Proper Usage of Face Masks: Automated Detection Using Artificially Generated Dataset and Deep Learning Methods

Mahesh Dhaka
Department of Computer Science
Lakehead University
Thunder Bay, Canada
mdhaka@lakeheadu.ca

Quazi Abidur Rahman
Department of Computer Science
Lakehead University
Thunder Bay, Canada
quazi.rahman@lakeheadu.ca

***Abstract*—In the wake of Covid-19, an increasing number of individuals are not following the face mask guidelines issued by authorities. The research paper focuses on detecting whether people are wearing face masks properly or not. The experiment uses three different datasets to train a Convolution Neural Network model for the classification of Correctly and Incorrectly worn face masks. The CNN model performs a total of 5 different experiments on various combinations of data obtained from these datasets. The experiments also use Generative Adversarial Networks to generate more images to deal with the issue of class imbalance. The first three experiments involve applying a CNN model on these datasets one by one, where only one dataset got used for each experiment. In the fourth experiment, all three datasets got merged and, a CNN model got applied to them. In the last experiment, along with merging all three datasets, a GAN model got used to resolve the class imbalance issue. There is a plethora of research on face mask detection, recognition, or even on detection whether people are wearing face masks or not but this research is unique in a way that it focuses on whether subjects are wearing face masks correctly or not.**

***Index Terms*—Correctly Masked Faces, Incorrectly Masked Faces, MTCNN, GAN, Face Mask, Masked Face Detection**

I. Introduction

An increasing number of people are not following the face mask guidelines issued by authorities in wake of Covid-19. Some individuals do not put on the masks at all, while others do not cover their mouth, nose, or chin appropriately. The most common way of incorrectly wearing a face mask is to keep the nose out of the mask and wear it on the rest of the face. It can be a tedious task for authorities to monitor such individuals. As the evidence suggests that healthy individuals may get infected by coming in close contact with Covid-19 patients. Not following the mask guidelines appropriately in confined spaces is one of the major contributors to the increasing number of Covid-19 cases.

The research paper attempts to solve this issue by using a Convolution Neural Network model. The model gets trained on a total of 3 different datasets which contains images of correctly and incorrectly worn face masks. Some of the images in the experiment are artificially generated by dynamically putting a face mask on people's faces. To achieve that the experiment uses MTCNNN face detection model for detecting facial key-points and thereafter puts different types and colors of masks on these faces. The experiment also uses some real-world masked face images to train the model.

II. Literature Review

Face detection is an active area of research for the last decade. A plethora of traditional and modern face detection techniques are available currently. One of the traditionally used face detection techniques is Viola-Jones Face Detection Algorithm[8]. The detection technique was introduced in the year of 2001 and is used for object detection in real-time although still primarily focuses on face detection. The technique has one of the lowest numbers of false-positive cases among traditional object detection methods. It has four main components: Harr feature selection, Creating an integral image, Adaboost training, and Cascading classifier. The most popular modern-day face detection techniques are based on Convolution Neural Network [1].

One of the most effective and currently widely used face detection techniques is Multi-task Cascaded Convolutional Networks (MTCNN)[5]. The researchers created a dataset named Driver FACE dataset with low resolution 2,845 images consisting of 5,171 human faces. MTCNN model achieved precision, recall, and F-score of 87%, 94%, and 91% respectively [2].

There is also some research done on face mask detection on a human face. The researchers in this paper focus on detecting whether an individual is wearing a face mask or not [9]. The paper proposes a hybrid model using deep learning and classical machine learning methods. The model consists of two components: The first component is used for feature extraction using Restnet50. And the second component is used for the classification of a masked face image. This component uses decision trees, Support Vector Machine, and ensemble algorithm to classify images. The experiment uses three different datasets: Real-World Masked Face Dataset, Simulated Masked Face Dataset, and the Labeled Faces in the Wild Dataset and it achieved a classification accuracy of 99.64%, 99.49%,100% respectively using SVM.

In another research conducted by researchers on detecting whether people are wearing face masks or not, researchers apply ResNet50, MobileNet, and GoogleNet on two different masked face datasets [10]. The first dataset has a total of 3833 images among which 1918 are without a face mask and the other 1915 images are with a face mask. And the second dataset consists of a total of 1650 images where 824 images are without a face mask and 826 images are with a face mask. The research performs a comparative analysis of various deep learning models on these datasets. The researchers claimed that MobileNet with a global pooling block for face mask detection outperforms other deep learning classification models. The purposed model achieved 99% classification accuracy on Dataset1 and 100% classification accuracy on the Dataset2.

