Frankfurt University of Applied Sciences
M.Sc. in High Integrity Systems

A Project Report on

Real-Time Mask Detection

Submitted for the Requirements of M.Sc. Program
"**Advanced Real Time Systems**" Winter Semester 22-23

Authors' Name	**Matriculation Number**
Saddam Hossain	1440339
Mohammad Sayedur Rahman	1293832
Shrabanti Saha Rimi	1377509

Under the Supervision of
Prof. Dr.- Ing. Matthias Deegener

Table of Content

Chapter **Page**

1. **Abstract** 3

2. **Introduction** 3

3. **State of the Art** 3

4. **Methodology** 4

5. **System Design** 4

6. **Requirements**

    6.1 – Hardware 5

    6.2 – Software 6

7. **System Architecture**

    7.1 – Deep Learning Architecture 7

    7.2 – CNN Layers Architecture 8

    7.3 – Mobile Application Architecture 9

8. **Software Implement** 10

9. **Outputs** 11

10. **Results** 12

11. **Discussion** 12

12. **Conclusion** 12

13. **Source Code of The Project** 13

14. **Literature** 13

1.Abstract

Real-time mask detection is a digital vision technology that implements artificial intelligence to identify whether individuals are wearing the mask properly or not. During the COVID-19 pandemic, wearing a face mask has become necessary to stopping transmission of the virus. This technology can be used in various public places like hospitals, airports, schools, and public transport systems to ensure that individuals are following the mask mandate.

Real-time mask detection uses a deep learning algorithm to evaluate photos taken by camera whether a person is using a mask or not. If people are not wearing a mask, the system can send an alert notification to the authority by providing a sound.

Keywords: mask detection, deep learning, convolution neural network, real-time, COVID-19

2.Introduction

The COVID-19 pandemic has created a huge effect on the globe and changed many things about our daily lifestyles. One of the significant changes is the mandatory use of masks in public places to prevent the spread of the virus. Yet, many people continue to disregard these measures, endangering both themselves and others. To ensure the safety of individuals, enforcing mask-wearing policies in public places is essential. Real-time mask detection systems can help enforce these policies by detecting individuals who are not wearing masks or are wearing them improperly. In this paper, we propose a deep learning-based approach for real-time mask detection.

3.State of the Art

Real-time mask detection using Convolutional Neural Networks (CNN) has become an important application in the current pandemic situation. Mask detection can be done using various computer vision techniques, but CNN has shown promising results due to its ability to learn complex features from images.

The state-of-the-art approaches for real-time mask detection using CNN are as follows:

1. YOLOv5: This is a real-time object detection algorithm that uses a single-stage architecture for detection. YOLOv5 can detect masks in real-time and is highly accurate.

2. EfficientDet-Mask: This method uses an EfficientDet object detector and a mask classification network to detect faces and classify whether they are wearing a mask. It has achieved high accuracy on benchmark datasets.

3. MobileNetV3: This is a lightweight neural network architecture that is designed to run efficiently on mobile devices. MobileNetV3 has been used for real-time mask detection, and it has shown good accuracy and speed.

4. ResNet: Residual Networks (ResNets) are deep neural networks that use residual connections to avoid the vanishing gradient problem. ResNets have been used for real-time mask detection, and they have shown good accuracy.

These state-of-the-art approaches for real-time mask detection using CNN have shown promising results and can be used for various applications such as monitoring compliance with mask mandates, detecting non-compliance, and contact tracing.

4.Methodology:

Our proposed approach for real-time mask detection uses a CNN model to classify facial images into two categories: masked, unmasked. The CNN architecture consists of multiple convolutional layers, followed by max-pooling layers, and ends with fully connected layers. The input to the model is a 260x260 RGB image of a face. The model was trained on a dataset of 10,000 images, consisting of 5,000 masked images (233MB) and 5000 unmasked images (129MB).

Dataset:https://www.kaggle.com/datasets/muhammadahsan026/facemask-dataset-covid1910k-images-2-folders?select=FaceMask-Dataset-covid-19

5.System Design

The input image for the face mask recognition system is depicted, and it contains both people wearing masks and those who are not. In the first step, the faces in the input image are identified using facial landmarks. Starting with the extraction of each face's regions of interest, we use those to apply facial landmarks later on. Subsequently, to increase generalization, we use fly mutations. Following stage 1, the stages of ROI extraction, batching, and resizing fall under intermediate processing, which occurs between stage 1 and stage 2.

