2.2. Created dataset

While considering publicly available datasets, the class cloth mask and FFP2 mask still had insufficient training images, and moreover, we wanted to have a more heterogeneous dataset. The synthetic data with generated cloth and FFP2 masks had an uniformerly visual representation. Hence, we decided to created our own dataset by crawling the web. We used Python for Image Scraping and download images from Google[1] satisfying the characteristics of a cloth mask and FFP2 mask.[2] With image scraping techniques, we downloaded 200 images for several research queries on google. [3] Examples of research terms are: 'person wearing FFP2 face mask', 'person wearing cloth face mask', 'silk cloth face mask', or, 'cotton cloth face mask'. Envisioning the second phase of our project, we already wanted to have included training images that better represent the population. Thus, we included search terms such as 'child wearing cloth face mask', 'hijab with face mask', 'asian person wearing FFP2 face mask'.

An advantage of Image Scrapping is that we obtained qualitative better images, but the disadvantage is that each image has a different size, i.e. the images are not equally sized, as well as having much background noise. Thus, we needed to apply cropping methods such that the images obtained by scrapping where removed from background noise and have an equal size.

[1] in Phase 2, we would like to look at other websites, such as https://unsplash.com//s/photos/cloth-mask-person

[2] See the Jupyter Notebook for Image Scraping with Google

[3] with a reference to https://towardsdatascience.com/image-scraping-with-python-a96feda8af2d

Table 1. Split of data

| training set | validation set | test set | total |
|---|---|---|---|
| 3024 | 594 | 337 | 3955 |
| 75% | 15% | 10% | 100% |

Table 2. Size of each category or label in the dataset

| cloth | FFP2 | surgical | without | total |
|---|---|---|---|---|
| 1,015 | 1,001 | 939 | 1,000 | **3955** |

Figure 5. A general example of our data obtained by Image Scrapping of Google that is labeled 'FFP2' and 'cloth'

## 2.3. Exploratory Data Analysis

In this section, we will give an overview of the specifics of our data. As previously mentioned, the project did not demand that we have a validation set, however, we decided to include a validation set. This means that we split our data in (i) training set, which is 75 percent, (ii) validation set, which is 15 percent, and (iii) test set, which is 10 per cent of the total data. See Table 1 for the specifics.

In addition, we did not want to have a imbalance between the size of different the subsets of our data with different labels. In other words, the size of the images labeled 'surgical' should approximately equal those labeled 'FFP2'. Table 2 shows the sizes of each subset with a certain label.

## 2.4. Pre-processing

- Cropping

It is very important for model training that the training data (or images) have the same size. We obtained equally sized data from most of the publicly available data, meaning that we did not need to apply cropping techniques or a bounding box. However, for the images obtained by image scrapping and some of the data from publicly available data did

not adhere to this standard. For this purpose, we applied cropping techniques to obtain equally sized images and we aimed to crop the image in such a way that we solely obtained the face of one person. In addition, for the images that had several faces displayed, we used a bounding box to obtain images with only one face. It must be noted that we did not fulfill this requirement completely, but we will in the second phase of this project. In the preprocessing phase, we resized the images to a image of 200x200.

- Normalization

We applied normalization to all images in our dataset. This means that each images has been subtracted by the mean and divided by the standard deviation.

## 3. CNN Architecture

### 3.1. Layers

There are mainly three layers that make up our CNN model:

- Convolutional layers: Convolutional layers can be used to extract specific features from the image. Each convolutional layer learns specific features of image.
- 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.
- Fully connected layers: These layers use the features extracted by the convolutional layers and use the extracted features for decision making.

### 3.2. Architecture

VGG16[9] model configuration has been used as the architecture for out model. The model contains 16 layers in total, specifically 13 convolutional layers and 3 fully connected layers.
Each convolutional layer as well as the fully connected layer have activation relu, as relu does not activate all neurons at the same time, neurons with transformation output of less than zero wouldn't be activate, so during backpropogation process not all neurons will observe a update in weights and bias, due to which relu turns out to be less computationally expensive as compared to other activation functions.

The first layer takes an input of size 224x224 and has 64 kernels of size 3x3 and padding of 2 so the output size after this layer is 224x224x64.

The second layer takes an input of size 224x224 and has 64 kernels of size 3x3 and padding of 2 so the output size after this layer is 224x224x64. After second layer max pool of kernel size 2x2 and stride of 2 is applied which reduces the image dimensions to 112x112x64.