III. Dataset Description

The research uses three different datasets for the experiment. The detailed information about these datasets can be view below:

The first dataset is obtained from visual.io. It is a data-sharing platform for computer vision research. The dataset got made available on the platform by Halim Benhabiles and Mahmoud Melkemi, individual AI researchers [4].

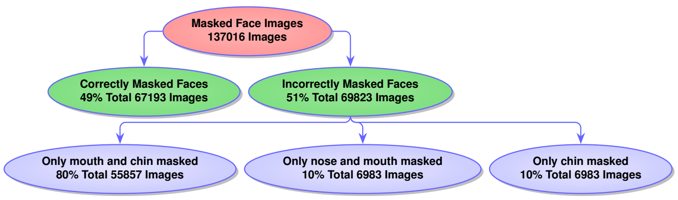

Fig. 1. Dataset1 Image Distribution By Category

Fig. 2. Dataset1 Sample Images

It contains 1,37,016 images of masked faces that are generated using the Generative Adversarial Network (GAN). Images can be divided into two categories: Correctly Masked Faces (CMF) and Incorrectly Masked Faces(IMF) with a total number of 67,193 and 69,823 images respectively. IMF image category can be further divided into three subcategories: only mouth and chin masked, only nose and mouth masked, and only chin masked with 55,857, 6,983, and 6,983 images respectively. All the images in the dataset have dimension of 1024×1024. The dataset consist images of individuals with different race, gender, and ethnic background. Despite all the benefits of this dataset, it has some drawbacks. One of the major drawbacks is that all the subjects are front-facing in the image. So, the model may not work for facial images with angels. Another issue with the dataset is that it covers only medical face masks. And lastly, images in the dataset are artificially generated.

The role of Dataset2 is to overcome some of the drawbacks of Dataset1. It contains images from real-world settings and got obtained from Kaggle [11]. Originally, it was made available for face mask detection, but the model would use it for correctly versus incorrectly wore face mask detection. The dataset has a total of 853 images with each image containing several faces. Some of the faces are too small to be used for the experiment. Considering image that has the size of 16X16, there are in total 2800 images.

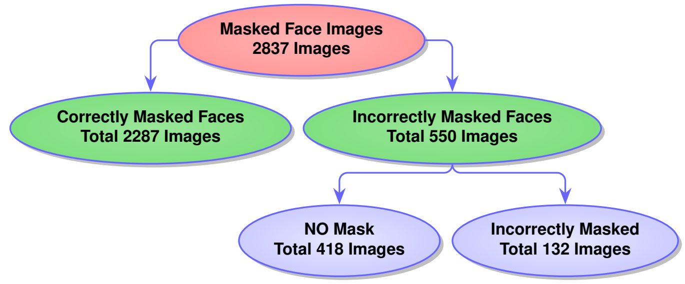

Fig. 3. Dataset2 Image Distribution By Category

Fig. 4. Dataset2 Sample Images

The dataset can be divided into three categories: Correctly Masked Faces, Incorrectly Masked Faces and No Mask Faces with 2287, 95, and 418 images respectively. As we can see the category of Incorrectly Masked Faces just contains 95 images, another 37 images of incorrectly masked faces got obtained from the internet. Because the category of No Mask Faces is also part of Incorrectly Masked Faces, it can also be categorized into two categories: Correctly Masked Faces and Incorrectly Masked Faces with 2287 and 550 images respectively with a total of 2837 images. One of the main benefits of Dataset2 is that it consists of images from the real world and includes faces with different angles and gender.

The role of the Dataset3 is to overcome a few drawbacks of Dataset1 that even Dataset2 cannot resolve. It got artificially prepared by putting different types and colors of face masks on people's faces using MTCNN [5]. Subjects for the experiment are obtained from the Surveillance Cameras Face Database [6]. It consists of 120 subjects with different facial angles for example Front, left, right, and other angels in between. The experiment uses the MTCNN model to detect facial key points and thereafter puts different types of face masks on people's faces dynamically. MTCNN model cannot detect faces all the time so it could only use 79 subjects efficiently. Overall, this method generated a total of 5935 images. These images can be divided into two categories: 2844 Correctly Masked Faces and 3091 Incorrectly Masked Faces. The latter category has images with four different subcategories: Only Mouth and Chin Masked, Only Nose and Mouth Masked, Only Chin Masked, and No Face Mask with 948, 948, 948, and 237 images respectively.

