Face Mask Detection System Constructed with a Combination of ResNet50 and Support Vector Machine

Ruitian Wu
Department of Data and Computational Science
Duke Kunshan University
ruitian.wu@dukekunshan.edu.cn

Jingwei Li
Department of Data and Computational Science
Duke Kunshan University
jingwei.li698@dukekunshan.edu.cn

Abstract

COVID-19 pandemic is causing a serious global health crisis. As suggested by the World Health Organization (WHO), one of the effective protection methods is to wear a face mask. Therefore, a model that could detect face masks is eagerly need to help governments to surveil their citizens espically in public areas. In this paper, a deep learning model and a hybrid model using deep and machine learning for face mask detection will be introduced. The first model is Convolutional Neural Network (CNN). The second model is a hybrid model that has two components where the first uses Resnet50 for feature extraction, and the second part is to combine SVM with the extracted features and do the classification. The CNN classifier achieved 91.87% accuracy. In the hybrid model, it achieved a better performance with an accuracy of 94.26%. For our models, there are two innovation points: Firstly, we further included a class called "wearing masks incorrectly" to make our surveillance system more harsh and applicable to real-life. Secondly, we combined the advantages of deep learning framework and machine learning classifiers to achieve the outcome of "1+1>2". Also, future improvement directions are pointed out in our research including expanding dataset, retraining detection models, augmenting images and creating more classes of labels.

Keywords: Face Detection, Cascade Classifier, Multi-class Classification, Support Vector Machine, Convolutional Neural Network, ResNet50

1 Introduction

The world is facing a huge health crisis due to the rapid transmission of coronavirus (COVID-19). According to WHO [1], the majority of the positive cases are found in crowded and overcrowded areas. It was prescribed by the scientists that wearing a mask in public places can prevent transmission of the disease[2].

Meanwhile, with the rapid development in the area of computer science, technologies such as facial recognition and image detection are becoming increasingly mature. It is natural to come up with an idea of applying these advanced technologies to the detection of masks.

The main aim of this work is to use machine learning and deep learning algorithms to develop a model for detecting whether a person who wears a mask, who does not wear a mask or who wears a

Preprint. Under review.

mask incorrectly. We applied two deep learning algorithms: CNN and ResNet, and combined SVM with ResNet to generate two models and afterwards compared their performances. The rest of the paper is divided into 4 sections and is as follows.

2 Literature Review

Since the outbreak of the COVID-19, the task of face detection has become a quite heated topic, and many studies have been produced employing diverse models. On the one hand, the urgent crisis requires people to find a way to mollify its impact. On the other hand, with the fast development of computer technologies, including Computer Vision and Deep Learning, people are now mastering more tools to build face detection models.

Basically, for all studies, the task of face detection can be disassembled into two parts:

- The first is the object detection task. In order to determine whether a certain is wearing mask or not, it is vital firstly to locate their exact locations in videos or images.
- The second task is a classification task. After determining the location of each face and extracting them from the original image, we should classify each of them to a certain category which shows their conditions of whether they are wearing masks.

The divergence of difference studies mainly lies in the choice of algorithms for the two tasks. To sum up, the methods employed by previous researchers could be classified into two categories:

1. Single-stage Task. The single-stage detectors treat the detection of region proposals as a simple regression problem by taking the input image and learning the class probabilities and bounding box coordinates [3]. Among the methods, one famous method is called YOLO (You Only Look Once). The method divides an image into a grid of size GxG, and each grid generates N predictions for bounding boxes. Each bounding box is limited to have only one class during the prediction, which restricts the network from finding smaller objects [4]. Afterwards, the method is being continuously updated to version 2, 3 and even 4, which has better performance. Although, YOLOv3 is executed faster than Single-Shot Detector (SSD) but does not perform well in terms of classification accuracy [3]. Moreover, YOLOv3 requires a large amount of computational power and might not be suitable for mobile devices.
2. Two-stages Task. It first predicts proposals in an image and then applies a classifier to these regions to classify potential detection [3]. One famous method is called R-CNN, which applies a selective search algorithm to extract a set of object proposals at an initial stage and applies different classifiers for predicting objects [5]. Then a faster version of R-CNN is designed as well, which enables nearly cost-free region proposals by gradually integrating individual blocks (e.g. proposal detection, feature extraction and bounding box regression) of the object detection system in a single step [6]. Although two-stage tends to have a higher accuracy during detection, for video surveillance mode, its inference speed is still quite low, which could be unsatisfactory.

Generally speaking, most models employed some deep learning frameworks while predicting and could achieve a relatively high accuracy while predicting– most of them could have an accuracy of more than 95%. So for our further study, our main goal is not to achieve as high accuracy as possible but instead mainly focused on two points:

1. For most classification models, they are simply employing the deep learning models by just continuously adding and modifying layers. Usually for the dense layers, a simple softmax activation function is employed. However, there is possibility that we could replace the dense layer with some other machine learning algorithms, like Support Vector Machine or AdaBoost. Here, nerual network frameworks are not used for classification but just feature extraction and we would like to apply some more explicit machine learning algorithms to achieve possibly better performance.
2. For most studies, the classification task is binary, indicating there are usually two categories in total: Wearing masks and not wearing masks. However, in reality, the situation could be much more complicated. For example, even if someone is wearing a mask, sometimes they

are not wearing it properly. The proper use of masks requires strict adherence to general hygiene measures, among which adequate coverage of the mouth and nose, avoiding gaps between the face and the mask, stands out. A partial, incorrect, or asymmetric fit poses a high risk for the transmission of infection [7]. Therefore, we would like to add a new category called "wearing masks improperly" to make clearer distinctions.

