Comparative Study: Masked Face Classification and Recognition

Yufei Zhao
University or Rochester
yzhao87@u.rochester.edu

Jiayi He
University of Rochester
jhe36@u.rochester.edu

Abstract

Nowadays, masks have become the essential tool to fight against the spread of the Covid-19 virus. Researchers started developing many Deep Learning algorithms to deal with the new problems related to masks. Most of the algorithms that have been released are related to either the classification or the recognition problems. In this project, we develop our own algorithm of classification and recognition. Our method will give us information about people respecting the sanitary protocol and make it easy to recognize people wearing masks. We develop six models that will treat the classification of people wearing masks and not, the classification of masked women and masked men, and the masked face recognition. Six models in the project are built with the LeNet-5, the MobileNetV2, and the VGG-16 architecture, while transfer learning and fine-tuning are applied to fix the issue related to insufficient data, especially for the case of the recognition task. We will have two models based on different architectures for each task and compare their performance via accuracy.

1. Introduction

The Covid-19 epidemic has quickly swept the world, posing a significant threat to human life and health. More than 270 million people have been infected with the Covid-19 virus and about 3 million people have been infected in the recent 28 days [4]. So, wearing masks has become an essential protective measure. For the convenience of schools, companies, and other institutions monitoring whether members are wearing masks in public spaces in real-time, face recognition with masks becomes extremely important. Also, due to the strengthening of control measures, people now almost have to wear masks when going out, so it is necessary to recognize faces in the case of wearing masks.

The current face recognition technology is relatively complete and has been applied in many fields, most of which are implemented by Convolutional Neural Networks and Deep Learning algorithms. Deep Learning face recognition generally first performs face detection to deduct the face area and locates the feature point. Then, it performs Affine Transformation according to the location feature points to align the face and sends the aligned face to the feature extraction network to extract facial features. Finally, it uses the features for face recognition. We corresponded this process to the case of wearing a mask. More specifically, we designed and built this masked face classification and recognition project via Masked and Non-masked Face Classification, Masked-face Women/Men Classification, and Masked-face Recognition based on Neural Networks and Transfer Learning.

To achieve different purposes, we built three different data sets that contain both data for train and validation. We collected part of our data from public databases on the Internet. However, due to the short time since the outbreak of COVID-19, there are not many publicly available facial mask databases on the web at present. Therefore, to enrich the datasets for this project, we self-made part of facial mask images for training and validation. To be more specific, in this project, we mainly focused on the comparative studies of three different models for various classification/recognition tasks, including Masked and Non-masked Face Classification, Masked-face Women/Men Classification, and Masked-face recognition. This report will describe the datasets, methods, experiment process and results, and the conclusion of our study in detail.

2. Related Works

Intensive work has been done to perform the face classification/recognition task. Convolutional Neural Networks (CNN) are among the most popular neural network frameworks that solve face recognition tasks. Over years, many variants of CNN are developed for more specific scenarios and purposes and high accuracy, including LeNet-5 [12], VGG-16 [16], and MobileNetV2 [9]. These models of CNN are compared with their strengths and weaknesses being analyzed in numerous studies and researches. As the first proposed CNN architecture, LeNet-5 reduces the number of parameters and exploits the spatial correlation in the images. It is known for its contribution on digit recognition tasks without distortion, while it doesn't perform well on multiple image recognition tasks [6] [14]. VGG-16 improves by applying smaller size kernels/filters and has better feature extraction ability to achieve a much higher accuracy [17]. However, its intensive use of parameters makes it computationally expensive and sometimes difficult to set up on low-resource systems [8]. On the contrary, MobileNetV2 models can easily be deployed in recourse constraint environment [10].

In the context of the Covid-19 pandemic, though some reviews of different CNN models on masked face recognition are conducted [1] [7] and provide the most accurate model based on their datasets, mask detection and face recognition tasks are not well explored yet. Some of the existing successful models are developed and evaluated based on a large dataset, thus they may have a poor performance on a smaller dataset. While in some cases we do not have enough data or enough time to label the data, it is worth presenting a comparative study of CNN models on a smaller dataset. Besides, most algorithms in this field are either about classification or recognition problems. In this paper, we will develop an algorithm of classification and recognition with CNN models and compare between three Convolutional Neural Network-based models which are LeNet-5, MobileNetV2, and VGG-16 with a semi-customized dataset of Masked and Non-masked faces. We want to see whether the pre-Covid19 analysis of the different CNN models still preserve in our relatively smaller masked face datasets.

