BT4221: Big Data Techniques and Technologies
Semester 1 AY 20/21

COVID-19 Face Mask Detection

Final Report

Group 11

Ernest Chng Yong Cheng A0189158M
Luke Izumi Lau Chuan A0190047J
Jolene Tan Shimin A0190385Y
Dylan See Keane Loong A0183045J
Lee Kai Ming, Jabez A0183817W

Table of Contents

1 Topic

1.1 Introduction

The aim of this project is to create a face mask detector that accurately determines if an individual is wearing a face mask. In addition, we aim to provide supplementary information by predicting the individual's age as well as gender.

This report begins with a detailed description of our group's motivation and objectives. Next, it covers our data collection, feature selection and model development. Our results are then presented and analysed accordingly. Finally, our group presents our findings and conclusion.

1.2 Motivation

As we all know, COVID-19 has been the biggest topic of 2020. It is a highly contagious disease that caught the world by surprise and has made its way into almost every single country on the world map. Since its outbreak in China, the number of cases has been constantly on the rise and was declared a pandemic by WHO on 11 March 2020. As of 28 November 2020, the virus has infected more than 60 million individuals worldwide and resulted in 1.45 million deaths.

The severity of the virus called for stricter regulations by governments all around the world in hope to curb its spread. According to the CDC (Centers for Disease Control and Prevention), 50% of transmissions happen before individuals start to develop any symptoms of COVID-19. In addition, based on research, the use of masks were found to reduce transmissions by 70% if worn the right way (Washington-Harmon, 2020). As a result, a key guideline recommended by WHO was the use of mandatory face mask coverings. Not only will wearing masks help prevent the spread of infection, it is deemed as a good hygiene practice to adopt that will protect people from germs and illnesses.

In the context of Singapore, the government has mandated everyone to wear a mask upon leaving their place of residence by law and individuals who do not abide will face a fine of S$300. Since its enforcement, most citizens have abided by the rule. However, over the course of the year, there have been more than 100 cases where people choose not to wear a face mask in public and flout the face mask rule. The current system in place is to deploy government officers to conduct inspections, detect non-compliant individuals and issue these individuals with a warning or fine (National Environmental Agency, 2020). However, this process of monitoring whether face masks are being worn has been extremely costly.

In addition, it has been found that there may be several groups of people (e.g. males in their 20's) that have a higher tendency to choose not to wear masks due to various reasons as well (Jarry, 2020). These groups of people might be of a certain age group or gender, and it is important to be able to accurately identify these groups of people as well.

With these in mind, our team has come up with the idea to automate the identification process of individuals who are non-compliant to the mask regulations. Our solution is to create a face mask detector that identifies whether an individual is wearing a face mask. In addition, this detector will also aim to detect a person’s age and gender in order to scope down the tropes of people who adhere to or flout the mask regulations.

1.3 Objective

This project aims to help companies reduce costs. The realistic alternative is to hire human screeners to ensure that people wear masks, which is unfavourable because it is resource-intensive. The detector would ease the logistics for companies.

More importantly, non-compliant individuals pose great risk to the health, well-being and operations of a company. Thus, our face mask detector aims to be a deterrent to non-compliance. It will be a simple and effective way to ensure greater compliance. With such a system in place, employees will be more mindful and there is less chance that they will flout the rules.

In addition, we collect data on the age and gender of people who are non-compliant for companies to consider other relevant directives. For instance, with this data, we are able to find out more behind the reasons why certain groups of people have a higher rate of non-compliance compared to other groups, be it psychological reasons or societal reasons. Companies will then be able to target their education of the importance of wearing masks at these groups and to clarify any misconceptions that these groups of people might have about wearing masks.

1.4 Use Case in Offices

The main use case for our detector would be for offices. We want to place these detectors in areas of high human interaction, such as in the lobbies of office buildings or in meeting rooms to ensure that people adhere to wearing a mask at all times where possible.

With more people returning to offices after long work-from-home stints, offices can expect increased population densities and close-contact interactions. Employees should not have to be exposed to health-threatening dangers at work. They have the right to be working in clean, safe and conducive environments where they can perform.

As covered extensively above, employers have much to benefit from this too, especially in operations and resource allocation considerations. All in all, offices are very fitting to be using such a technology.

2 Data Collection

The process of data collection was required to train our model to differentiate between images of individuals with and without masks. Our team made use of three primary channels: collecting open-source datasets from Kaggle, web-scraping through image databases from Google and creating customised images.

2.1 Open-source datasets

Kaggle is an online community of data scientists and machine learning practitioners. It is a common place for users to find and publish data sets that can be used for machine learning and dataset. Our team was able to find the main bulk of our training images from open source datasets from Kaggle. We manually screened through various relevant repositories and combined them into a single dataset. In total, we were able to collect around 1800 images from Kaggle.

2.2 Web-scraping

To expand our dataset and explore images beyond what is available in open-source datasets, we created our own web scraper in Python that scrapes through image databases from Google based on search queries.

The following describes the steps of web-scraping:

1. Launch a WebDriver

   Apart from installing the 'Selenium' library to automate the web browser interaction process, we had to identify our Chrome version and install the corresponding ChromeDriver to use Selenium. Through this, we were able to pair Selenium with Google Chrome to automate the fetching of images.

