Unlocking Doors: A TinyML-Based Approach for Real-Time Face Mask Detection in Door Lock Systems

Azzedine El mrabet[1], Ayoub Tber[2], Mohamed Benaly[3], Laamari Hlou[4], Rachid El gouri[5]
[1,2.5]Laboratory of Advanced Systems Engineering, National School of Applied Sciences, Ibn Tofail University, Kenitra, Morocco
[3,4]Laboratory of Electronic Systems, Information Processing, Mechanics and Energetics, Faculty of Sciences, Ibn Tofaïl University, Kenitra, Morocco

Article Info

Article history:

Received Apr 30, 2023
Revised Jun 11, 2023
Accepted Jun 24, 2023

Keywords:

Face Mask Detection
TinyMl
YOLOv5
Door Lock Systems
Raspberry pi

ABSTRACT

In response to the rapid spread of coronaviruses, including COVID-19 and seasonal common cold viruses, this article introduces a proposed system for enhancing door lock systems using TinyML technology for real-time face mask detection. The research project focuses on developing a machine learning model based on the YOLOv5 architecture to classify individuals based on their mask-wearing behavior correctly, incorrectly, or not at all in high-risk spaces prone to the transmission of coronaviruses, such as healthcare facilities, laboratories, and public settings. The study outlines the hardware and software tools utilized, including the Raspberry Pi 4, camera hardware, and the YOLOv5 machine learning model. The model is trained using a dataset containing three different classes and converted to a TFLite format for efficient implementation on the Raspberry Pi. Evaluation results demonstrate a mean Average Precision (mAP) of 0.99 and an inference rate of 10FPS for a 128-frame size input. This proposed system offers practical implications for enhancing door lock systems and promoting public health and safety during outbreaks of coronaviruses, including COVID-19 and other seasonal coronaviruses, providing a valuable approach to decrease the spread of these diseases and mitigate transmission risks in high-risk spaces, thereby contributing to the overall reduction of public health threats.

Corresponding Author:

Azzedine EL MRABET,
Laboratory of Advanced Systems Engineering, National School of Applied Sciences, Ibn Tofail University
BP 242 Av. From the University, Kenitra 14000, Morocco
Email: azzedine.elmrabet@uit.ac.ma

1. INTRODUCTION

Diseases caused by coronaviruses, such as COVID-19 and the seasonal common cold, have a rapid spread and pose a significant health threat. These diseases particularly affect high-risk spaces, including healthcare facilities, laboratories, and other public settings susceptible to the transmission of coronaviruses. Research [1] has demonstrated that coronaviruses, with COVID-19 being a prominent example, primarily spread through droplet and aerosol transmission during social interactions with infected individuals. To mitigate the spread of coronaviruses in these high-risk spaces, wearing protective masks has been recommended as a practical and scientific strategy by global health organizations. However, ensuring the effectiveness of mask-wearing measures necessitates an automated detection approach to monitor and enforce proper mask usage. This approach leverages computer vision techniques to identify individuals who are wearing masks correctly, thereby granting access only to those individuals in these spaces. By implementing this approach, intelligent epidemic prevention and control efforts can be enhanced. Furthermore, in addition to mask usage, individuals are encouraged to practice social distancing and undergo regular temperature screenings in public places[2], These additional measures have proven to be convenient and effective in reducing the transmission of coronaviruses.

To address these challenges, our proposed approach is based on computer vision technology. It utilizes advanced machine learning algorithms and real-time vedio analysis to detect and classify individuals based on their mask-wearing behavior. By accurately identifying those who are wearing masks correctly, our approach aims to provide access only to individuals who comply with the recommended safety measures in high-risk spaces. This intelligent access control system can play a crucial role in minimizing the transmission of coronaviruses and maintaining the safety of the environment.

The field of computer vision has seen widespread use in recognizing facial expressions[3] and disease diagnosis[4] due to the rapid development of deep learning in recent years. Classical object detection methods like YOLOv4[5], YOLOv3[6], Faster RCNN[7], CornerNet[8], RetinaNet[9], and FCOS[10] have also emerged. There are two main types of face and mask detection algorithms: real-time detection methods that prioritize speed[11], [12] and high-performance detection methods that prioritize accuracy. Several research studies have investigated the face mask detection, proposing models such as BlazeFace inspired by MobileNet and SSD[11], DRUID[13], and a model used for detecting faces, which combines the SqueezeExcitation Network and ResNet[14]. Mask detection has also been addressed using techniques such as YOLOv3, Faster RCNN[15], and transfer learning methods based on ResNet50 and YOLOv2[12].

The proposed system for face mask detection in door lock systems is based on the integration of TinyML with image processing and machine learning techniques. In recent years, TinyML has emerged as a promising technology for implementing machine learning algorithms on resource-constrained devices such as microcontrollers and embedded systems. TinyML-based systems offer the potential for low-cost, low-power, and high-performance solutions for real-time applications such as mask detection.

To provide a comprehensive understanding of the existing research landscape, we have conducted a review of related work on face mask detection and recognition using deep learning, object detection platforms, and the YOLO algorithm. The findings from these articles are relevant to our study and provide valuable insights into the advancements in face mask detection techniques.