3. Datasets

3.1. Overview

In this project, we organized our database by collecting public databases [5] [13] on the Internet and self-making own image database by generating masks on non-mask face images.

For Masked and Non-masked Face Classification, the dataset for this section contains

- 300 non-masked face images, and 300 masked face images for training (total number of training images is 600)

- 153 non-masked face images, and 153 masked face images for validation (total number of validation images is 306)

For Masked-face Women/Men Classification, the dataset for this section contains

- 1099 masked men face images, and 1100 masked women images for training (total number of training images is 2199)

- 1026 masked men face images, and 1025 masked women images for validation (total number of validation images is 2051)

For Masked-face Recognition, the dataset for this section contains

- Five famous people masked face image datasets, including Colin Powell, Donald Rumsfeld, Gerhard Schroeder, Hillary Clinton, and Jennifer Capriati

- The total number of training images for Colin Powell is 215, and the total number of validation image for Colin Powell is 106

- The total number of training images for Donald Rumsfeld is 292, and the total number of validation images for Donald Rumsfeld is 107

- The total training number of images for Gerhard Schroeder is 206, and the total number of validation images for Gerhard Schroeder is 106

- The total training number of images for Hillary Clinton is 200, and the total number of validation images for Hillary Clinton is 106

- The total training number of images for Jennifer Capriati is 198, and the total number of validation images for Jennifer Capriati is 114

As there are few clear facial pictures of these celebrities wearing masks that can be collected on the Internet, we made this part of the dataset by searching their photos on different occasions and manually adding mask elements.

3.2. Data Augmentation

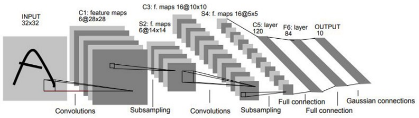
We implemented Data Augmentation technique to process image dataset in order to increase model performance and avoid over-fitting issues. This technique mainly contains flipping, rotation, shearing, cropping, zooming in/out, changing brightness, and contrasting [11]. In this project the transformations applied on images in the dataset include scaling, rotation, zooming in/out, shifting, flipping, and shearing.

- Zooming in/out: transforming images to different sizes, enabling models to identify and process smaller images by making models less sensitive to image scales

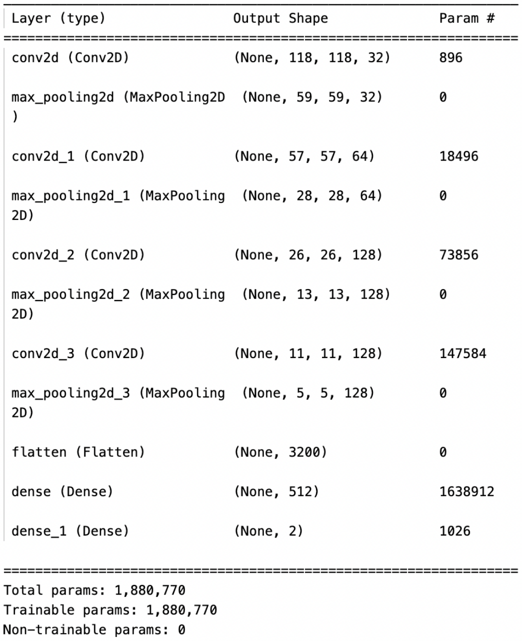
- Rotation: rotating images by specific degrees

- Shifting: moving pixels of an image from one position to another position

- Shearing: shifting one part of the image like a parallelogram



(a) Typical LeNet-5 Architecture



```
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 118, 118, 32)      896

 max_pooling2d (MaxPooling2D  (None, 59, 59, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 57, 57, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 28, 28, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 26, 26, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 13, 13, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 11, 11, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 5, 5, 128)        0
 2D)

 flatten (Flatten)           (None, 3200)              0

 dense (Dense)               (None, 512)               1638912

 dense_1 (Dense)             (None, 2)                 1026

=================================================================
Total params: 1,880,770
Trainable params: 1,880,770
Non-trainable params: 0
```

(b) LeNet5 Architecture for Mask and Non-mask Face Classification

Figure 1. LeNet-5 Architecture

4. Methods

4.1. LeNet-5 Architecture

LeNet-5 is a gradient-based learning CNN structure. The typical LeNet-5 structure diagram is shown in Figure 1 (a). It has three convolutional layers, two pooling layers, and one fully connected layer. The fully connected layer reduces the number of neurons to reduce parameter training. Our modified LeNet-5 model takes 120 × 120 pixel RGB images. Indeed, all the initial images are resized to the desired size (120 × 120). For the convolutionary layers, it uses very small filters, which is the minimum size to capture left/right and up/down. Max pooling layers are employed after the convolutionary layers with strides dimension of 2×2 to down-sample to previous output. At last, the architecture contains two fully connected layers and ends up with two channels, which corresponding to the binary classes in the mask detection/classification case and gender detection/classification case.