2. Specify search query to extract image links

   Using the ChromeDriver to navigate via selectors and pages, we created a function that would search for the query specified, move into the image section and get the links of the images. The function takes in the following 3 input arguments:

   - query: the search term
   - max_links_to_fetch: integer input that defines the number of images to scrape.
   - webdriver: the instantiated Web Driver

3. Download the images with the image links

   To download the images from the extracted image links, we used the 'Pillow' module. We created another function that would download the images and save the downloaded images into organised folders based on query. The function takes in the following 3 input arguments:

- folder_path: directory where all images are saved
- file_name: name of file to store images
- url: list of image links from step 2

With the above web scraper, we explored and downloaded the images using search queries such as “faces” and “faces without mask”. In total, we were able to retrieve around 500 images through web-scraping.

2.3 Custom images

From our initial data collection via the above two methods, we realised that images of faces with masks were limited. This led us to explore other ways to expand our dataset further such as creating our own images through the following two methods:

1. Super imposing masks onto images with faces.
2. Image augmentation which creates modified versions of images by rescaling and applying translations and rotations.

We also used image augmentation with a broader purpose in mind, to increase the diversity of images available for our model so as to increase its generalisability and reduce overfitting eventually.

At the end of this data collection process, we were able to accumulate up to more than 2500 images, where 50% of the images consist of individuals with masks and the other 50% of images consist of individuals without masks.

3 Data Exploration and Preprocessing

3.1 Data exploration

Upon consolidating the images that we collected via the above data collection methods, we performed data exploration to generate useful insights from our data.

Through exploring the data, it came to our attention that there were several issues with the data we collected, such as:

1. Images of the wrong target audience

   Given that the use case of the face mask detector is in offices, our target audience is working adults. This required us to ensure our dataset excludes images of babies and children. However, we found that there were images in our dataset that did not meet these exact requirements. Additionally, there were cartoon images in our dataset as well, which do not provide an actual and accurate representation of humans and will affect our model's prediction accuracy. We attribute this issue mostly to the Google images that were web scraped as it is not possible that the Google search queries provide exactly all the images that are relevant to us.

2. Duplicated images

   Moreover, we found duplicate and near-duplicate images in our dataset. This was expected as our dataset was sourced from multiple data sources, which could have been acquired from the same source without our knowledge.

3.2 Data cleaning & preprocessing

As we are familiar with the idea of GIGO, which is short for "garbage in, garbage out", data cleaning and preprocessing is an important step to ensure that the data our model trains on is accurate, consistent and useful enough for the model to learn from. Thus, knowing the above issues, we worked on cleaning and preprocessing the dataset to address those issues before fitting them into our model for training.

To resolve the first issue where our dataset consisted of images with the wrong target audience, we manually sifted through and removed the irrelevant images that do not fit our use case.

Subsequently, to remove duplicated images, we used a command line tool called 'imgdupes' that easily searches and deletes duplicate and near-duplicate images. 'imgdupes' uses the 'ImageHash' library to calculate the perceptual hash by analyzing the image structure on luminance. This finds sets of near-duplicate images with Hamming distance of phash less than 4 from the target directory and by default, displays the list of duplicate images. To delete the duplicate images, we used the following command line:

```
imgdupes --recursive --delete --imgcat <image-directory> phash 4
```

Finally, with the cleaned data set, we converted the images from BGR to RGB and they are rescaled to 256 × 256 and a crop of 227 × 227 before converting them into array format and scaling the pixel intensities in the input image to the range [-1, 1], in preparation for feature selection.

At this stage, we finalised the data with which the neural network would work with. We managed to accumulate 2000 images, with an equal proportion of target individuals with and without masks.

4 Feature Selection

Feature selection refers to the process of selecting a subset of features that are relevant for model construction. The notion of feature selection works on the premise that the data contains certain features that are either redundant or irrelevant in the image classification problem. As such, removing these features would not result in a significant loss of information (Zhou X and Wang J, 2015).

Generally, a simplified model that can yield comparable classification accuracy is preferred over a complex model. Thus, reducing the number of features to be included in the model can help in developing a simpler model that is more interpretable and easier to be understood by the general populous. In addition, we can avoid the overhead costs of longer training time and also improve generalisation by reducing the possibility of overfitting. Thus, it is important to conduct feature selection prior to model development.

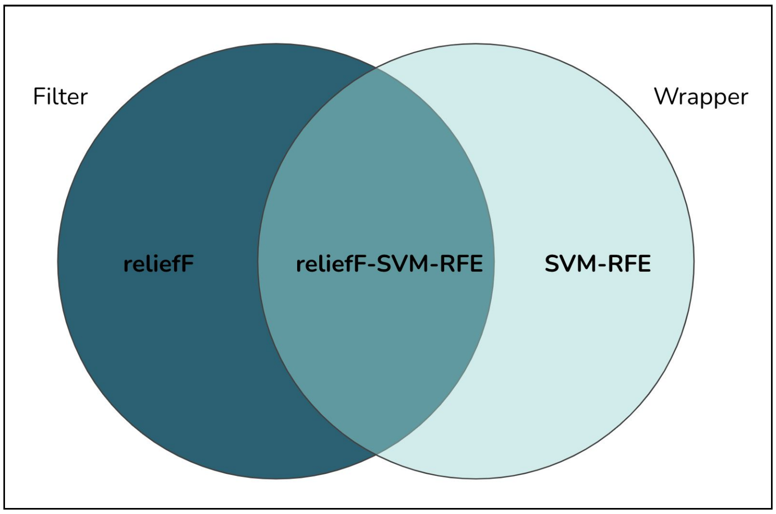


Figure 1: Variants of Feature Selection Algorithms

In this project, we explored two variants of feature selection algorithms: reliefF and SVM-RFE. Based on different evaluation criteria, the algorithms can be separated into two categories: filter and wrapper methods. The two methods differ in terms of the logic for their evaluation process. Filter methods work by evaluating the relationship between features and the outcome variable to compute the importance for each feature. On the other hand, wrapper methods work by measuring the “usefulness” of features based on the classifier performance.