Similarly third and fourth convolutional layers of 128 kernels of size 3x3 would result in 114x114x128 image size, after which max pool of kernel size 2x2 and stride of 2 is applied which reduces the image dimensions to 56x56x128.

The fifth through seventh convolutional layers of 256 kernels of size 3x3 would result in 56x56x256 image size, after which max pool of kernel size 2x2 and stride of 2 is applied which reduces the image dimensions to 28x28x256.

The eights through tenth convolutional layers of 512 kernels of size 3x3 would result in 28x28x512 image size, after which max pool of kernel size 2x2 and stride of 2 is applied which reduces the image dimensions to 14x14x512.

Finally the last three convolutional layers of 512 kernels of size 3x3 would result in 1414x512 image size, which is further reduced by max pool layer which reduces it to 7x7x512.

Once all the convolutional layers are done the 14th and 15th layers are fully connected linear layers which takes an input of 25088 which would output a vector of size 4096.

Output of 15th layer is passed to the 16th layer which is the final layer which outputs an array of size 4(which is the number of classes to predict from), output label can be found by using argmax on the output of this layer.

### 3.3. Hyperparameters

- Optimizer: We have used Adam optimizer for training, as adam optimizer computes the learning rate for some of the parameters, it converges much more faster as compared to stochastic gradient descent.
- Learning Rate: Learning rate of 0.01 is used for training. It is the most commonly used learning rate when using adam optimizer, we tried other values too but 0.01 converges to global minimum.
- Early Stopping: Early stopping is used to stop the model for training when it starts to overfit on the training data. Early stopping of 3 is used which stops training when validation loss doesn't decrease for three epochs.
- Epochs: The model is trained on 30 epochs, but due to early stopping the model stops training after 18 epochs.
- Loss: Cross entropy loss is used to compare the model output with the true label, it's value lies between 0 and 1, higher value indicates that the model output diverges from the true value.
- Scheduler: OneCycleLR is used to vary learning rate from maximum value to minimum value, first the

learning rate is increased from minimum value to maximum value thereafter it is decreased to minimum value. Here the maximum learning rate is chosen to be 0.01.

In Figure 6 and 7, we show the accuracy and training entropy loss respectively.

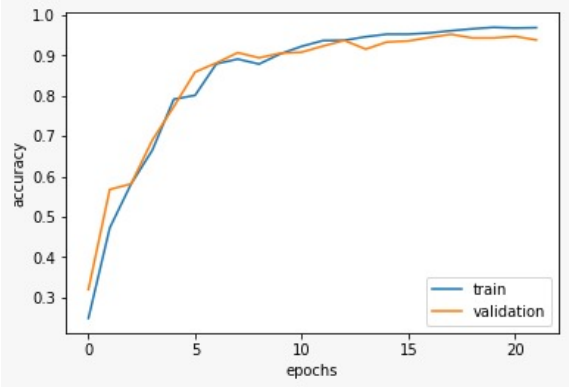


Figure 6. Accuracy

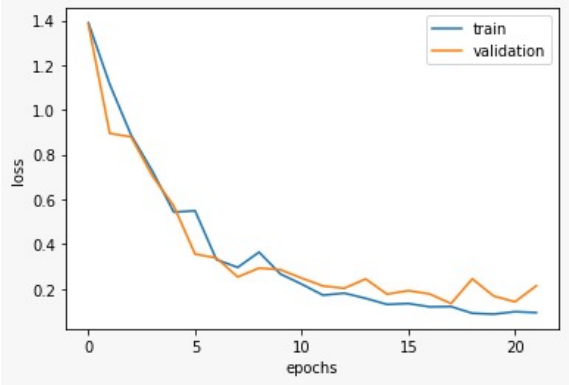


Figure 7. Entropy Loss

# 4. Evaluation

## 4.1. Need for Evaluating the Model

There are a variety of ways for modelling data regularity. Furthermore, learning techniques based on the same collection of examples result in distinct models altering the method's parameters. Because the same problem and training data might generate a greater variety of diverse models, it stresses the need of evaluating the classification model. As a result, one of the most important aspects of the intelligent data analysis process is the analysis of discovered knowledge.