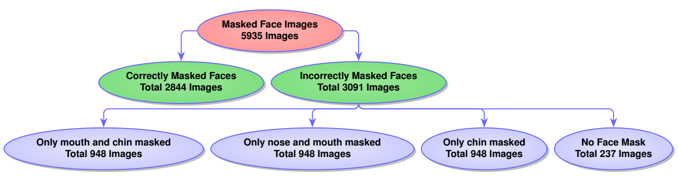

Fig. 5. Dataset3 Image Distribution By Category

Dataset3 solves the issue of angle by including faces with different angles. It also resolves the concern of having only one type of face mask in Dataset1 by imposing various types and colors of face masks on subject's faces.

Fig. 6. Dataset3 Sample Images

IV. Methodology

A. Neural Networks:

Neural Network is a series of algorithms to identify various patterns in data using algorithms that are based on the human brain. The mind consists of millions of neurons that communicate with each other to make a decision. Some of these neurons are in the active phase and others are not. In the same way, neural networks also have active and non-active neurons. There are several types of neural networks that can be applied to a problem depending on its type. The mainly used neural networks are Multilayer Feedforward Neural Network, Convolution Neural Network, and Recurrent Neural Network. The next step would be to understand a Convolution Neural Network. It is a deep learning-based domain-aware neural network that takes an image or text data as an input and assigns weights and bias to the neurons in the network. An image has length, width, and depth properties. The Convolution Neural Network consists of three main operations convolution, max-pooling, and ReLU. Instead of Max pooling using Stride Convolution is becoming widely popular in the last several years. The convolution operation is similar to matrix multiplication in CNN. Filters play a vital role in CNN to identify patterns where filters spatial dimension must not be more than layers spatial dimension [3].

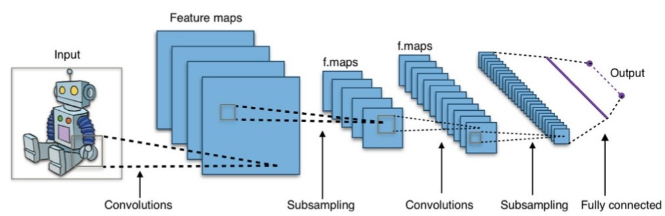

Fig. 7. CNN Model Example (Source:Wikipedia)

B. MTCNN Face Detection:

One of the most effective and currently widely used face detection techniques is Multi-task Cascaded Convolutional Networks (MTCNN)[5]. The model is based on CNN architecture and typically achieves near to a hundred percent accuracy for high-resolution image datasets. The network can be divided into three parts: P-Net, R-Net, and O-Net. P-Net takes a scaled image as an input and runs a 12X12 kernel through the image to search for a face. When the kernel reaches the corner, it shifts downwards by 1px and carries on the same process throughout the image. The output of P-Net is the probability of faces being in each bounding box along with the coordinates of the bounding box. The architecture of R-Net is like P-Net with just more layers. It takes the output of P-Net as an input and returns more accurate bounding box coordinates. In the end, O-Net takes the output of P-Net as an input and returns 3 sets of output: the probability of faces in a bounding box, the coordinates of the bounding box, and the coordinates of the facial landmarks.