3 Methods

In the following section, we will cover detailedly the data we used for the model, the pilot test we conducted before former model building and the basic introduction to the algorithms we employed in this project.

3.1 Data Overview

For the dataset of our project, we initally turned to an open-source dataset named "Face Mask Detection" on Kaggle [1]. For the dataset, it has 853 images of human faces and commensurate number of annotations for each image.

In each image, there are various faces which are labelled "wearing mask", "not wearing mask", or "wearing mask improperly" separately. Moreover, the location of each face in images has already been given in annotations so that we could directly mark them from the original image. Moreover, the images could be visualized as below:

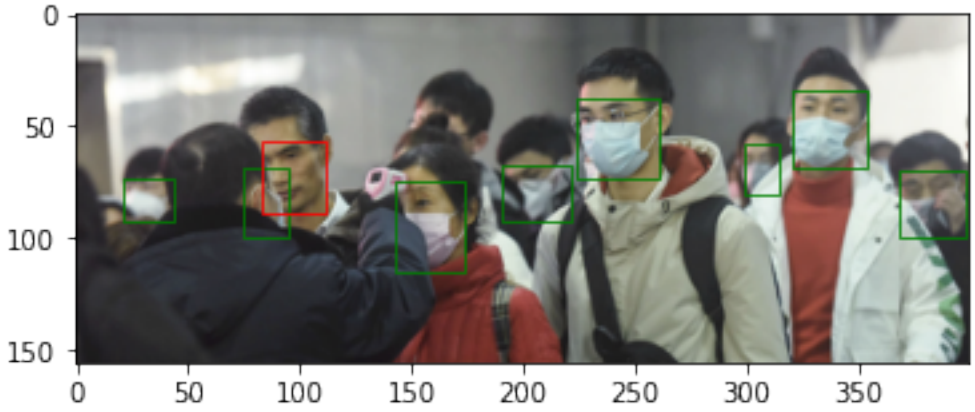

Figure 1: Visualization of Data

With the given annotations, we extracted all the faces and their corresponding labels to create a new dataset. Apart from image segmentation, we did two more things to ensure training efficiency: Firstly, we set a threshold for the image, which is a size of 40×40. Images below this size will be obliterated from our dataset due to low resolution. Secondly, we resized all images to a size of 128×128 to ensure that in our following tests, the inputs are commensurate.

3.2 Pilot Test

After we already had two main elements of the data set: images and labels, we are going to start our pilot test. Pilot testing is one traditional and specialized form of qualitative research, is used in the development of quantitative survey items that are piloted on study participants in order to test the reliability and validity of items. In our case, we plan to apply Support Vector Machine (SVM) to our current data set as our pilot test.

Below are the basic procedures we take for our pilot test:

1. Grey the faces to avoid the impacts of colors and brightness on our model.

[1] https://www.kaggle.com/andrewmvd/face-mask-detection?select=images

2. Convert images to matrices so that it became readable and interpretable.
3. Normalize and reshape the matrices to construct a vector of length $128 \times 128 \times 3$
4. Make classifications with the obtained features and labels using a SVM model (hyper-parameter: Gaussian kernel and a regularization factor of 10)
5. Examine the performance of the model with its accuracy and confusion matrix.

After running the pilot test model, we got the outcome of the SVM model. For the accuracy, the optimal result that we get from this SVM model is **88.67%**. From the figure, it seems that SVM has already achieved a quite impressive performance. However, if we take a look at the confusion matrix, we could find some issues regarding the result:

True label/Predict label	Incorrect	Yes	No
Incorrect	1	10	1
Yes	0	241	4
No	0	19	24