The input module, the deep learning model, and the output module are the three primary parts of the real-time mask detection system. The input module is in charge of obtaining the camera's input picture or video stream. The deep learning model analyzes the input data and forecasts the presence or absence of a mask. The prediction's outcome is shown on the output screen via the output module.

Input Module: The input module collects the camera's video feed and provides it to the deep learning model for analysis. The module makes use of a video camera to record videos from various perspectives. To guarantee that the deep learning model receives high-quality input data, the module also must be able to change the video quality and resolution.

Deep Learning Model: The deep learning model oversees determining whether masks are present in the incoming data or not. A sizable collection of photos, both with and without masks, is used to train the model. Convolutional neural networks (CNNs) are used in the model to extract features from the input data and a fully connected layer is used to produce the prediction. Convolutional layers, pooling layers, and activation functions are only a few of the many layers that make up CNN. By applying convolutional operations to the image, the convolutional layers extract the features from the input data. The feature maps are downscaled by the pooling layers to lessen the computational burden. The model gains non-linearity from the activation functions, which helps it to recognize intricate patterns.

Output Module: The output module projects the deep learning model's prediction onto the output screen. The findings can be shown by the module in a variety of ways, such as a green box or a red box. The module can also give the user feedback, such as a warning sound when someone is not wearing a mask.

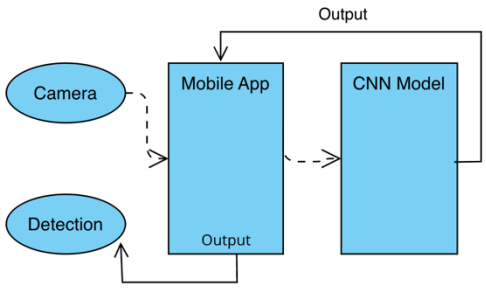

Fig:1

6.Technology

Python is the most commonly used programming language for developing neural network models and applications. In our Project, Python has been used for deep learning tasks and Swift has been used for developing our iOS Mobile Applications.

TensorFlow is required to develop and train the neural network model as A deep learning framework.

6.1.Hardware

As we have created iOS Application, It is mandatory to use MacOS as operating system. A high-resolution camera is required to capture images for mask detection. A resolution of at least 720p is recommended for accurate detection. The hardware requirements for real-time mask detection using CNN can vary depending on the specific implementation and the desired performance. A mid-range desktop computer or laptop with a dedicated GPU should be sufficient for most applications. So, we have used MacBook Pro and iPhone 14 Pro as hardware.

Fig:2

6.2.Software

Jupyter Notebook which is Integrated development environment (IDE), is used for developing and testing the application. Xcode is used to create iOS Mobile Application and It's a mendaroty to developing iOS Applications and Its only available on MacOS.

Fig:3

7.System Architecture

In this project, System Architecture has been divided into two parts, first one is Deep Learning Architecture and Second one is Mobile Application Architecture. CNN and Machine learning methods are included in the Deep Learning Architecture part. On the other hand, Real time camera implementation, Model Conversion, Prediction and notification result have been included in the Mobile Application part.