4.2. MobileNetV2 Architecture

MobileNetV2 is a typical CNN architecture that seeks to perform well on mobile devices. It is based on an inverted residual structure where the residual connections are between the bottleneck layers [15]. Fig 2 (a) shows that this architecture basically contains the initial fully convolution layer with 32 filters, followed by 19 residual bottleneck layers. More specifically, in Fig 2 (a), T represents expansion multiple, C represents the number of output channels, N represents the number of repetitions, and S represents the stride length. It should be noted that the stride is equal to 2 at 7 to 10 bottlenecks, where the resolution drops from 28 to 14, and the stride is equal to 1 under normal conditions. And Fig 2 (a) shows only 17 bottlenecks when there should be 19. Compared with traditional CNN, MobileNetV2 has its bottleneck. It mainly undertakes the process of expansion, convolution, and compression as well as output.

The MobileNetV2 model (Figure 3) that we used took 120×120 pixel RGB images as we did with the first model, and these prepared images were put into the pre-trained MobileNetV2 model, which contains 30 layers distributed as convolutional layer with stride 2, depthwise layer, pointwise layer that doubles the number of channels, depthwise layer with stride 2, and pointwise layer that doubles the number of channels. Finally, we processed the output of the model with global average pooling and fully connected layers based on the tasks we performed.

4.3. VGG-16 Architecture

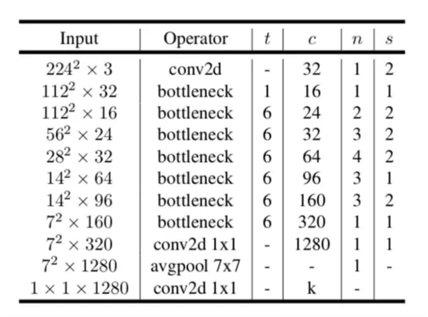
Typically, the VGG-16 architecture consists of twelve convolutional layers, some of which are followed by maximum pooling layers. It maintains this order of convolution and max-pooling layers throughout the architecture. Then two fully-connected layers and a softmax classifier. The 16 in VGG16 alludes to the fact that it has sixteen weighted layers. In our study, we use the same layers as shown in Fig 4 and freeze all the convolutional layers and pooling layers. So, the model only trains fully connected layers and that is how we fine-tune the VGG16 model for our purpose and apply Transfer Learning. Adam optimizer is used for the VGG16 model.

The softmax function used in VGG-16 architecture:

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{1}$$

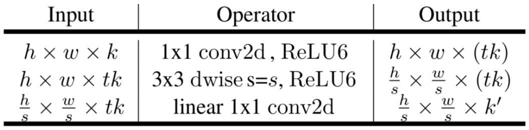
4.4. Transfer Learning and Fine Tuning

Deep Learning models can be trained on one task, and then fine-tuned on another related task, otherwise known as transfer learning [3] [2]. This is because for similar domains the weights learned at lower levels are similar. In our study, we lock almost all previously learned weights



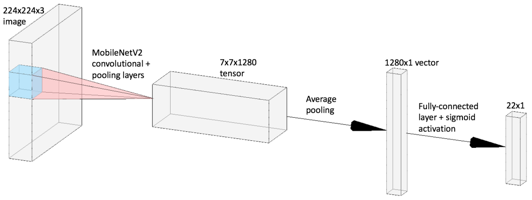
Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	

(a) Typical MobileNetV2 Architecture



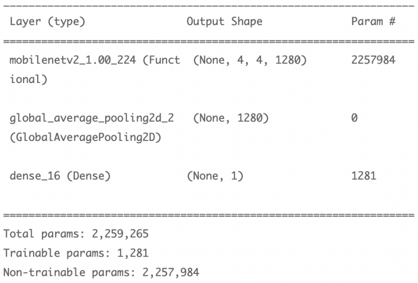
Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