Table 1: Confusion matrix for pilot test

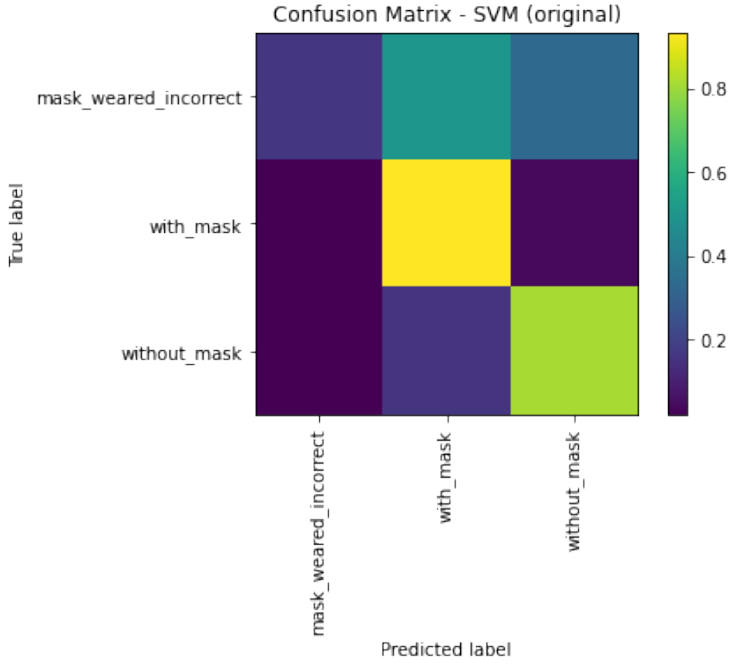

Figure 2: Confusion Matrix Plot for Pilot Test

From the confusion matrix, it is not hard to find out that while predicting faces with masks, the model tends to make accurate decisions but when it comes to the other two categories, the performance is not that satisfactory. However, it's extremely important for us categorize the people without masks and not wearing masks correctly since they are the most important groups of people that we care about in our study. Therefore, our pilot indicates that there are some extents of drawbacks in our dataset now. If we look at the component of our dataset, we can clearly find that our dataset is extremely biased: faces with masks take up over 80% of the whole set. Therefore, those faces without masks or wearing masks wrongly could not be sufficiently trained. Here the biased dataset is the first significant issue we have to fix.

What's more, during the pilot test, the way we extract features is quite trivial. We convert pixels to matrices and linearly arrange them. However, this could not ensure that we extracted the most significant information from the images. Factors like the relationship between different pixels were forsaken and therefore caused loss of information. Therefore, a more comprehensive way of feature

engineer is needed in our model. Therefore, from the pilot test, we determine two directions for model improvement: Enlarge our data set to a more balanced one and find smarter ways of feature engineer.

3.3 Data Extension

After we conducted the pilot test, the biggest problem we found is the data set is very biased, the detailed situation is shown in **Table 2**:

Labels	Numbers
with_mask	821
without_mask	13
mask_wore_incorrect	46

Table 2: The number of labels in the data set

As shown in **Table 2**, the number of labels with_mask is much more numerous than the other two labels. This imbalance among different labels is likely to result in poor model performance, which in our case, let our model have a strong ability in classifying people with mask while performs poor in classifying people without a mask and who does not wear mask correctly.

3.3.1 New Data Source

In order to solve the biased data set problem, we decided to increase data with labels without_mask and mask_wore_incorrect to make these three labels as even as possible. The new data source is provided as follows:

- **without_mask**: We take the ordinary pictures of human faces as our extension for the without_mask label and it can be found on Kaggle[2].
- **mask_wore_incorrect**: We found a data set with people wearing mask incorrectly on Github[3]. One thing need to mention is that, in this data set, the creators used the mask picture incorrectly mapped onto the face as a case of wearing the mask incorrectly.

An overview of our new data set is shown in **Table 3**. We also provides three samples with different labels shown in below (See **Figure 3, 4, and 5**).

Labels	Numbers
with_mask	821
without_mask	730
mask_wore_incorrect	536

Table 3: The number of labels in the new data set

Figure 3: with_mask sample

Figure 4: without_mask sample

Figure 5: mask_wore_incorrect sample

[2]https://www.kaggle.com/prasoonkottarathil/face-mask-lite-dataset

[3]https://esigelec-my.sharepoint.com/:f:/g/personal/cabani_esigelec_fr/EirjS8ew7-5LnO8I56Uk63wBKebwSlukFBFBaO8N25wn3g?e=Ho1jHG

3.4 Face Detection

After we extend our data and solve the biased problem, we decide to conduct face detection to our data first and then do photo resizing. The reason we want to do face detection is we are concerned that other non-related factors in the figures may have a negative influence on our model's performance. By face detection, we can make the images contain only people's faces and masks.

3.4.1 Haar-like Features

The algorithm we used for face detection is Haar cascades[8]. The core basis for Haar classifier object detection is the Haar-like features (see **Figure 6**). These features use the change in contrast values between adjacent rectangular groups of pixels. The contrast variances between the pixel groups are used to determine relative light and dark areas. Two or three adjacent groups with a relative contrast variance form a Haar-like feature[9]. Haar-like features are used to detect an image [10].

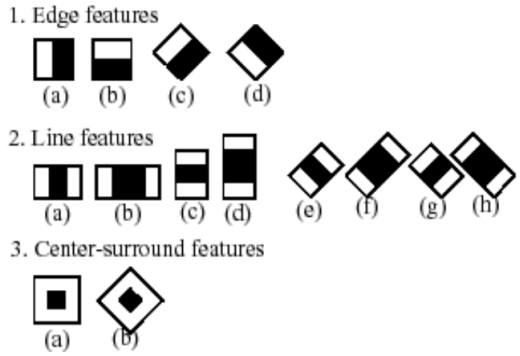

Figure 6: Common Haar Features

3.4.2 Cascade Classifier

Although calculating a feature is quite fast and efficient, it is still impractical to calculating all the images with all the features. In order to solve this problem, the idea cascade classifier was proposed.