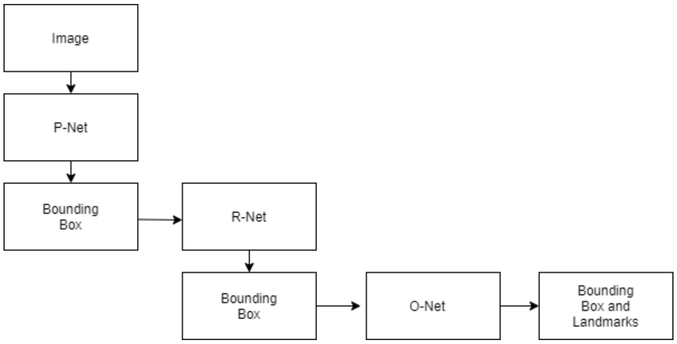

Fig. 8. MTCNN Model Architecture

C. Generative Adversarial Networks:

A CNN model needs a significant amount of data for the training, but sometimes data availability is scarce. In such a scenario, more data can be generated using Generative Adversarial Networks(GAN)[7]. It is a machine learning framework designed by Ian Goodfellow in 2014. The framework simultaneously trains two different models: a generative model and a discriminative model. The first one generates some data using random noise and the second one estimates the probability that the sample came from the training data instead of from the generative model. The training procedure for the generative model is to maximize the probability of the discriminative model making a mistake. The generator model in GAN is a fully connected network that cannot be efficient enough for image data. To resolve that issue, a Deep Convolutional Generative Adversarial Network(DCGAN) can be used. It is a type of GAN model where the model uses transposed convolution technique.

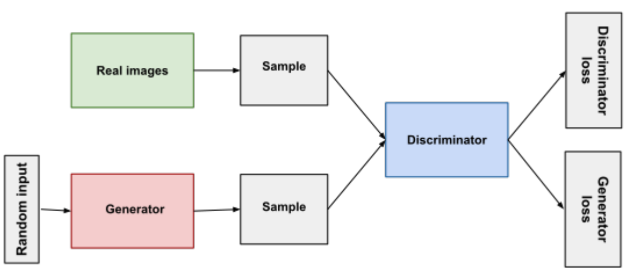

Fig. 9. GAN Model (Credit: Google ML Blog)

D. Data Preprocessing:

Before applying various machine learning models, datasets need to be preprocessed. The data for the experiment is created using three different datasets. Preparation of it can be understood below:

Preparing Dataset1: After obtaining data from the given source, the following steps got performed with it:

- Loading data in two separate arrays, one for correctly masked faces and another one for incorrectly masked faces
- Grayscale conversion
- Image size reduction to 16×16
- Saving data for later use

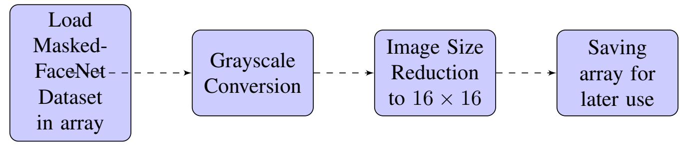

Fig. 10. Dataset1 preparation flowchart diagram

Preparing Dataset2: Dataset2 got prepared using a Kaggle face mask images datasets in a real-world setting. Every image of the Kaggle dataset contains several face images. The following steps got performed to prepared the data:

- Locate and crop faces in the images using coordinates of faces in a given XML file.
- Create two different array faces images, one for correctly masked faces and another for incorrectly masked faces using provided labels in the XML files.
- Reduce image size to 16×16
- Save data for later use

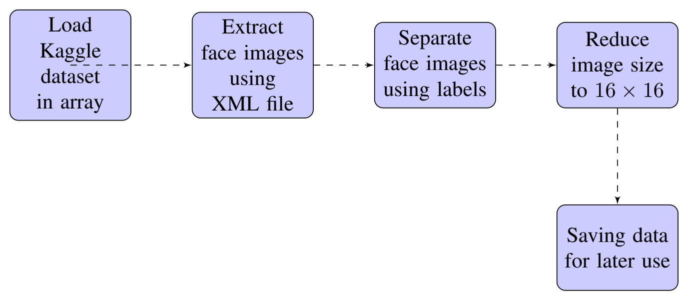

Fig. 11. Dataset2 preparation flowchart diagram

Preparing Dataset3: Dataset3 got prepared using face images obtained from the SCFace dataset[6]. The following steps got performed to create the dataset:

- Detect facial coordinates in images using MTCNN.
- Apply various types of face masks on faces using their facial key points.
- Reduce image size to 16×16
- Saving data for later use