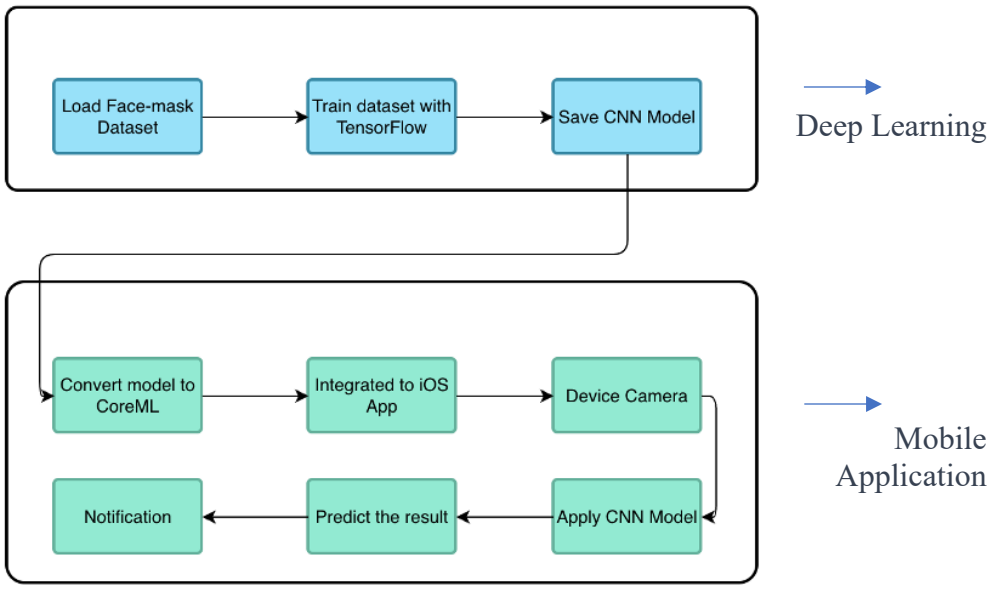

Fig:4

7.1 Deep Learning Architecture

Convolutional Neural Networks (CNNs) are a type of deep neural network architecture which is used for image classification and object detection. It consists of several layers such as:

Input Layer: The input layer of a CNN accepts the raw image data as input. The input data is usually preprocessed to reduce noise and standardize the data.

Convolutional Layer: The convolutional layer is the major building block of a CNN. This layer uses a set of filters and it knows as kernels, to perform convolution calculation on the input image. In this layer, It performs the detection of corner, edges and other features.

Activation Layer: the activation layer is a nonlinear transformation applied to the output of each convolutional layer. The purpose of the activation layer is to introduce nonlinearity into the network, which enables the CNN to learn more complex and sophisticated representations of the input data. The activation layer applies a mathematical function to the output of each convolutional layer, such as the Rectified Linear Unit (ReLU) function or the Sigmoid function.

Pooling Layer: The pooling layer downsamples the feature maps produced by the convolutional layer, reducing the dimensionality of the data and making it computationally more efficient.

Fully Connected Layer: The fully connected layer takes the output of the previous input neuron layers and performs a linear transformation on this layer. The purpose of this layer is to map the output of the convolution and pooling layers to the class scores.

Output Layer: The output layer produces the final output of the model, which can be either a class label or a probability distribution over the possible classes.

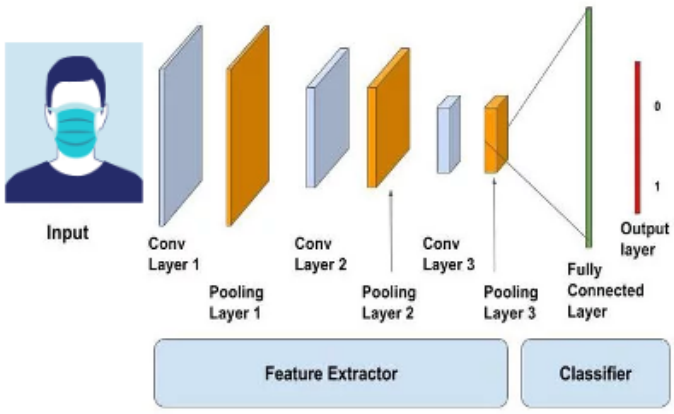

Fig:5

Fully Connected layers and Output layers are called Classifier and Rest of the layers are called Feature extractor layers.

7.2 CNN Layers Architecture:

Mask detection is a computer vision task that involves identifying whether an individual in an image or video is wearing a mask or not. Convolutional Neural Networks (CNNs) are well-suited for this task due to their ability to automatically learn and extract features from images.

CNNs can be trained to identify patterns and features in images that are associated with wearing a mask, such as the shape and texture of the mask, as well as the presence of ear loops or straps. These features can then be used to make a prediction about whether a person in an image is wearing a mask or not.

Overall, the use of CNNs for mask detection allows for efficient and accurate identification of mask-wearing behavior, which is an important step in controlling the spread of infectious diseases such as COVID-19.