The cascading of the classifiers allows only the sub-images with the highest probability to be analyzed for all Haar-features that distinguish an object [9].

3.5 Image Augmentation

Before importing our detected faces into the models, there is one more step that we have to implement to ensure data reliability, which is called image augmentation. Image augmentation artificially creates training sets through different ways of processing, like random rotation, shifts, shear and flips. An example of image augmentation can be shown as below [11]:

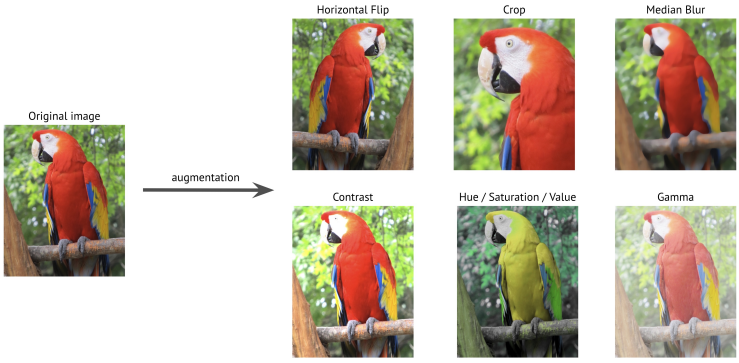

Figure 7: Image augmentation

With image augmentation, we can create more variability of the data, preventing it from being influenced by other factors like brightness, rotation, color and so on. Thus, our model could to some extent prevent the impacts of overfitting.

3.6 Convolutional Neural Network

After all the pre-work has been finished, we are ready to construct our first deep learning model, which is the Convolutional Neural Network (CNN, or ConvNet). The CNN model we construct in this work is based on the LeNet-5 proposed by Yann LeCun, et al. in 1989 [12]. CNN is a class of artificial neural network, most commonly applied to analyze visual imagery. There is a series of important concepts in CNN, for example, input layer, convolutional layer, pooling layer, and dense layer, etc. In the rest of this section, I will briefly go through these ideas and provide the CNN model we construct for mask detection.

3.6.1 Input Layer

In the input layer of CNN, the data (images) input format is different from that of a fully connected neural network (one-dimensional vector). The input format of CNN's input layer retains the structure of the image itself. For a black and white 28×28 picture, the input of CNN is a 28×28 neurons as shown in **Figure 8**:

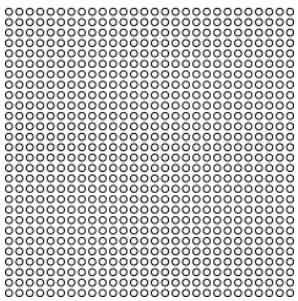

Figure 8: A sample input neuron

In our case, as we mentioned before, we have resized our images into 128×128, and with RGB information. Therefore, the input of our model should be $128 \times 128 \times 3$, where 3 represents the RGB channels.

3.6.2 Convolutoinal Layer

Recall that the input format of our CNN model is $128 \times 128 \times 3$, and now we defined a 5×5 local receptive fields[12]. With the local receptive fields, we can map the original data into a layer called the hidden layer. A hidden layer can be thought of as having fixed size receptive fields to feel some of the characteristics of the former layer. In full connected neural networks, all the characteristics are passed between layers.

One local receptive field has a **convolutional kernel**, which is used for calculating the new values from the former layer to the hidden layer; **stride** is the interval between the local receptive fields scans of the input.

An abridged general view is provided. (See **Figure 9**)

3.6.3 Pooling Layer

When the input passes through the convolutional layer, if the receptive field is small and the stride is small, the obtained feature map is still large. Dimensionality reduction can be performed on each feature map through the pooling layer, and the output depth remains unchanged, which is the number of feature maps.

There are basically two ways to do the dimensionality reduction:

- **Max pooling**: Take the maximum in the matrix.

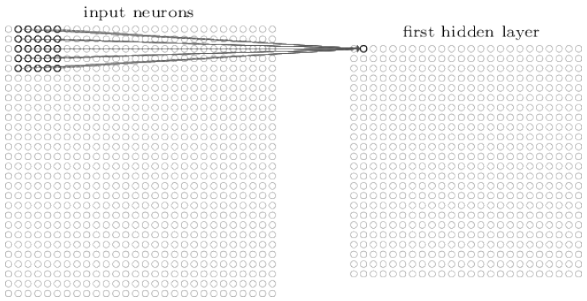

Figure 9: How local receptive fields work

- **Average pooling**: Take the average in the matrix.