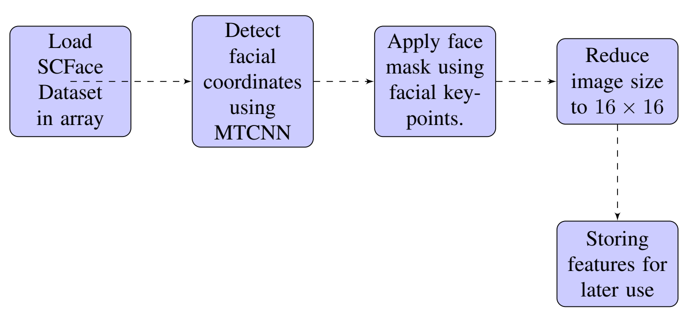

Fig. 12. Dataset3 preparation flowchart diagram

Dynamically Imposing face masks on people's faces is one of the key contributions of this research. It cannot be done manually because around 6000 face mask images need to be prepared for Dataset3. Before putting face masks on the subject's faces, facial key points needs to detected using the state-of-the-art MTCNN face detection model. The model returns a JSON object with three keys: 'box', 'keypoints', and 'confidence'. The first key contains a bounding box coordinate of the face with values x, y, w, and h. The second key has another JSON object with keys 'left_eye', 'right_eye', 'nose', 'mouth_left', 'mouth_right'. Each key has a pixel value with x and y coordinates. And the last key 'confidence' has the probability of a face being found in a bounding box. To put face masks on the subject's faces, the experiment focuses on the value inside the 'keypoints' JSON object.

The experiment extracts the key values of 'left_eye', 'right_eye', 'nose', 'mouth_left', 'mouth_right' from the 'keypoints' JSON object. These values provide the exact facial key-points required to dynamically impose a face mask. For the research, there is a need to prepared four different types of images: Correctly Masked Face, Only Mouth and Chin Masked, Only Nose and Mouth Masked, and Only Chin Masked. The exact placement of a face mask on a face varies depending on the required mask category. For example, to generate Only Chin and Mouth Masked category image, y_{min} for the face mask inside the face image should be less than the 'y' value of nose position in the face image. However, the value of x_{min} and x_{max} for the face mask in the face image does not rely on the type of the image category. If 'facecord' is a returned JSON object from the MTCNN model, x_{min} and x_{max} can be defined in Eq1 and Eq2 respectively.

$$x_{\text{min}} = facecord['keypoints']['left\_eye'][0] - x_{\text{Adjuster}} \quad (1)$$

$$x_{\text{max}} = facecord['keypoints']['right\_eye'][0] + x_{\text{Adjuster}} \quad (2)$$

Where the value of x_{Adjuster} can be defined as follows:

$$\begin{aligned} x_{\text{Adjuster}} = (&facecord['keypoints']['right\_eye'][0] \\ &- facecord['keypoints']['left\_eye'][0])/2 \end{aligned} \quad (3)$$

The value of y_{max} remains same for three out of four masked faces categories: Correctly Masked Faces, Only Mouth and Chin Masked, and Only Chin Masked. It can be calculate as follow:

$$y_{\text{max}} = facecord['keypoints']['nose'][1] + adjustment \quad (4)$$

The value of the 'adjustment' factor is one of the most complex calculations in this entire experiment. It is calculated using a function called 'findym'

$$adjustment = findym(p1, p2, para, nose\_height) \quad (5)$$

where the value of p1,p2 and para can be defined as follows:

$$p1 = facecord['keypoints']['mouth\_left'][1]$$

$$p2 = facecord['keypoints']['nose'][1]$$

$$para = 1$$

The 'findym' function is a recursive function and got used to maximize the value of 'adjustment'. The function maximizes the value in a way that y_{max} should not be more than the maximum height of the image. The value of adjustment should also not be more than 2.8 times the height difference between mouth and nose coordinates.

To calculate the value of y_{max} for Only Nose and Mouth Masked category following equation can be used:

$$y_{\text{max}} = facecord['keypoints']['mouth\_left'][1] \quad (6)$$

However, calculating y_{min} is not that simple and changes from Correctly Masked Faces to Incorrectly Masked Faces and even also within the subcategories of Incorrectly Masked Faces. Let's first calculate the value of y_{min} for Correctly Masked Faces. In this category, the face mask should completely cover the nose and should be below the eyes. So, y_{min} can be defined as:

$$y_{\text{min}} = facecord['keypoints']['left\_eye'][1] + 1 \quad (7)$$

The value of y_{min} for Only Mouth and Chin Masked, Only Nose and Mouth Masked, and Only Chin Masked category can be calculated using the following equations respectively.

$$y_{\text{min}} = facecord['keypoints']['nose'][1] + 4 \quad (8)$$

$$y_{\text{min}} = facecord['keypoints']['left\_eye'][1] + 1 \quad (9)$$

$$y_{\text{min}} = facecord['keypoints']['mouth\_left'][1] \quad (10)$$

Once we have x_{min}, x_{max}, y_{min}, and y_{max}, it can be used to put different types and colors of masks on faces for the respective Masked Faces categories.