## 4.2. Evaluation metrics

Here, Performance of the proposed system is evaluated by considering the actual and predicted classification. Accuracy of the system is calculated by using the confusion matrix obtained for the classifier used. We have used precision, recall, F1-score and accuracy to analyze the performance of our model. To calculate all these metrics, we need to have true labels and predicted labels. For precision, we calculate the numbers of instances which are correctly classified dividing by the number of predicated labels. For recall, we calculate the numbers of instances which are correctly classified dividing by the number of true labels. Harmonic mean of precision and recall is F1-score. In total we have about 4000 images in our dataset, and we split the dataset into 75 percent for training, 15 percent for validation and 10 percent for testing our model. Split of the dataset is automated splitting. We have 4 label classes: 0-cloth mask, 1-FFP2 mask, 2-surgical mask and 3-without mask.

## 4.3. Performance of Model

Below is the confusion matrix for four class classifier.

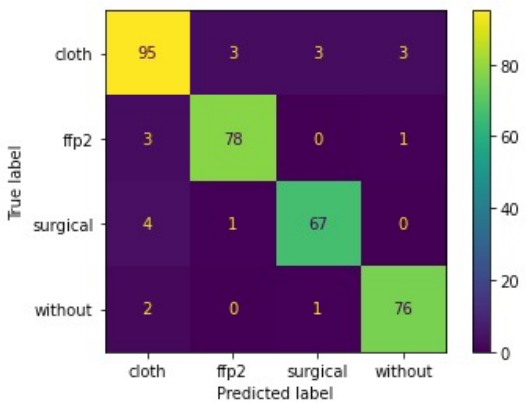


Figure 8. Confusion Matrix

| Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| cloth mask | 0.91 | 0.91 | 0.91 | 104 |
| FFP2 mask | 0.95 | 0.95 | 0.95 | 82 |
| surgical mask | 0.94 | 0.93 | 0.94 | 72 |
| without mask | 0.95 | 0.96 | 0.96 | 79 |
| Accuracy | - | - | 0.94 | 337 |
| Macro Average | 0.94 | 0.94 | 0.94 | 337 |
| Weighted Average | 0.94 | 0.94 | 0.94 | 337 |

Table 3. Classification Report

Average precision of our model is 0.94 and across all classes, it is equal to or greater than 0.90 and for class 1-ffp2 and class 3-without mask, it is maximum (0.95). Average recall of our model is 0.94 and for class 3-without mask,

it is maximum (0.96). Average F1-score of our model is 0.94. Accuracy of model is 0.94 and from all these, we can conclude that all parameters evaluated shows equally good results.

### 4.4. Conclusion and Improvements

This model performs well at most of the cases but at some extent, images having background noise creates problem for classifying the exact class label. Also, in few cases, it mis-classify between cloth mask and ffp2 mask due to bias based on colour and shape of the mask. In next phase of the project, we will try to eliminate all these biasing which makes our model imperfect for some sort of images.

## References


[1] V. C.-C. Cheng, S.-C. Wong, V. W.-M. Chuang, S. Y.-C. So, J. H.-K. Chen, S. Sridhar, K. K.-W. To, J. F.-W. Chan, I. F.-N. Hung, P.-L. Ho, et al. The role of community-wide wearing of face mask for control of coronavirus disease 2019 (covid-19) epidemic due to sars-cov-2. *Journal of Infection*, 81(1):107–114, 2020. 1

[2] P. B. Github. Prajna bhandary's github account. https://github.com/prajnasb/observations. 2

[3] H. in the loop. Medical mask dataset. https://humansintheloop.org/resources/datasets/medical-mask-dataset/. 2

[4] N. Ismail. The dynamics of government policy in handling corona virus disease 2019. *Jurnal Hukum Volkgeist*, 4(2):158–165, 2020. 1

[5] Kaggle. Face mask dataset on kaggle. https://www.kaggle.com/omkargurav/face-mask-dataset. 2

[6] MaskTheFace. Masktheface github aqeelanwar. https://github.com/aqeelanwar/MaskTheFace. 2

[7] PyTorch. Python library. https://pytorch.org/. 1

[8] scikit learn. scikit-learn library. https://scikit-learn.org/stable/. 1

[9] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 4

[10] Skorch. Skorch library. https://github.com/skorch-dev/skorch. 1

[11] M. Tan, Y. Wang, L. Luo, and J. Hu. How the public used face masks in china during the coronavirus disease pandemic: A survey study. *International journal of nursing studies*, 115:103853, 2021. 1

[12] UTKFace. Github utkface large scale face dataset. https://susanqq.github.io/UTKFace/. 2