Figure 10 provides a demonstration of how pooling works:

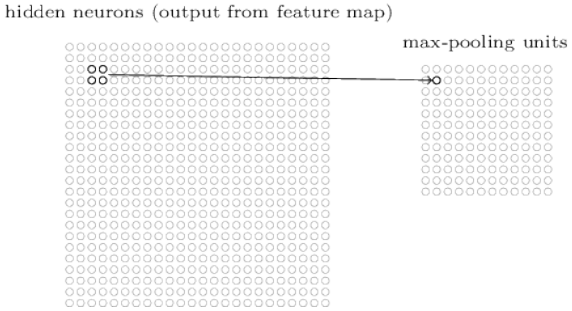

Figure 10: How pooling works

3.6.4 Dense Layer

The dense layer, also called the fully connected layer, plays the role of a classifier in the convolutional neural network. In real practice, the fully connected layer can be realized by convolution operation: the fully connected layer whose front layer is fully connected can be transformed into the convolution whose kernel is 1×1; The front layer is the fully connected layer of the convolution layer, which can be transformed into the global convolution where the convolution kernel is $h \times w$, where h and w are respectively the height and width of the convolution results of the front layer.

3.6.5 CNN Model Construction

Table 4 below is the proposed structure of our Convolutional Neural Network:

Layer	Output Shape	Param #
Conv2D	(126, 126, 128)	3584
MaxPooling2D	(63, 63, 128)	0
Conv2D	(61, 61, 256)	295168
MaxPooling2D	(30, 30, 256)	0
Conv2D	(28, 28, 256)	590080
Flatten	(200704)	0
Dense	(256)	51380480
Dense	(3)	771

Table 4: Construction of our CNN model

3.7 ResNet + SVM

3.7.1 Intuition behind ResNet

In the previous subsection, we employed a CNN model to make the classification case. As we mentioned previously, CNN is used to extract features from more complicated attributes. Moreover, the level of features is highly dependent on the number of layers [13]. Recent studies have found out that the network depth is of crucial significance and models tend to benefit from deeper layers [14].

However, one crucial question is: Is improving the accuracy of the model as easy as simply accumulating layers? The answer should be negative. As noticed by previous researches, as the depth of the network increases, adding more layers to the network might even cause training errors to increase, which is not related to overfitting. This is usually called a "degradation problem" [13]. This could be shown in the below figure [13]:

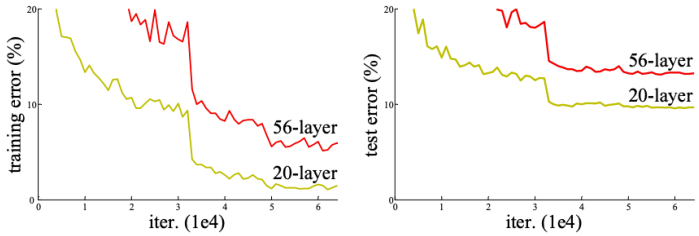

Figure 11: Degradation problem brought by layer stacking

Therefore, a method which could increase depth of networks while not causing degration problems is under consideration and that's basically the intuition behind ResNet.

The inspiration for the model is that, for a neural network with shallow layers, if we add new a layer of identity unit to the neural network, the depth will be increased while the training error will not increase. With the inspiration, Kaiming He et al. designed an algorithm of similar intuition [13].

3.7.2 ResNet

In the ResNet, different from common CNN models, there is a special unit called Residual Learning Unit. For a structure of stacked layer, it is employed to measure the residual between inputs and output, and its basic structure could be shown as below:

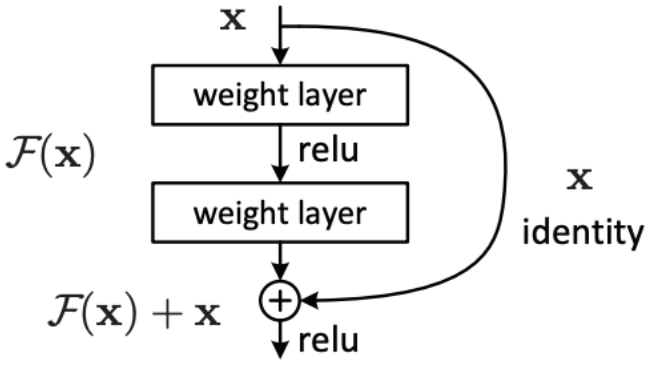

Figure 12: Structure of a Residual Learning Unit

For the unit, it is applied on a stack of layers. When the input is x, the learned output is denoted as $H(x)$. The residual $F(x)$ here would represent $H(x) - x$. Afterwards, instead of representing output with $H(x)$, we will replace it with $F(x) + x$. Therefore, a "shortcut" path is added to each unit. If the residual for a stack of the layer is extremely small or even 0, we could view it as an identity unit

which skips the complex computation of inside layers. This is actually quite similar to the "short circuit" in physics.

Alternatively, if we put it more mathematically, the idea will be much clearer.

Firstly, we could represent a Residual Learning Unit as:

$$y_l = h(x_l) + F(x_l, W_l) \tag{1}$$

$$x_{l+1} = f(y_l) \tag{2}$$

Where x_l and x_{l+1} represent the input and output of Residual Learning Unit l. F is the Residual Function and $h(x_l)$ represents a identity map. f indicates a ReLU activation function. Based on the above notations, we could get a learning feature from shallow layer l to deeper layer L:

$$x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i) \tag{3}$$

And if we use the chain rule to get the gradient backward, we will get:

$$\frac{\partial loss}{\partial x_l} = \frac{\partial loss}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial loss}{\partial x_L}(1 + \frac{\partial}{\partial x_L}\sum_{i=l}^{L-1} F(x_i, W_i)) \tag{4}$$

From the equation, we could see that even if the gradient of residuals is small, in view of the existence of the term 1, it is unlikely that the gradient will vanish. Therefore, residual learning is an easy and safe way to increase the depth of networks.