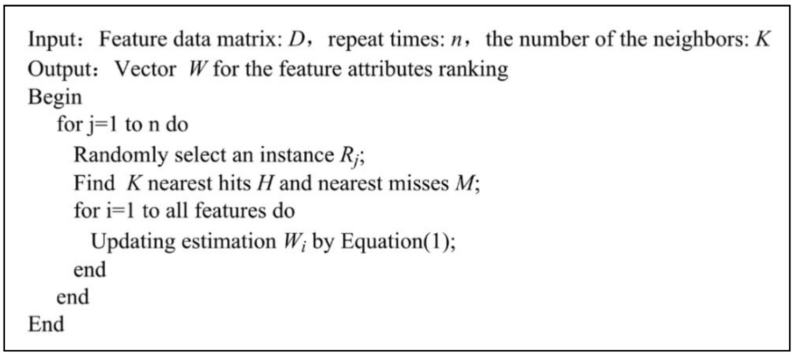
In general, wrapper methods are able to achieve a higher prediction performance as compared to that of a filter method. Such superiority is attributed to the lack of precise tuning to a specific model in filters that produces a more generalised feature set. However, while we acknowledge the better classification accuracy achieved, wrapper methods are also more computationally intensive.

To harness the advantages of both filter and wrapper methods, we decided to explore the reliefF-SVM-RFE algorithm which is a hybrid approach. This approach integrates steps from both the reliefF (filter) and SVM-RFE (wrapper) methods. In the first phase, the reliefF algorithm is utilised to find an initial subset of features. Here, many redundant features are first filtered out to help reduce the computational demands. In the second phase, SVM-RFE is conducted to directly evaluate the importance of each feature with respect to the Support Vector Machine classifier.

To evaluate the performance of the three algorithms, we evaluated their classification accuracy over the number of features to be included and compared the average classification accuracy. We also compared their computational time required. Ideally, an algorithm that achieves a high prediction performance and is computationally efficient would be chosen.

4.1 reliefF

The reliefF algorithm is a K-nearest neighbours-based feature selection algorithm that is part of the filter family in feature selection. It evaluates the importance of each feature according to how well it is able to distinguish between instances that are within close proximity.



```
Input：Feature data matrix: $D$，repeat times: $n$，the number of the neighbors: $K$
Output：Vector $W$ for the feature attributes ranking
Begin
  for j=1 to n do
    Randomly select an instance $R_j$;
    Find $K$ nearest hits $H$ and nearest misses $M$;
    for i=1 to all features do
      Updating estimation $W_i$ by Equation(1);
    end
  end
End
```

Figure 2: Pseudo-code for reliefF algorithm

The algorithm works by randomly selecting an instance R_i from its class. Next, it searches for K of its nearest neighbours from the same class (called nearest hits H) and K of its nearest neighbours from each of the different classes (called nearest misses M). Using that information, it updates the importance measure W_i for feature i according to values for R_i, hits H and misses M. If a feature importance difference in value is observed between R_i and its nearest hits H, then the importance measure W_i is decreased. On the other hand, if a feature importance difference in value is observed between R_i and its nearest misses M, then the importance measure W_i is increased. This process is repeated N number of times until the feature importance measure converges.

As mentioned above, the importance measure W_i is as follows:

$$W_i = W_i - \frac{\sum_{k=1}^{K} D_H(k)}{n \cdot k} + \sum_{c=1}^{C-1} p_c \cdot \frac{\sum_{k=1}^{K} D_M(k)}{n \cdot k}$$

Figure 3: Importance Measure W_i

where $D_H(k)$ (or $D_M(k)$) is the sum of distance between the selected instance and its kth nearest neighbour in H (or M), p_c is the prior probability of class C.

4.2 SVM-RFE

The SVM-RFE algorithm is a wrapper method that evaluates the importance of features based on the performance of a Support Vector Machine classifier. It utilises a recursive feature elimination method that is basically an iterative backward removal of features. This involves including all the features in the beginning and recursively removing the features deemed the least important for classification in a backward elimination manner.

The following describes the weight vector of the SVM:

$$\boxed{w = \sum_{l} \alpha_l \gamma_l \chi^{(l)} .}$$

Figure 4: Weight vector of SVM

where α_l is the Lagrange multiplier, γ_l is the class label of the lth sample, and □$^{(l)}$ is the feature vector.

Using this weight vector, we can compute the importance of the ith feature by taking $C_i = w_i^2$ where C is the ranking criterion.

4.3 reliefF-SVM-RFE

The reliefF-SVM-RFE is a hybrid approach that uses both reliefF and SVM-RFE in its computations. It is a two-stage process that uses the reliefF algorithm in the first step and the SVM-RFE in the second step. We have established that reliefF is generally effective in providing a good estimate of feature importance, however it does not consider the classifier to be used in feature selection. On the other hand, the SVM-RFE algorithm tackles this issue but is computationally expensive.