The experiment puts face masks on faces with different angels and for that, it needs masks with respective angels. The experiment uses photo-shopped masks with front, left, and right angels to put on faces. The label of an image can provide information about its face angel.

E. Apply CNN Model:

A CNN model got created using Keras and have the following architecture:

- It begins with two 2D Convolution layers
- A flatten layers follow the Convolution layer
- And finally, it ends with three Dense layers
- The last Dense layers of the model use Sigmoid activation function to predict the class, where 0 refers to Correctly Masked Faces and 1 refers to Incorrectly Masked Faces

For training and testing the model, 5-fold cross-validation technique got used where four folds are used for the training and one fold is reserved for testing the model. By default, the model gets trained for 1000 epochs and, early stopping got used by monitoring the training accuracy. The model stops training when classification accuracy reaches above 99.5% or 1000 epochs get completed.

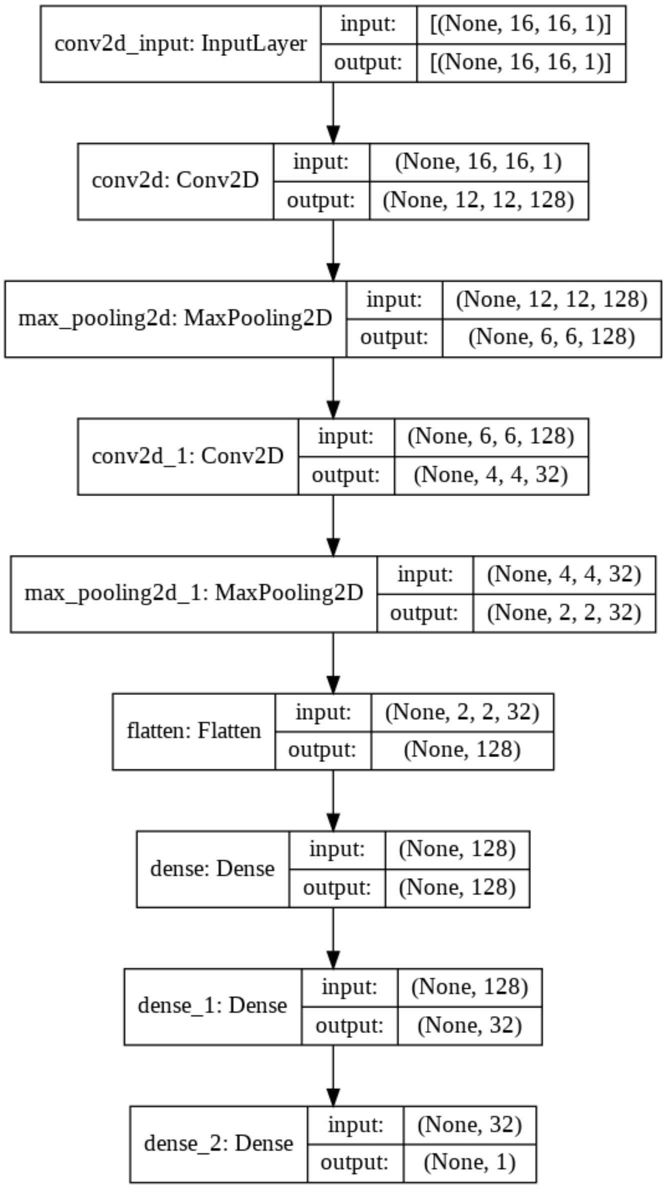

Fig. 13. CNN Model Architecture

V. Experimental Analysis

There is a total of five experiments got performed using the CNN model that uses three different datasets. Some of the experiments use only one dataset while others use a combination of multiple datasets and even uses additional GAN-generated images to deal with the issue of class imbalance. In all the experiments, it was made sure that the mean response value is equal across all the folds. These experiments can be understood below:

Experiment1: The first experiment uses only Dataset1 for the research. It contains images of Correctly Masked Face and Incorrectly Masked Faces with 67,193 and 69,823 images respectively. A CNN model got applied on Dataset1 using 5-fold cross-validation.The performance of the model can be viewed in TABLE I.