Then we will directly add our Residual Learning Unit to the original Neural Network. For a deep neural network, a Residual Learning Unit will be added every two or three layers.

3.7.3 ResNet+SVM

As we proposed in the pilot test, the biggest drawback for common machine learning models during image classification is poor features extraction. With the development of Deep Neural Networks, we could conduct a feature extraction phase by automatic processing [15], which could be directly applied to data without human intervention. Also, replacing the last dense layer, which is usually activated by Softmax, could possibly improve the performance of the original neural network.

Therefore, for the second model, we employed a model that combined two frameworks: Firstly, we employed the ResNet to extract sufficient features from the images. Then, we applied a Support Vector Machine classifier on the obtained features to give them clear classifications.

For our model, the input size will be $128 \times 128 \times 3$d and the output will be a vector of 8192 features. Altogether, there are 177 layers, including Zero Padding Layers, Convolutional Layers, Pooling Layers, Batch Normalization Layers, **Residual Learning Layers**, and Dense Layer. For our model, the batch size is 64, the number of epochs is 40 and the learning rate is 0.001. The detailed structure could be seen from our codes.

4 Results

4.1 Results of CNN

By using the CNN model we constructed before hand and set the epoch to be 10, the final results we get are shown as follows:

From the results above, it is proper for us to say that CNN has done a quite good job in classifying people wearing masks or not and wearing masks correctly or not. The final accuracy we get for the training set is 95.14% and 91.87% for the testing set (see **Table 5**). As shown in **Table 6 and Figure 13**, we can conclude that the model has a good ability in classifying the three different classed. I

Accuracy	Value %
Training accuracy	95.14
Validation accuracy	91.87

Table 5: The number Accuracy of the CNN model

True label/Predict label	Incorrect	Yes	No
No	137	4	12
Yes	6	250	7
Incorrect	15	7	189

Table 6: Confusion matrix for CNN

mean that the model does have the classifying capability which is different from the SVM model we did on the original biased data set. Recall the result of the pilot test, the accuracy of SVM was also not bad (88.67%); however, the confusion matrix shows that model did badly in classifying data with labels without_mask and mask_wore_incorrect. Now, CNN shows a great improvement in classifying all the data. We can conclude that the convolutional neural network we construct for this classification is quite satisfactory.

4.2 Results of ResNet + Support Vector Machine

Before running the SVM classifier, we first ran all 40 epochs of the ResNet to get the optimized dense layer we needed. To measure the performance of our ResNet model, we used the loss and accuracy of both the training set and test set.

Table 7 is a table describing the result of the ResNet we trained after 40 epochs.

From the table, we can see that we have already trained excellent features from our ResNet.Therefore, we could employ the trained features on a new Support Vector Machine model. Here the input of the SVM is the vector of 8192 features we get from the ResNet model, and the output is the labels of the image, which are "with mask", "without mask", and "wearing masks wrongly". After multiple attempts, we chose a Gaussian kernel and a regularization factor of 10.

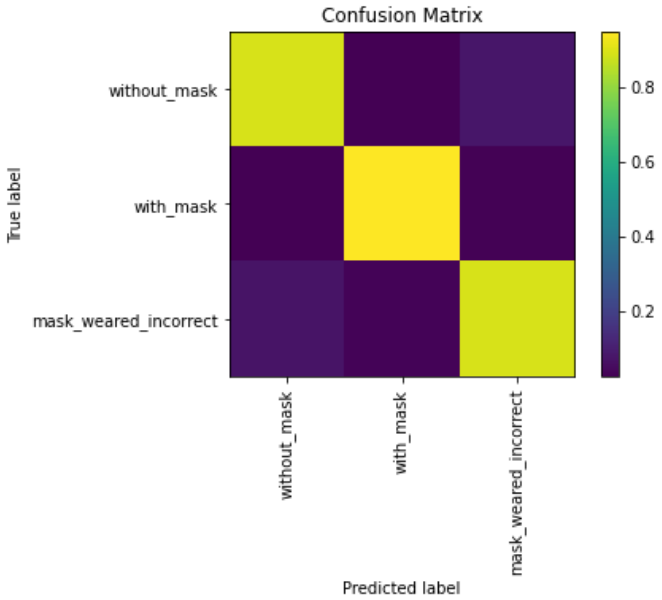

Figure 13: Confusion Matrix Plot for CNN

Metric	Value
Training Loss	3.1201
Training Accuracy	0.9568
Test Loss	3.2192
Tess Accuracy	0.9203

Table 7: Performance of ResNet

After running the model, we got an accuracy rate of 94.26% for the test set, which is higher than any employed model before. Besides, we could show the confusion matrix we got from the outcomes (see **Table 8 and Figure 14**).