(b) MobileNetV2 Bottleneck Structure



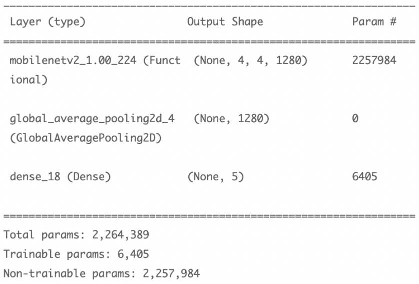
(c) MobileNetV2 Global Average Pooling and Fully Connected Layer

Figure 2. MobileNetV2 Architecture



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 mobilenetv2_1.00_224 (Funct  (None, 4, 4, 1280)       2257984
 ional)

 global_average_pooling2d_2   (None, 1280)             0
 (GlobalAveragePooling2D)

 dense_16 (Dense)            (None, 1)                 1281

=================================================================
Total params: 2,259,265
Trainable params: 1,281
Non-trainable params: 2,257,984
```

(a) MobileNetV2 Architecture for Masked-face Women/Men Classification

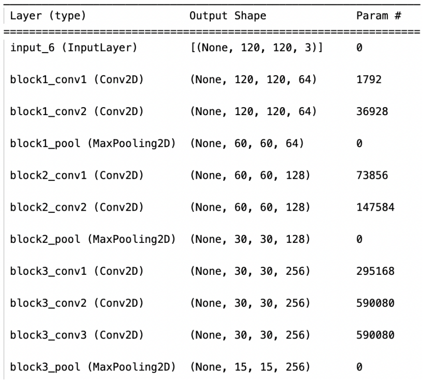


```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 mobilenetv2_1.00_224 (Funct  (None, 4, 4, 1280)       2257984
 ional)

 global_average_pooling2d_4   (None, 1280)             0
 (GlobalAveragePooling2D)

 dense_18 (Dense)            (None, 5)                 6405

=================================================================
Total params: 2,264,389
Trainable params: 6,405
Non-trainable params: 2,257,984
```

(b) MobileNetV2 Architecture for Masked Face Recognition

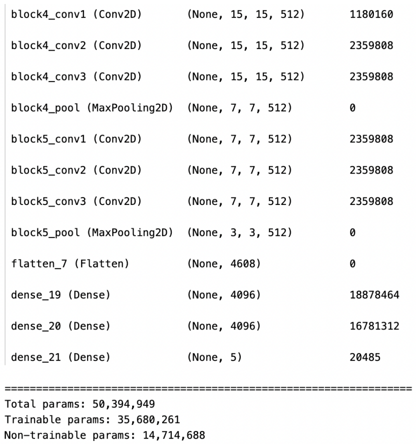
Figure 3. MobileNetV2 Architecture

barn those on the output layer. We manage to take advantage of the feature extraction stage of the network and only tune the final classifier to work better with our dataset. The pre-trained models are especially useful when dealing with smaller datasets. We apply transfer learning and fine tuning to the MobileNetV2 and VGG-16 architecture.



```
 Layer (type)                Output Shape              Param #
=================================================================
 input_6 (InputLayer)        [(None, 120, 120, 3)]     0

 block1_conv1 (Conv2D)       (None, 120, 120, 64)      1792

 block1_conv2 (Conv2D)       (None, 120, 120, 64)      36928

 block1_pool (MaxPooling2D)  (None, 60, 60, 64)        0

 block2_conv1 (Conv2D)       (None, 60, 60, 128)       73856

 block2_conv2 (Conv2D)       (None, 60, 60, 128)       147584

 block2_pool (MaxPooling2D)  (None, 30, 30, 128)       0

 block3_conv1 (Conv2D)       (None, 30, 30, 256)       295168

 block3_conv2 (Conv2D)       (None, 30, 30, 256)       590080

 block3_conv3 (Conv2D)       (None, 30, 30, 256)       590080

 block3_pool (MaxPooling2D)  (None, 15, 15, 256)       0
```

(a) VGG-16 Architecture Part 1