To take advantage of both methods, the reliefF-SVM-RFE algorithm combines the weight vector W_i from reliefF and C_i from SVM-RFE to get a new ranking criterion G_i. The computation for G_i is as follows:

$$\boxed{G_i = W_i + C_i .}$$

Figure 5: New ranking criterion G_i

where C_i is the ranking indicator from SVM-RFE method. Note that the two weight vectors are normalised to the range between 0 and 1 inclusive which can help to eliminate the error resulting from the two different dimensions.

Therefore, G_i is the final ranking indicator to be used by the reliefF-SVM-RFE algorithm.

4.4 Evaluation performance

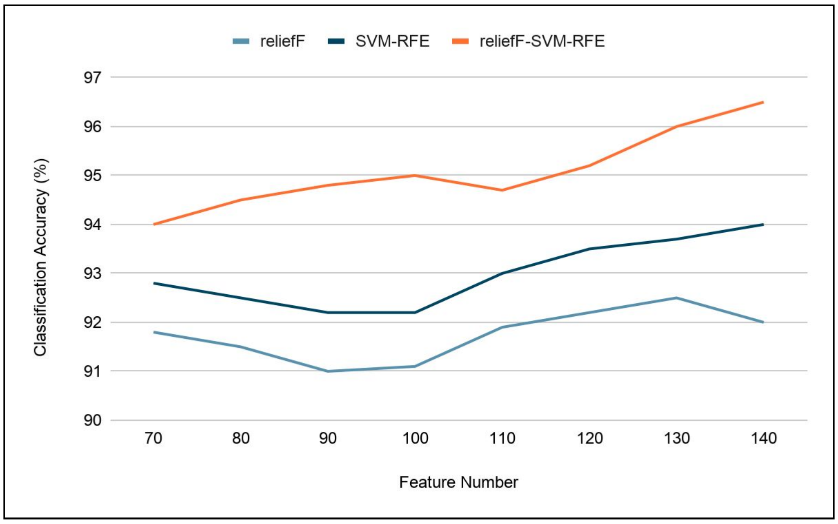


Figure 6: Classification accuracy of the 3 feature selection algorithms over the number of features included for each class

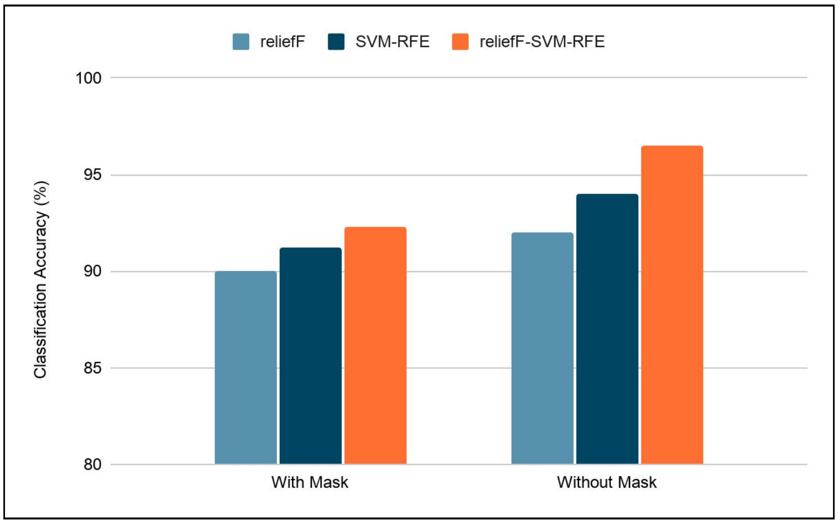


Figure 7: Average classification accuracy of the 3 feature selection algorithms for each class

Selection Algorithm	Computational Time (s)
reliefF	248
SVM-RFE	43,255
reliefF-SVM-RFE	3403

Figure 8: Comparison of Computational Time taken for the 3 feature selection algorithms

From experimentation, we observed that the hybrid approach (relief-SVM-RFE) obtained the best average classification accuracy for each class amongst the other two algorithms included.

As for computational time, reliefF-SVM-RFE performed second best with 3403s taken. It has significantly reduced the amount of time required when comparing solely with SVM-RFE. While we can note that it did not beat the reliefF in computational efficiency, we have already established that filters are inherently more efficient than wrapper methods. As such, it is not surprising that we obtained a computational time that took longer as compared to reliefF.

In conclusion, considering the fact that reliefF-SVM-RFE outperformed the other two algorithms in average classification accuracy and also significantly reduced the amount of computational time taken, we have chosen to use reliefF-SVM-RFE algorithm as our choice for feature selection.

5 Model Description

This project aims to achieve the following objectives - To detect if an individual is wearing a face mask, and subsequently identifying their age and gender. Our main approach involved the training of two separate models to identify the following 3 variables (face mask, age and gender). In the training process for our classifier, we utilised transfer learning and built our own CNN architecture on top of the pre-trained model. Our choice to do so was to reduce the training time and yet achieve a high performance and accuracy rate. We made use of Convolutional Neural Networks (CNN) as it is commonly used in image classification and is very effective in reducing the high dimensionalities of images without loss on the quality of the model.