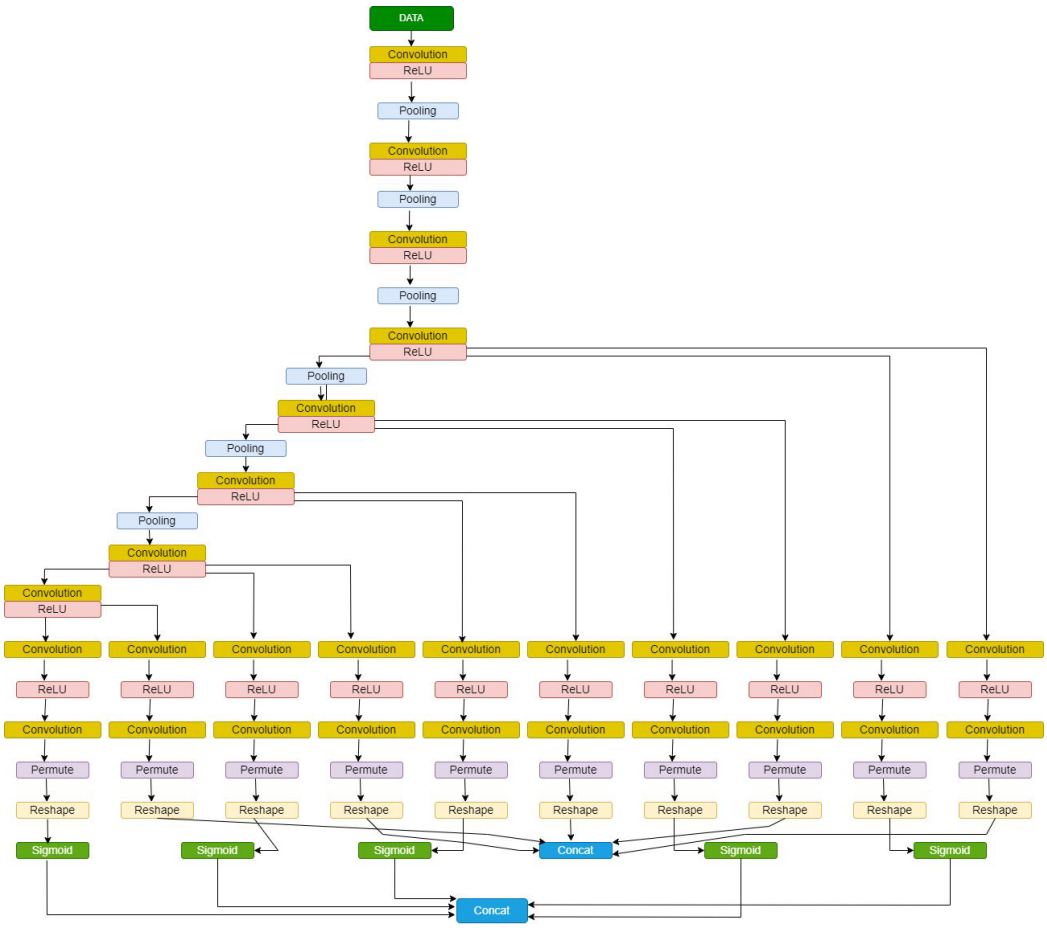

Fig:6

In the figure (Fig:6), We have designed our CNN model and There are 28 Convolution layers, 18 Rectifier linear unit (ReLU) as activation functions, 6 Pooling max layers, 5 sigmoid activation functions have been used in our approaches. Finally, we have marge the five sigmoid activation function in the output layer.

7.3 Mobile Application Architecture:

In this Mobile Application, we have used Model-View-Controller (MVC) design architecture. In this pattern, the app is divided into three main components: Model, View, and Controller. The Model represents the app's data, the View represents the UI, and the Controller acts as an intermediary between the Model and the View.

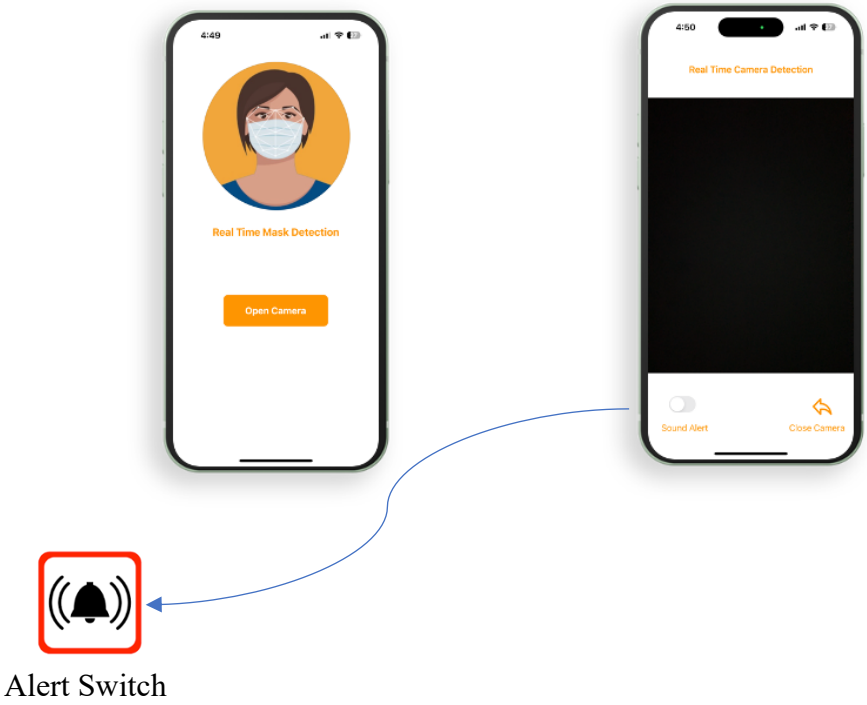

Fig:7

This mobile application is basically 2 pages application. In the home page, we can open the device camera for detection face-mask and in the camera page, it will detect the face-mask and will notify the user if it detects no-mask.