True label/Predict label	Incorrect	Yes	No
Incorrect	100	4	3
Yes	6	140	3
No	6	2	154

Table 8: Confusion matrix for ResNet + SVM

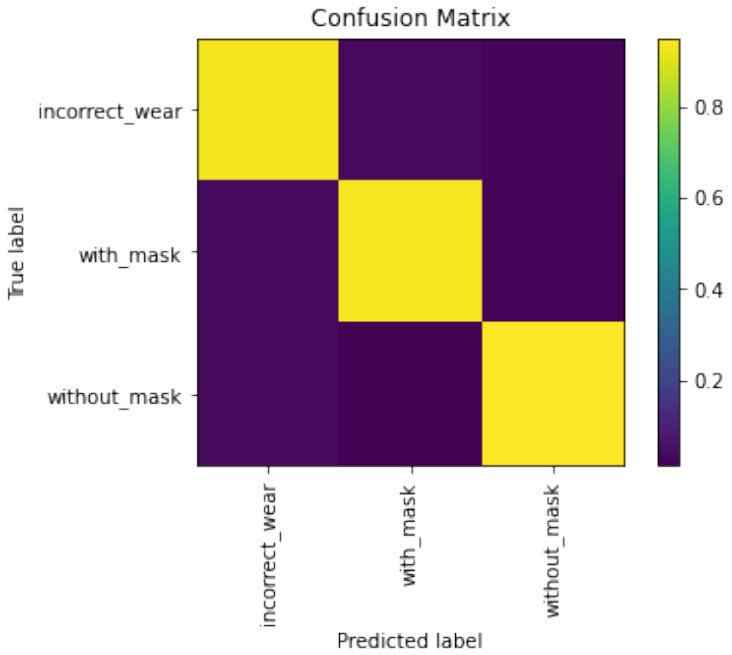

Figure 14: Confusion Matrix Plot for ResNet + SVM

From the result, we could see that for for each group, the majority of the data is correctly classified and there are no observed biases.

Then we are going to make a comparison of all results.

4.3 Comparison of Results

Then we are going to give a comparison to all the results in the Pilot Test, CNN Model and ResNet + SVM Model. Here, we not only employed the accuracy as the metric but also used other metrics, including $Recall, Precision, F1Score$. Here since we have three classes, we have to make some little adjustments to the classes so that these metrics can be calculated. Here, since we are caring more about those who are not wearing masks or wearing masks improperly, we will assign them as Class 0 while assigning those wearing masks correctly as Class 1. Below is a table showing their differences:

Model	Accuracy	Precision	Recall	F1 Score
SVM	88.67%	86.2%	45.4%	59.5%
CNN	91.87%	96.2%	89.6%	92.8%
ResNet	94.26%	96.6%	94.4%	95.4%

Table 9: Comparison of Results

Also, we could compare their confusion matrix (see **Figure 15, Figure 16 and Figure 17**):

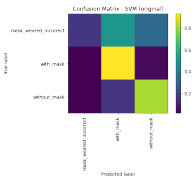

Figure 15: SVM Confusion Matrix

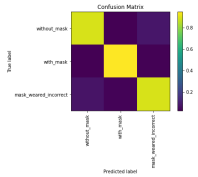

Figure 16: CNN Confusion Matrix

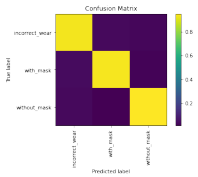

Figure 17: ResNet Confusion Matrix

From the given table, we could find out that our ResNet + SVM model outperforms the other two algorithms in all metrics, especially in the recall rate. Also, from the figures, we could see that ResNet tend to have an excellent performance in any class and have little bias. Therefore, we could see that by combining well-tuned SVM and ResNet algorithms, we could achieve a desirable model.

5 Discussions

5.1 Experiences Learned

From the above results, we found out that the ResNet + SVM model has the best performance compared to other mentioned models. Then, one important question that we are focusing pn here is: Why the model could outperform other models? If we learned about this in the project, the experiences could be transferred into future studies. Looking back on the whole procedure, we came up with a few critical points:

1. Avoid biased datasets. When we have too biased datasets, meaning that the ratio of one class is much higher than the others, our model tends to have a bias towards one certain group. Usually, the accuracy would be high for a model trained upon a biased model. However, we could not rely on the accuracy too much. For example, simply predicting that every item belongs to the class with most samples will always ensure high accuracy. If we look at the confusion matrices or other metrics, the results might not be satisfactory. Therefore, we enlarged our dataset and made it more balanced.
2. Improve the variability and adaptability of our datasets. In the last point, we mentioned that the labels should be balanced in order to make results more reliable. Also, the attributes of our samples should have great variability so that the model will not be an overfitting one. These could be achieved in two ways. On the one hand, while collecting our data, we should pay attention to the randomness of our data, meaning that we should cover faces of all races, genders, ages and so on. On the other hand, after getting the data, we could employ some data augmentation strategies to slightly adjust the location, rotation, brightness or color. These could make our model more adaptable.
3. Extract features more wisely. One reason that our pilot didn't have satisfactory performance is that the feature extraction is too hasty. To fix this, we employed some deep learning models to extract features of different levels comprehensively. Sometimes the process of feature engineering is even more important than model selection.
4. Combine deep learning frameworks with machine learning models. Although deep learning could perform well while extracting features, they are not necessarily excessive while classifying objects. Therefore, we combined the wisdom of some machine learning methods with beautifully extracted features to reach the outcome of "1+1>2". And from the fact that

ResNet + SVM outperforms ResNet singly, we could assume that such a combination is reasonable.