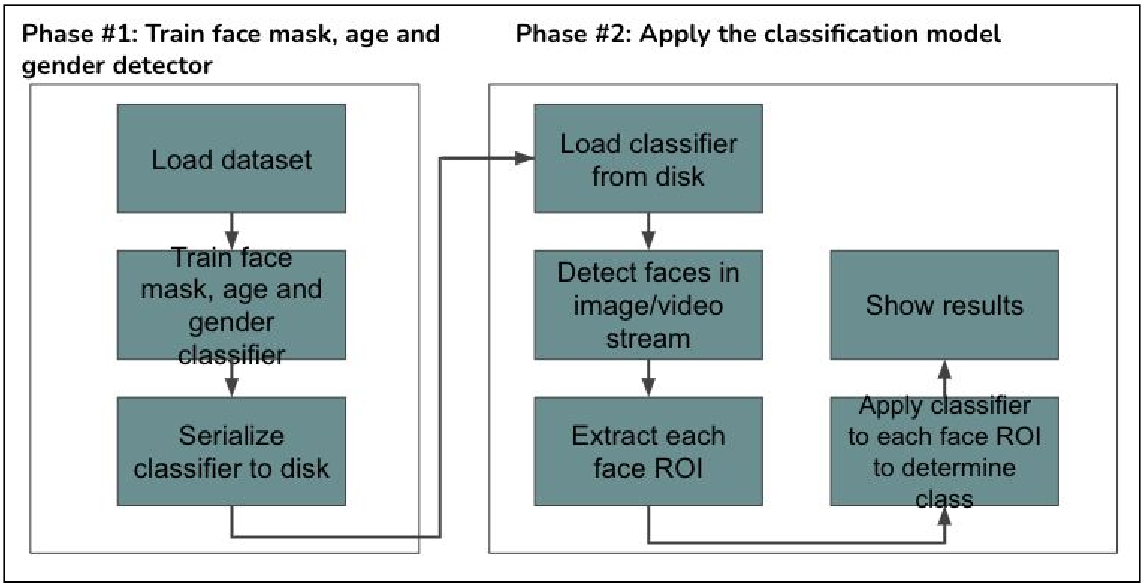

Phase #1: Train face mask, age and gender detector
Phase #2: Apply the classification model
Load dataset
Train face mask, age and gender classifier
Serialize classifier to disk
Load classifier from disk
Detect faces in image/video stream
Extract each face ROI
Apply classifier to each face ROI to determine class
Show results

Figure 9: Model breakdown

The training process can be broken down into two different phases - Training of our classifier and model application (both on videos and images as inputs).

5.1 Face mask detection model

The Face mask detection model is made by fine-tuning a MobileNet V2 architecture that is designed as a convolutional neural network. As a relatively lightweight model, it performs well on devices with limited computational capacity. Thus, making it a perfect choice for us since we had computational limitations with our own local machine. The MobileNetV2 is based on an inverted residual structure where the residual connections are between the bottleneck layers and the intermediate expansion layer uses lightweight depthwise convolutions to filter features as a source of non-linearity.

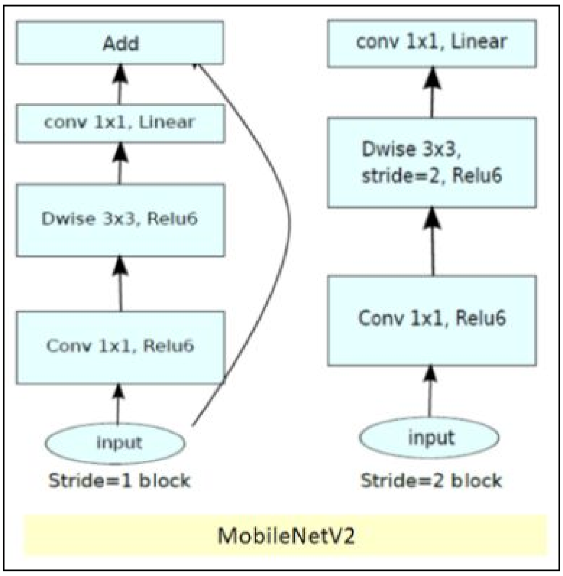
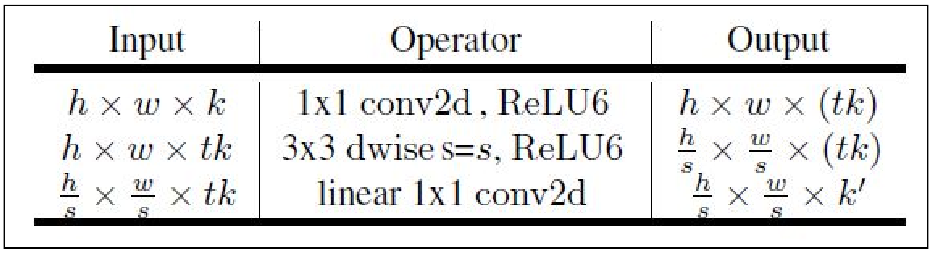


Figure 10: MobileNet V2 Architecture

In the MobilenetV2 architecture, there are two types of blocks - one is a residual block with stride of 1 and the other is a block with stride of 2 for downsizing. There are 3 layers for both types of blocks. The first layer is a 1x1 convolution with ReLU6, the second layer is a 3x3 depthwise convolution layer with ReLU6 and the third layer is a 1x1 convolution without any non-linearity. Dropout and batch normalisation are also used during training. (S. Mark, H. Andrew, Z. Menglong, Z. Andrey and C. Liang-Chieh, 2018)



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	$\frac{h}{s} \times \frac{w}{s} \times k'$

Figure 11: Input and Output of Model

5.1.1 CNN model

The network comprises three convolutional layers and two fully-connected layers with a small number of neurons.