```
 block4_conv1 (Conv2D)       (None, 15, 15, 512)       1180160

 block4_conv2 (Conv2D)       (None, 15, 15, 512)       2359808

 block4_conv3 (Conv2D)       (None, 15, 15, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 block5_conv1 (Conv2D)       (None, 7, 7, 512)         2359808

 block5_conv2 (Conv2D)       (None, 7, 7, 512)         2359808

 block5_conv3 (Conv2D)       (None, 7, 7, 512)         2359808

 block5_pool (MaxPooling2D)  (None, 3, 3, 512)         0

 flatten_7 (Flatten)         (None, 4608)              0

 dense_19 (Dense)            (None, 4096)              18878464

 dense_20 (Dense)            (None, 4096)              16781312

 dense_21 (Dense)            (None, 5)                 20485

=================================================================
Total params: 50,394,949
Trainable params: 35,680,261
Non-trainable params: 14,714,688
```

(b) VGG-16 Architecture Part 2

Figure 4. VGG-16 Architecture

5. Experiments

For our experiment, we used Keras [15] with TensorFlow backend. Keras is a high-level neural networks API, written in Python which provides a huge amount of functions and models regarding neural networks and image processing. The models we used were trained and tested in Google Colaboratory which a research tool for machine learning education to get GPU support in order to reduce training time.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \qquad (2)$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative.

5.1. Masked and Non-masked Face Classification

In this Masked and Non-masked Face Classification task, we used the LeNet-5 architecture and MobileNetV2 archi-

Model	Min Accuracy	Avg Accuracy	Max Accuracy
LeNet-5	0.91	0.92	0.95
MobileNetV2	0.92	0.94	0.96

Table 1. Masked and Non-masked Face Classification Validation Results out of 10 runs

Model	Min Accuracy	Avg Accuracy	Max Accuracy
LeNet-5	0.60	0.72	0.79
MobileNetV2	0.76	0.83	0.89

Table 2. Masked-face Women/Men Classification Validation Results out of 5 runs

tecture. We wanted to study the difference between the two architectures in this case by comparing the resulting accuracy.

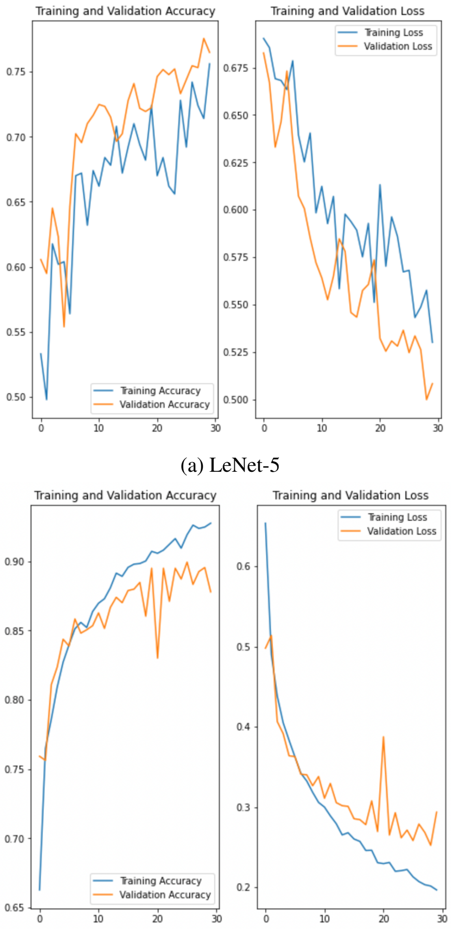
For this task, we employed the first dataset, which contains images of masked faces and non-masked faces. The results we got are shown in Table 1. The accuracy we got for both models is satisfactory, and it can be explained by the clear difference between a masked face and a normal face. Also, by comparing the results of LeNet-5 architecture and MobileNetV2 architecture, we realized that the use of transfer learning has further improved the performance, evidenced by higher accuracy of the MobileNetV2 architecture.

5.2. Masked-face Women/Men Classification

In this Masked-face Women/Men Classification task, we used the LeNet-5 architecture and MobileNetV2 architecture. We wanted to study the difference between the two architectures in this case by comparing the resulting accuracy.

For this task, we applied the second dataset, which contains images of masked women faces and men faces. The results we got are shown in Table 2 and Figure 5. Having a model that classifies between masked people whether they are male or female is very challenging, and that is why the performance that we got is less than the first case of simple masked and non masked face classification. We could still notice the impact of transfer learning on our results. Indeed, having a pretrained model that uses stored knowledge from other data is very helpful in our case.

Besides, the enlargement of the second dataset did enhance the accuracy for both models and reduced the overfitting problems. With a dataset of 300 train images, we could only get an accuracy of about 0.60 for LeNet-5 architecture and about 0.66 for MobileNetV2 architecture.



(a) LeNet-5

(b) MobileNetV2

Figure 5. Masked-face Women/Men Classification

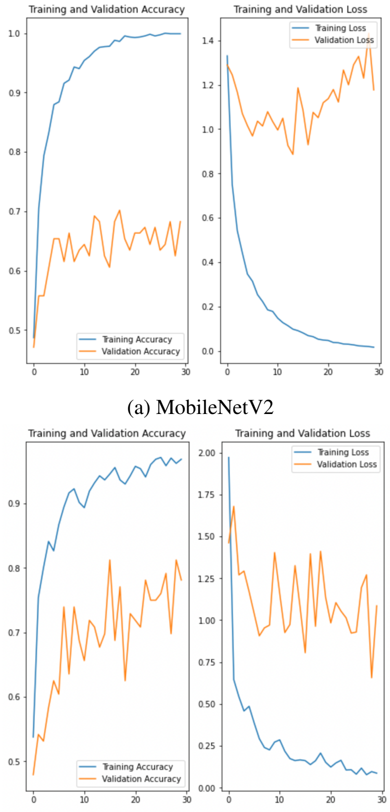
5.3. Masked Face Recognition