This system will also work on multiple faces on single frame. It will detect multiple face with mask and without mask surrounding with green box or red box according to our CNN model prediction.

There is a sound alert switch button in the bottom of the camera page, user can active the switch if he/she wants to generate alert sound in case of no-mask detection.

8.Software Implementation

The Mobile Application will take real time video frame using its physical camera and send to CoreML model. This model will predict the result and show the output on the mobile screen.

Model conversion is one of the most important parts. As we are developing an iOS Mobile Application, TensorFlow model is not supported in the Apple development cycle. For this reason, we have converted our TensorFlow model to Apple native Machine Learning Framework which is CoreML. Using a python script, we have converted TensorFlow model to CoreML Model.

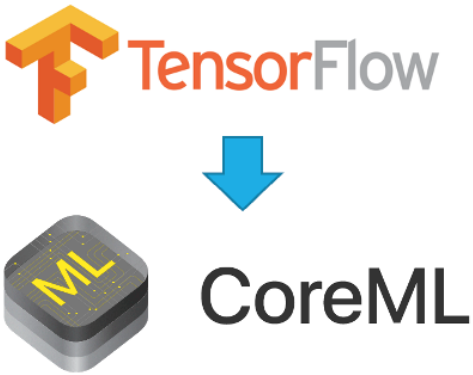

Fig:8

After model conversion, the following figure (Fig:9) shows the layer distribution for our CNN model which is generated from XCode.

Layer Distribution

Layer	Count
Convolution	28
BatchNorm	18
ActivationReLU	18
LoadConstantND	10
ReshapeStatic	10
AddBroadcastable	10
Transpose	10
PoolingMax	6
ActivationSigmoid	5
ConcatND	2

Fig:9

9.Outputs

We have tested with different colored masks and different types of masks like surgical mask, FFP2 mask. If the system can detect mask, it will show the green box surrounding the human face. If the system cannot detect the facemask properly, it will show the red box surrounding the human face. The output images are shown below.

In figure 10, we can see 2 images. On the left it is showing “mask” with a green box as the person is wearing the surgical mask properly. On the right it is showing “no mask” with a red box as the person is not wearing the mask at all.


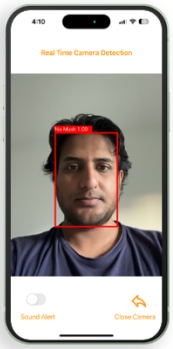

Fig:10

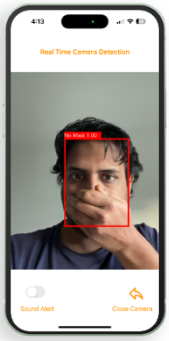
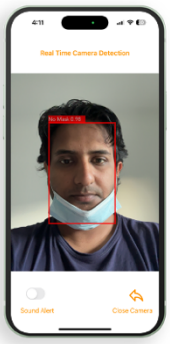

Fig:11

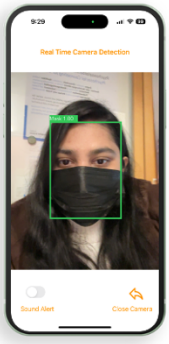
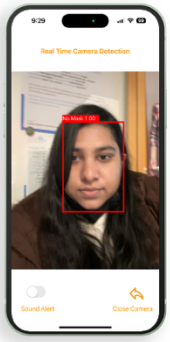

Fig:12

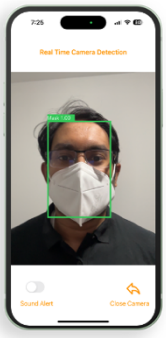
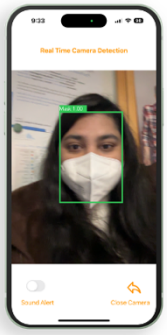

Fig:13

However, there are some challenges that we went through while building the project. Like in Fig. 14, in the left image the person is covering the face with a white paper but the system still detecting “mask” with the green box, which is unusual. On the right, the person is wearing the surgical mask improperly (Not covering the nose, which is against mask mandate) and yet it is detecting the “mask”.