The following describes the layers involved:

1. Pooling Layer: Average Pooling
   a. Allows the network to keep its translation invariance and helps prevent overfitting
   b. Calculates the average for each patch of the feature map and converts it into a single value by taking the average.
2. Fully connected layer: Flatten
   a. Flattens the input by transforming a multidimensional vector into a single dimensional vector
   b. Fed into the neural network (fully connected layer) to generate the prediction.
3. Dropout layer: Dropout
   a. Dropout = 0.5
   b. Dropout works by randomly setting nodes to be dropped-out with the given probability of 50%.
   c. Applies dropout to the input and is used to prevent our model from overfitting.

5.1.2 Model evaluation

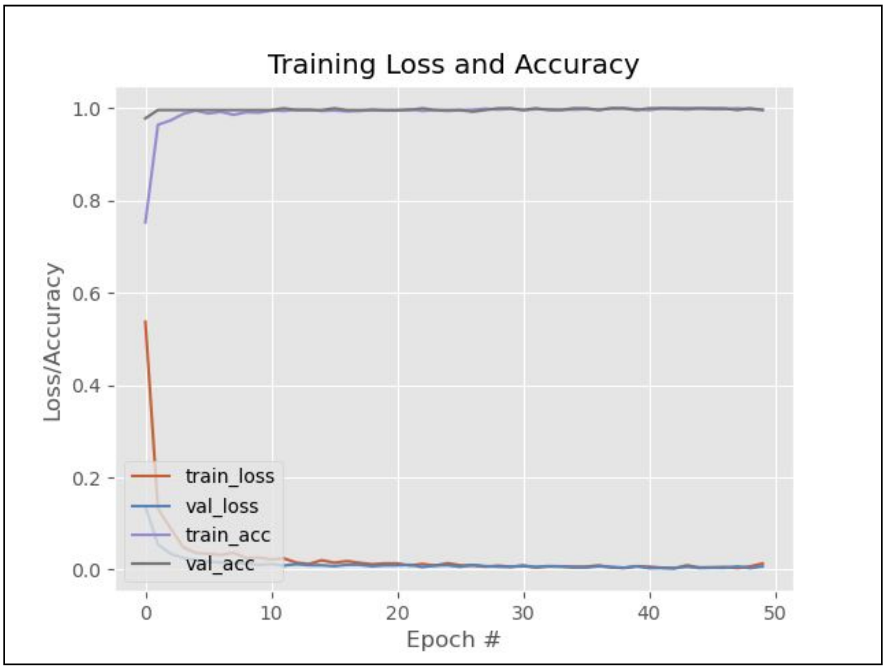


Figure 12: Plot of Accuracy/Loss vs Number of Epochs of Face Mask Detector Model

Our choice of epoch number was 50 for this model as we were utilising transfer learning and did not want to risk overfitting. From Figure 12, we are obtaining close to ~98% accuracy on our training and validation set. As there is no significant difference in accuracy between the training and validation set, there are little signs of overfitting. Our model should generalise well to images outside our training and validation datasets.

5.2 Age and Gender detection model

5.2.1 CNN model

The following describes the layers involved in the Age and Gender Detection CNN model:

1. First convolution layer:
    a. Contains 96 filters of size 3x7x7 pixels. This is followed by a ReLU, max pooling pooling and a local-response normalization (LRN).
2. Second convolution layer:
    a. Contains 256 filters of size 96x5x5 pixels, followed by a ReLU, max-pool and LRN to reduce the output size.
3. Third convolution layer:
    a. Contains 384 filters of size 256x3x3 pixels, followed by ReLU and a max pooling layer.
4. Fully connected layers:
    a. Fully Connected Layer 1: 512 neurons fully connected to the output of the third convolution layer, followed by a ReLU and dropout layer.
    b. Fully Connected Layer 2: 512 neurons fully connected to the 1x512 output layer of the first fully connected layer followed by a ReLU and dropout layer.

    c. Fully Connected Layer 3: The third layer yields and maps the un-normalized class scores to the final classes for age or gender.

5. Output layer:
    a. Softmax layer which assigns probability for each class and gives the loss and final class predictions.