In this Masked-face Recognition task, we used the MobileNetV2 architecture and VGG-16 architecture, both of which are popular CNN models in previous non-masked face recognition experiments. We wanted to study the difference between the two architectures in this case by comparing the resulting accuracy. Since the size of our dataset for this task is relatively small despite efforts, so we applied transfer learning on both models to achieve higher accuracy.

For this task, we employed the third dataset, which contains images of masked face images with their identity labels. The results we get are shown in Table 3 and Figure 6. The accuracy we got to classify 5 people is reasonable based on the size of our dataset and the limited resources of our GPU. The real-time recognition result we got from the model by uploading test images are usually correct. It is worth noticing that VGG-16 model defeats MobileNetV2 model in this case, possibly because of its stronger feature extraction ability. However, there was still a potential overfitting problem in this task after we enlarged the dataset and applied the data augmentation techniques.

Model	Min Accuracy	Avg Accuracy	Max Accuracy
MobileNetV2	0.62	0.66	0.69
VGG16	0.73	0.76	0.78

Table 3. Masked Face Recognition Validation Results out of 5 runs



(a) MobileNetV2

(b) VGG16

Figure 6. Masked Face Recognition

6. Conclusion

In this project, we introduced mask detection via basic masked and non-masked face classification, masked-face women/men classification, and masked-face recognition using six models based on three CNN architectures(LeNet-5, MobileNetV2, and VGG-16) and Transfer Learning. Both the MobileNetV2 model and VGG16 model were fine-tuned in order to train our datasets. The MobileNetV2 model is relatively smaller in size comparing the other two. Generally, we got good performance on the six models. The transfer learning models perform much better. For the last task where both models are transfer-learning models, VGG-16 stands out in terms of validation accuracy.

References

[1] Tanvir Ahmed, Prangon Das, Md Firoj Ali, and Md-Firoz Mahmud. A comparative study on convolutional neural network based face recognition. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–5. IEEE, 2020. 2

[2] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings, 2012. 3

[3] Rich Caruana. Learning many related tasks at the same time with backpropagation. In *Advances in neural information processing systems*, pages 657–664, 1995. 3

[4] Johns Hopkins Coronavirus Resource Center. Covid-19 dashboard. 1

[5] Muhammed Dalkıran. Lfw simulated masked face dataset, Sep 2020. 2

[6] Amita Dev, Arun Sharma, and SS Agrawal. Artificial Intelligence and speech technology. 2

[7] Faisal Dharma Adhinata, Nia Annisa Ferani Tanjung, Widi Widayat, Gracia Rizka Pasfica, and Fadlan Raka Satura. Comparative study of vgg16 and mobilenetv2 for masked face recognition. *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, 7(2):230–237, 2021. 2

[8] Klemen Grm, Vitomir Štruc, Anais Artiges, Matthieu Caron, and Hazım K Ekenel. Strengths and weaknesses of deep learning models for face recognition against image degradations. *Iet Biometrics*, 7(1):81–89, 2018. 2

[9] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017. 1

[10] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, 2020. 2

[11] Renu Khandelwal. Data augmentation techniques in python, Dec 2019. 2

[12] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[13] Prithwiraj Mitra. Covid face mask detection dataset, Jul 2020. 2

[14] Witold Pedrycz and Shyi-Ming Chen. *Deep Learning: Algorithms and Applications*. Springer, 2020. 2

[15] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 3

[16] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1

[17] Chiranjibi Sitaula and Mohammad Belayet Hossain. Attention-based vgg-16 model for covid-19 chest x-ray image classification. *Applied Intelligence*, 51(5):2850–2863, 2021. 2