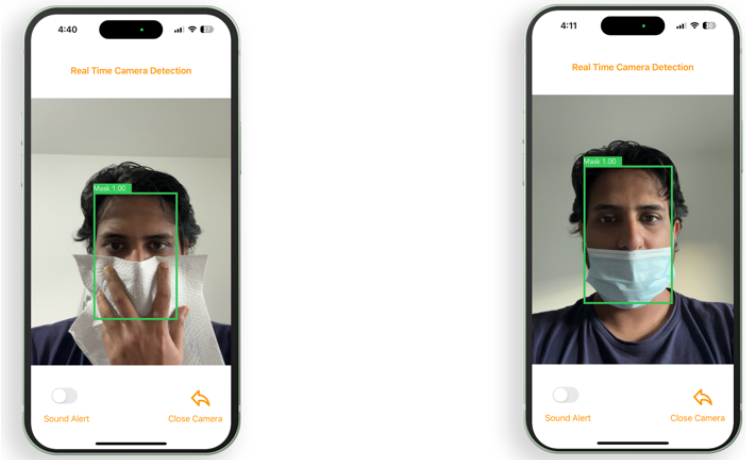

Fig:14

10.Results:

We evaluated our proposed approach on a publicly available dataset of 2,000 images, consisting of 1000 masked images and 1000 unmasked images. Our model achieved an accuracy of 91.9% for mask images and 89.6% for unmask images. The results show that our proposed approach is effective in detecting individuals who are not wearing masks or are wearing them improperly.

11.Discussion:

To achieve more accuracy in our model, we need more diversity of dataset which helps to solves our challenges as well. There are several techniques to achieve more accuracy in the CNN architectures. Data augmentation is one of them. Data augmentation helps to increase the diversity of the training set, which can improve the generalization ability of the network.

12.Conclusion:

The COVID-19 pandemic has led to a significant change in our daily lives, including the mandatory use of masks in public places. However, enforcing these guidelines can be challenging, as some individuals do not adhere to them. Real-time mask detection systems can help enforce these policies and ensure the safety of individuals. In this paper, we proposed a deep learning-based approach for real-time mask detection, which uses a CNN architecture to classify facial images into two categories: masked and unmasked. Our proposed methodology results achieve an accuracy of 91.9%. The proposed approach shows promising results and can be implemented in various settings to help enforce mask-wearing policies in real-time. Future work can include the deployment of the proposed approach in real-world scenarios and the exploration of other deep learning architectures to improve the accuracy of the model.

13.Source Code of The Project:

GitHub: - https://github.com/Sabuj-CSE11/RealTimeMaskDetection

14.Literature:

1. S. Singh, U. Ahuja, M. Kumar, K. Kumar and M. Sachdeva, "Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment", *Multimed. Tools Appl.*, vol. 80, no. 13, pp. 19753-19768, 2021.

2. Ambika Lakhera, Priyansh Jain, Ruchi Gajjar, Manish Patel, "Face Mask Detection for Preventing the Spread of Covid-19 using Knowledge Distillation", *2021 International Conference on Computational Performance Evaluation (ComPE)*, pp.815-819, 2021.

3. Qiang Wang, Lan Hou, Jon-Chao Hong, Xiantong Yang, Mengmeng Zhang, "Impact of Face-Recognition-Based Access Control System on College Students' Sense of School Identity and Belonging During COVID-19 Pandemic", *Frontiers in Psychology*, vol.13, 2022.

4. Goyal, H., Sidana, K., Singh, C. *et al.* A real time face mask detection system using convolutional neural network. *Multimed Tools Appl* **81**, 14999–15015 (2022).

5. Kaur G., Sinha R., Tiwari P., Yadav S., Pandey P., Raj R., Vashisth A., Rakhra M. Face mask recognition system using CNN model. Neurosci. Inform. 2022;2:100035. doi: 10.1016/j.neuri.2021.100035.

6. https://github.com/AIZOOTech/FaceMaskDetection
7. https://github.com/tf-coreml/tf-coreml#new-beta-tfcoreml-converter-with-core-ml-3
8. https://www.kaggle.com/datasets/muhammadahsan026/facemask-dataset-covid1910k-images-2-folders?select=FaceMask-Dataset-covid-19
9. https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939
10. https://learnopencv.com/generative-and-discriminative-models/
11. https://developer.apple.com/documentation/coreml