5.2.2 Model evaluation

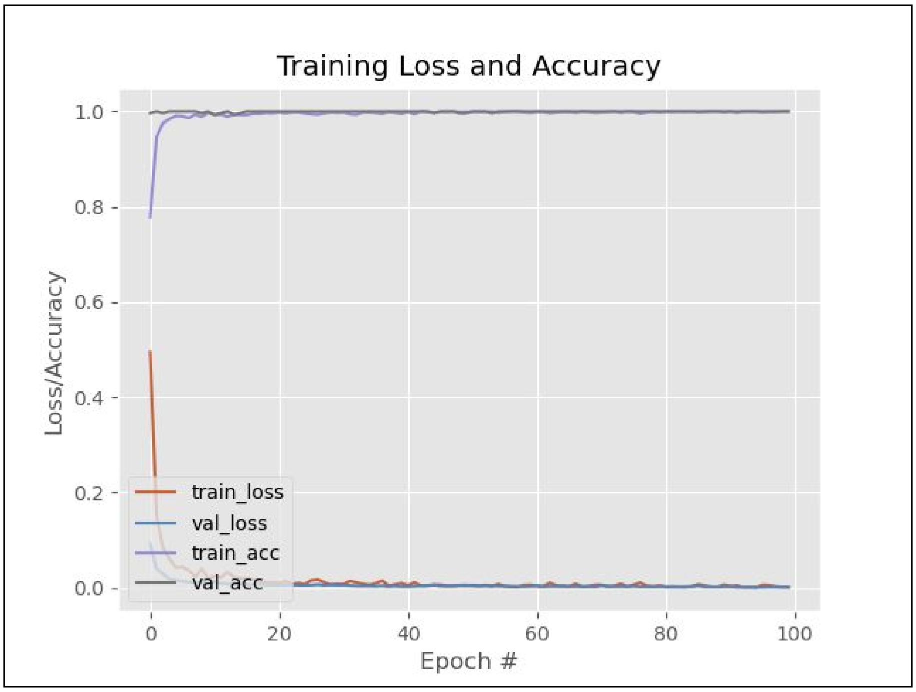


Figure 13: Plot of Accuracy/Loss vs Number of Epochs of Age & Gender Detection Model

Again, we can observe from Figure 13 that we are obtaining close to ~99% accuracy on our training and validation set. Since there is no significant difference in accuracy between the train and validation set, we conclude that there are little signs of overfitting. Our model should generalise well to images outside our training and validation datasets.

5.3 Experiments

During the training phase, we explored several different methods. Initially, our model was not performing as well as we would have wanted. We speculated that the unsatisfactory performance was attributed to the fact that our training images were not capturing different angles of an individual’s face. Particularly, this affects our video stream significantly where one would be facing the camera at various angles.

To achieve a better performance, we implemented further image augmentation to create modified versions of the images in our dataset. This allowed us to introduce different angles of an individual’s face into our dataset.

In addition, we cropped each image to be centered around the individual's face and reduced it to 227 x 227 pixels. This helped to reduce the amount of noise in our training data by filtering out unimportant data points. The implementation of these two techniques helped to improve the quality of our training and we were able to achieve a higher model accuracy rate.