Type of Face Masks	Accuracy
Complete Dataset	97.90%
Correctly Masked Accuracy	97.81%
Incorrectly Masked Accuracy(IMF)	97.99%
IMF Only Chin Covered Accuracy	98.48%
IMF Mouth and Chin Covered Accuracy	99.09%
IMF Nose and Mouth Covered Accuracy	87.64%

TABLE I
Model Performance For Dataset1

It can be observed in TABLE I that a CNN model can classify Correctly and Incorrectly wore face masks. Another important piece of information to notice is that accuracy for the Nose and Mouth Covered category is significantly lower than any other category of masked faces. It is because masked faces in this category look almost similar to masked faces in the Correctly Masked Faces category. Another reason can also be that detecting chin feature is not as easy as detecting nose and mouth features in an image.

Experiment2: CNN model got applied on Dataset2 using 5-fold cross-validation. This dataset contains images of real-world people with Correctly and Incorrectly wore face masks with 2287 and 550 images respectively. The classification accuracy of the model can be found in TABLE II.

Type of Face Masks	Accuracy
Complete Dataset	91.85%
Correctly Masked Accuracy	95.58%
Incorrectly Masked Accuracy(IMF)	76.36%

TABLE II
Model Performance For Dataset2

Dataset2 doesn't perform as good as dataset1. One of the main reasons behind that is the limited amount of training data availability. Even within dataset2, the Correctly Faces Faces category classification accuracy is significantly higher than Incorrectly Masked Faces accuracy because the Incorrectly Masked Faces category has only 550 images.

Experiment3: CNN model got applied on Dataset3 using 5-fold cross-validation. The dataset contains images of Correctly Masked Faces and Incorrectly Masked Faces category with 2844, 3091 images respectively. The performance of the model can be viewed in TABLE III.

Experiment3 confirms one of the findings of experiment1. It can be noticed in both of these experiments that images where the nose and mouth are covered but the chin is not, have lower accuracy in comparison to any other category. As it can be seen that classification accuracy for the Chin Open category is only 95.62% in experiment3 and is lower than all the other categories of masked faces.

Type of Face Masks	Accuracy
Complete Dataset	98.06%
Correctly Masked Accuracy	98.07%
Incorrectly Masked Accuracy(IMF)	98.05%
IMF Nose Open Accuracy	96.04%
IMF Nose Mouth Open Accuracy	98.54%
IMF Chin Open Accuracy	95.62%
IMF No Mask Accuracy	97.91%

TABLE III
MODEL PERFORMANCE FOR DATASET3

Experiment4: In the experiment4, a new dataset got created from the merger of Dataset1, Dataset2, and Dataset3. A CNN model got applied to this newly formed dataset and stratified 5-fold cross-validation got used for training the model. The performance of the model can be viewed in TABLE IV.

Type of Face Masks	Accuracy
Complete Dataset	97.57%
Correctly Masked Dataset1(MaskedFaceNet)	99.29%
Incorrectly Masked Dataset1(MaskedFaceNet)	98.08%
Correctly Masked Dataset2(Kaggle)	87.93%
Incorrectly Masked Dataset2(Kaggle)	79.81%
Correctly Masked Dataset3(SCFace)	97.64%
Incorrectly Masked Dataset3(SCFace)	97.88%

TABLE IV
MODEL PERFORMANCE FOR ALL DATASETS MERGED TOGETHER

Experiment5: The dataset used for experiment5 is the same as experiment4. On top of that, it also uses GAN to generate some additional images for Dataset2 Incorrecly Masked Faces because this category has only 550 images. The performance of the model can be viewed in TABLE V.

Type of Face Masks	Accuracy
Complete Dataset	97.51%
Correctly Masked Dataset1(MaskedFaceNet)	97.96%
Incorrectly Masked Dataset1(MaskedFaceNet)	97.51%
Correctly Masked Dataset2(Kaggle)	89.63%
Incorrectly Masked Dataset2(Kaggle)	74.72%
Correctly Masked Dataset3(SCFace)	99.33%
Incorrectly Masked Dataset3(SCFace)	95.78%

TABLE V
MODEL PERFORMANCE FOR ALL DATASETS MERGED TOGETHER AND MORE IMAGES GENERATED USING GAN

The comparative analysis between experiment4 and experiment5 can not find noticeable performance differences despite tacking the issue of class imbalance using GAN. One of the main reasons behind that is GAN model does not have enough data for the training. Hence, the quality of GAN-generated images is significantly poor in comparison to original images.