5. Make a more detailed classification. Another advantage of our model is that we give more classes to our outcomes apart from a binary one. These might make training more complicated but could adapt to reality more as well.

5.2 Drawbacks and Future Work

Although we made some breakthroughs or innovations in the study, it is undeniable that there are still many things we could improve in the future. Below are some drawbacks and things we could do to fix them:

1. For the Haar Cascade Classifier, to save time, instead of training a model on ourselves, we employed a pre-trained setting. However, these could not ensure that the outcomes fit into our circumstances perfectly. For example, most faces in our dataset have masks in front of them, which could be different from the dataset used for the pre-trained model. Therefore, next time, we should attempt to train the face detection model based on our data.
2. For the augmentation of the dataset, we just replaced the original images with some slightly adjusted ones. However, we could make different adjusted copies for an image to make our dataset more adaptable and large in size. These could possibly help to improve test performance.
3. For our model, although we included a class called "wearing masks wrongly", the actual situation would be much more complicated. For example, there are different types of wrong ways to wear masks. Some people have their noses outside the masks, some might cover their glasses, and some might have bad lateral adjustment [7]. Therefore, to be more specific, we could further classify the current labels and apply them to real-life.
4. Another thing we could add on the model itself is a GUI for real-time detection. This could be built either on laptops or mobile devices. With such a procedure, our model could be altered from something theoretical into a real useful application.

6 Conclusion

For this project, in order to find the most reliable models, we tried different ways of data collection, feature extraction and multi-class classification. We were temporarily managed to build a model that has a rather impressive performance. By fixing some of our limitations, we could improve the feasibility of the algorithm. Then we can apply this model to real-life scenarios, help monitor whether people are wearing masks correctly and give corresponding warnings. By doing so, we could help to prevent the prevalence of Covid-19 to some extent.

References

[1] “Who Coronavirus (COVID-19) Dashboard.” World Health Organization. World Health Organization. Accessed October 17, 2021. https://covid19.who.int/.

[2] Feng, Shuo, Chen Shen, Nan Xia, Wei Song, Mengzhen Fan, and Benjamin J Cowling. “Rational Use of Face Masks in the COVID-19 Pandemic.” *The Lancet Respiratory Medicine* 8, no. 5 (2020): 434–36. https://doi.org/10.1016/s2213-2600(20)30134-x.

[3] Sethi, Shilpa, Mamta Kathuria, and Trilok Kaushik. "Face Mask Detection using Deep Learning: An Approach to Reduce Risk of Coronavirus Spread." Journal of Biomedical Informatics 120, (2021): 103848-103848.

[4] J. Redmon, S. Divvala, R. Girshick, A. Farhadi, You only look once: Unified, realtime object detection, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2016, vol. 2016-Decem, pp. 779–788, doi: 10.1109/CVPR.2016.91.

[5] Girshick R., Donahue J., Darrell T., Malik J. Region-based Convolutional Networks for Accurate Object Detection and Segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 2015;38(1):142–158. doi: 10.1109/TPAMI.2015.2437384.

[6] Cai Z., Fan Q., Feris R.S., Vasconcelos N. A unified multi-scale deep convolutional neural network for fast object detection. Lect. Notes Comput. Sci. (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 2016 doi: 10.1007/978-3-319-46493-022.

[7] Tomás, Jesús, Albert Rego, Sandra Viciano-Tudela, and Jaime Lloret. "Incorrect Facemask-Wearing Detection using Convolutional Neural Networks with Transfer Learning." Healthcare (Basel) 9, no. 8 (2021): 1050.

[8] Viola, P., and M. Jones. “Rapid Object Detection Using a Boosted Cascade of Simple Features.” *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, n.d. https://doi.org/10.1109/cvpr.2001.990517.

[9] Wilson, P. I., and Fernandez, J. "Facial feature detection using Haar classifiers." *Journal of Computing Sciences in Colleges*, vol. 21, no. 4, pp. 127-133. 2006.

[10] Open Computer Vision Library Reference Manual. Intel Corporation, USA, 2001.

[11] "What is image augmentation - Albumentations Documentaiton" https://albumentations.ai/docs/introduction/imageaugmentation/

[12] Lecun, Y., L. Bottou, Y. Bengio, and P. Haffner. “Gradient-Based Learning Applied to Document Recognition.” *Proceedings of the IEEE* 86, no. 11 (1998): 2278–2324. https://doi.org/10.1109/5.726791.

[13] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition." (2015).

[14] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In ICLR, 2015.

[15] Hend, B., Ouarda, W., Sayadi, F. E., and Ouni B. (2020). "CNN-SVM learning approach based human activity recogntion." Research Gate. DOI: 10.1007/978-3-030-51935-329.