6 Results

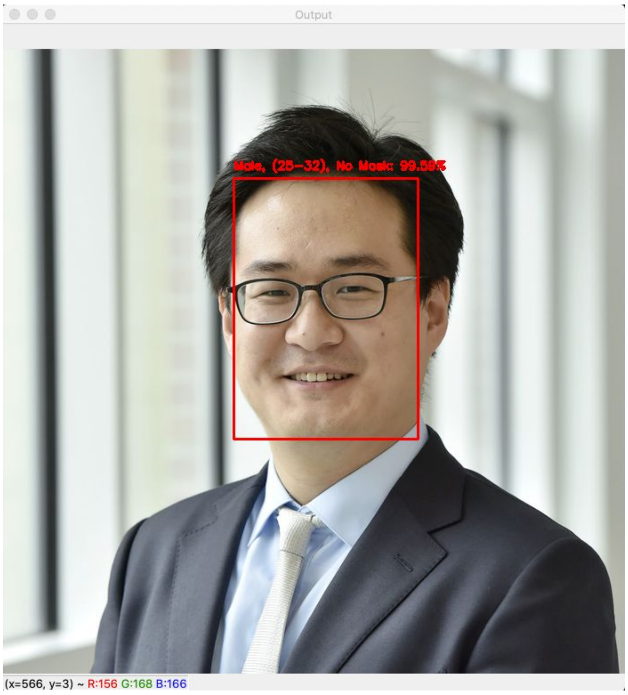

Output
Male, (25-32), No Mask: 99.58%
(x=566, y=3) ~ R:156 G:168 B:166

Figure 14: Face Mask Detector Model Sample Result 1

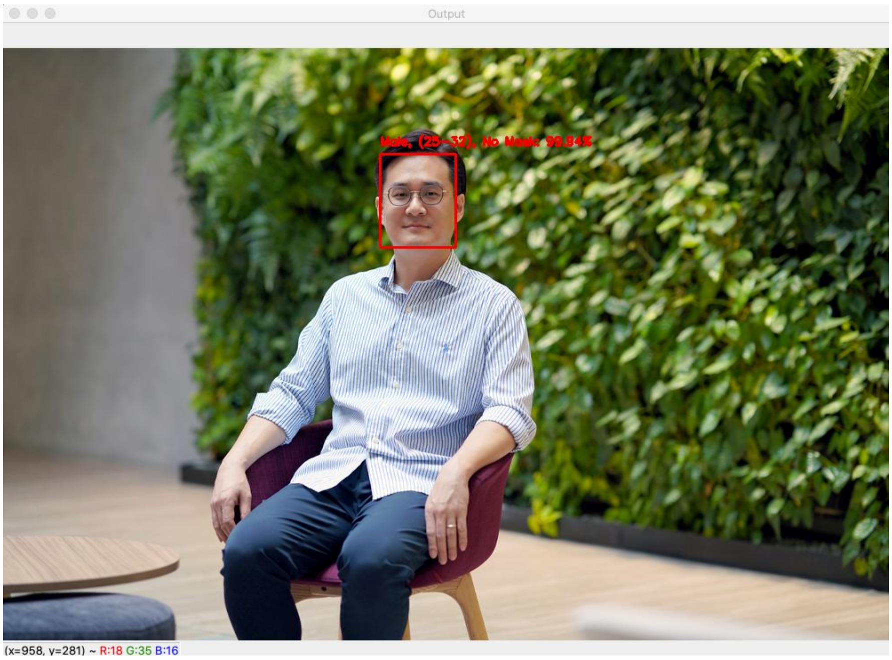

Output
Male, (25-32), No Mask: 99.84%
(x=958, y=281) ~ R:18 G:35 B:16

Figure 15: Face Mask Detector Model Sample Result 2

7 Discussion

In conclusion, we have successfully built a tool that offices can implement to ensure workplace safety.

7.1 Limitations

We acknowledge that there are limitations to our application. In practice, the face mask and gender detection generally worked well in our tests. However, age detection was unreliable.

Further, there were certain scenarios in which the detector simply did not manage to capture faces such as in low light conditions, for people who wear spectacles or headwear.

These problems surfaced during our small-scale experiments. There will most likely be similar complications that will surface in a more rigid and comprehensive testing environment. Given the different conditions and expectations that this detector will have to work with in an office, this is a big issue. How this affects the use case is a lower accuracy in identifying rule-violating employees, which defeats the purpose of what we had set out to do.

As with all Deep Learning models, data is critical. The bulk of our challenges was compiling enough images that would be useful, thus we faced a problem of insufficient training data. With our bounded resources, we were not able to capture a varied enough dataset. For instance, we should have paid special attention to features that would throw the model off, such as capturing data of individuals who wore eyewear or headgear.

7.2 Improvements

An extended use case that we think could be useful is a study of the impact of how mask wearing has on their operations. Employers will be curious to know if compliance has brought about positive good or not.

In order to get a reasonable metric that relates to the extent of compliant employees, we propose a time tracking functionality in the model that will be able to monitor the proportion of time employees are wearing a mask or not. More specifically, find the percentage that whenever a face is detected, this person is wearing a mask. This quantifiable metric will be compared against a business related metric of interest, such as their revenue or occurrences of sick leave.

There are a myriad of other optimizers that we can specify for CNN. We settled on Adam because it trains the neural network in less time and more efficiently. Mini-Batch Gradient Descent and Scholastic Gradient Descent are some other decent choices up for future considerations.

Manually tuning hyperparameters is long and also impractical. In our case, hyperparameters include learning rate, epochs and number of hidden layers, among many others. There are two generic approaches to testing search candidates. Grid search exhaustively searches all parameter combinations for given values. Random search sample a given number of candidates from a parameter space with a specified distribution.

Although grid search is useful in many machine learning algorithms, it is not efficient in tuning hyperparameters in this case as the number of parameters increases, computation grows exponentially. Random search has been found more efficient compared to grid search in hyperparameter tuning for neural networks (Bergstra and Bengio, 2012).

7.3 Learning Points

This project ensured that we had done ample research and comprehension of neural networks in order for us to implement it appropriately. Specifically, we now have extensive understanding of recognising problems in which neural networks can address. This is a valuable skill because of how much neural networks have revolutionised processes in many industries nowadays.

In the process, we have also learned how difficult it is to acquire a high quality dataset. In our case, we executed a workaround by using image augmentation and superimposing methods. Both aided us in artificially expanding the size of a training dataset by creating modified versions of images in the dataset. Importantly, it improved the performance and ability of our model to generalize.

For future projects, where we will be required to utilize such technology, we will be better prepared to because we can anticipate the processes that come with it.

7.4 Conclusion

The technology and applications discussed here are very pertinent to the current climate. We identified an area in which we could enhance social good, and proposed a practical solution by leveraging neural network technology.

With such a tool, offices, and perhaps even other operations like schools, malls and living spaces can better manage the risks that may come about from people neglecting mask wearing regulations.

8 References

Washington-Harmon, 2020, ‘Reasons Why You Should Wear A Mask, According to Experts’, *Health,* accessed 28 Nov 2020, <https://www.health.com/condition/infectious-diseases/coronavirus/why-you-should-wear-a-mask>

‘34 Enforcement Actions, Including Fines Of 300 Dollars Imposed On Two Members Of The Public, Taken For Non-compliance Of Safe Distancing Requirements At Hawker Centres’, *National Environmental Agency*, 11 April 2020, <https://www.nea.gov.sg/media/news/news/index/34-enforcement-actions-including-fines-of-300-dollars-imposed-on-two-members-of-the-public-taken-for-non-compliance-of-safe-distancing-requirements-at-hawker-centres>

Jarry, J 2020, ‘Why Some People Choose Not to Wear a Mask’, *McGill,* 3 September, <https://www.mcgill.ca/oss/article/covid-19-health/why-some-people-choose-not-wear-mask>

Xuan Zhou and Jiajun Wang. Feature Selection for Image Classification Based on a New Ranking Criterion, *Journal of Computer and Communications* 2015, 3, 74-79, 2015

S. Mark, H. Andrew, Z. Menglong, Z. Andrey and C. Liang-Chieh, *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510-4520

J. Bergstra and Y. Bengio. Random Search for Hyper-parameter Optimization, *Journal of Machine Learning Research* 13 (2012) 281-305, 2012