Comparative analysis between all the experiments: Let's compare the results obtained from the above five experiments.

It can be seen that Dataset3 slightly outperforms all other datasets. One of the reasons behind that is it includes images of subjects with different types and colors of face masks. It also has images of humans with different angles for example Front, Left, Right. Another reason for that is it doesn't have misclassified images.

Model	Accuracy	CMFA	IMFA
CNN Dataset1	97.90	97.81	97.99
CNN Dataset2	91.85	95.58	76.36
CNN Dataset3	98.06	98.07	98.05
CNN Merged Dataset	97.57	97.71	97.22
CNN Merged D. GAN	97.51	97.63	97.31

TABLE VI
PERFORMANCE COMPARISON OF VARIOUS EXPERIMENTS

VI. CONCLUSION

In this paper, a CNN model got applied on three different datasets for the detection of correctly and incorrectly masked faces. The experiment got performed by arranging data in 5 different ways. The research concludes that deep learning models can be used for the classification of correctly and incorrectly worn face masks. The model got classification accuracy of around 98% for Dataset1 and Dataset3 where both the datasets are artificially generated. However, it doesn't perform relatively well on Dataset2 which is a real-world dataset. One of the reasons behind that is a relatively small number of images are available in Dataset2 especially in Incorrectly Wore Face Mask Category with only 550 images. The experiment also uses GAN to generated more images for this category, but it doesn't help much because such a small size of training sample cannot train the GAN model well enough to generate good quality images. Even though, the earlier research on face mask detection can detect whether people are wearing a face mask or not. It cannot classify whether people are wearing face masks correctly or not. This research can be used to develop a real-world detection system for correctly and incorrectly worn face masks.

REFERENCES

[1] K. Dang and S. Sharma, ”Review and comparison of face detection algorithms,” 2017 7th International Conference on Cloud Computing, Data Science Engineering - Confluence, Noida, 2017, pp. 629-633, doi: 10.1109/CONFLUENCE.2017.7943228.

[2] S. Jing, C. Hu, C. Wang, G. Zhou and J. Yu, ”Vehicle Face Detection Based on Cascaded Convolutional Neural Networks,” 2019 Chinese Automation Congress (CAC), Hangzhou, China, 2019, pp. 5149-5152, doi: 10.1109/CAC48633.2019.8996162.

[3] S. Albawi, T. A. Mohammed and S. Al-Zawi, ”Understanding of a convolutional neural network,” 2017 International Conference on Engineering and Technology (ICET), Antalya, 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.

[4] Adnane Cabani, Karim Hammoudi, Halim Benhabiles, Mahmoud Melkemi,MaskedFace-Net â A dataset of correctly/incorrectly masked face images in the context of COVID-19,Smart Health,Volume 19,2021,100144,ISSN 2352-6483

[5] Zhang, K., Zhang, Z., Li, Z., and Qiao, Y. (2016). Joint face detection and alignment using multitask cascaded convolutional networks. IEEE Signal Processing Letters, 23(10):1499â1503.

[6] Mislav Grgic, Kresimir Delac, Sonja Grgic, SCface - surveillance cameras face database, Multimedia Tools and Applications Journal, Vol. 51, No. 3, February 2011, pp. 863-879

[7] ”Generative Adversarial Networks.” Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. ArXiv 2014.

[8] Viola, Paul Jones, Michael. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conf Comput Vis Pattern Recognit. 1. I-511. 10.1109/CVPR.2001.990517.

[9] Mohamed Loey, Gunasekaran Manogaran, Mohamed Hamed N. Taha, Nour Eldeen M. Khalifa,A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic,Measurement,Volume 167,2021,108288,ISSN 0263-2241
[10] I. B. Venkateswarlu, J. Kakarla and S. Prakash, "Face mask detection using MobileNet and Global Pooling Block," 2020 IEEE 4th Conference on Information Communication Technology (CICT), Chennai, India, 2020, pp. 1-5, doi: 10.1109/CICT51604.2020.9312083.
[11] Larxel, (2020, May). Face Mask Detection, Retrieved February 5, 2020 from https://www.kaggle.com/andrewmvd/face-mask-detection