Several approaches have been proposed in the literature to solve the crucial task of face mask detection in the pandemic. One such approach is Masked Face Recognition (MFR), which uses deep learning techniques to identify masked faces. A comprehensive review of recent works on MFR, including benchmarking datasets, evaluation metrics, challenges, and promising research directions, is provided in [16]. Another real-time face mask detection system that can detect multiple face masks in a single scene using You Only Look Once version 3 (YOLOv3) and TensorFlow lite platforms is presented in [17]. The proposed system is light, small, cheap, and has an accuracy rate of 99% during training and testing. In [18], a face mask-wearing condition identification method using image super-resolution with classification network (SRCNet) achieved 98.70% accuracy in identifying face mask-wearing conditions from unconstrained 2D facial images, outperforming traditional end-to-end image classification methods. A smart door lock system using face recognition and speech recognition techniques for secure access control is presented in [19]. Other related works include [20]–[22], which propose deep learning-based face mask detection methods using pre-trained CNN models, Raspberry Pi and YOLOv4 object detection algorithm, and real-time CNN models, respectively. However, none of these works focused on integrating face mask detection with door lock systems using TinyML technology.

In this article, we present a TinyML-based approach for a face mask detection in the door lock systems. We use a Raspberry Pi 4 as our hardware platform and employ a deep learning-based convolutional neural network (CNN) model, specifically YOLOv5, for mask detection. We describe the data collection and preprocessing procedures, including the use of an open-source face images dataset for training the YOLOv5 model. We evaluate the performance of the mask detection model using quantitative metrics and demonstrate its effectiveness in a door lock system.

The rest of the article is organized as follows: Section 2 describes the technical details of our implementation, including the hardware and software components used and the specific machine learning models and algorithms employed. Section 3 presents the results of our experiments in Section 4. We conclude the article with a summary of our key findings and future directions for research in this area.

2. RESEARCH METHOD

In this paper, we present a TinyML-based approach for real-time face mask detection in door lock systems. Our aim is to enhance the security and efficiency of door lock systems by leveraging trained Convolutional Neural Network (CNN) models, specifically YOLOv5, to detect the presence of face masks in real-time video streams. The utilization of TinyML technology allows us to deploy these CNN models on resource-constrained devices, such as the Raspberry Pi, facilitating real-time processing and analysis at the edge.

Our research focuses on implementing the face mask detection system, which we will thoroughly explain in the subsequent sections. The approach involves integrating the trained YOLOv5 model with the Raspberry Pi 4 and camera hardware, enabling seamless and efficient face mask detection.

By detecting and analyzing the presence of face masks in real time, our approach offers a practical and intelligent solution for enforcing mask-wearing protocols in high-risk spaces. This is particularly relevant in the context of infectious diseases, including coronaviruses such as COVID-19 and the seasonal common cold, where the use of face masks plays a crucial role in preventing the spread of the disease.

Through our research, we aim to contribute to the implementation of more efficient and secure door lock systems that promote public health and safety. By effectively detecting and monitoring mask usage, our approach has the potential to mitigate the transmission of infectious diseases and provide a reliable means of controlling access in high-risk environments. In the following sections, we will delve into the details of our proposed platforms, the training process of the CNN models, and the evaluation of the system's performance.

2.1. TinyML Approach for Face Mask Detection using Raspberry Pi

The TinyML approach, combined with the power of the Raspberry Pi, provides a highly effective solution for real-time face mask detection. TinyML technology allows machine learning algorithms to be deployed on resource-constrained devices like the Raspberry Pi, enabling efficient and low-latency processing directly at the edge. This approach offers significant advantages over cloud-based solutions, such as reduced reliance on internet connectivity and enhanced privacy.

To implement the TinyML approach, we utilize TensorFlow Lite, a lightweight version of TensorFlow specifically designed for mobile and embedded devices. The trained face mask detection model is converted to a format compatible with the Raspberry Pi using the TensorFlow Lite converter's Python API. This conversion process optimizes the model for deployment on the device, ensuring efficient utilization of computational resources and minimizing memory requirements.

The Raspberry Pi serves as an ideal platform for face mask detection, equipped with camera hardware for capturing real-time video streams. The computational power of the Raspberry Pi allows for swift inference of the face mask detection algorithm, enabling instantaneous analysis and decision-making. This real-time processing capability is crucial in environments where prompt detection of face masks is essential, such as public spaces, hospitals, and transportation systems.

By deploying the face mask detection system on the Raspberry Pi, we eliminate the need for constant cloud connectivity, ensuring the system's reliability and availability even in areas with limited internet access. Additionally, the low power consumption of the Raspberry Pi enables continuous operation without significant energy consumption.

The versatility of the Raspberry Pi and its integration with the TinyML approach make it adaptable to various environments and scenarios. The system can be easily deployed in airports, shopping centers, public transportation systems, and other high-traffic areas where mask compliance is mandatory. Moreover, the compact size and portability of the Raspberry Pi allow for flexible installation and usage across different locations.

The TinyML approach, combined with the Raspberry Pi, provides an efficient and practical solution for real-time face mask detection. By leveraging the capabilities of TensorFlow Lite and the computational power of the Raspberry Pi, we can achieve accurate and instantaneous detection of face masks, ensuring compliance with health protocols in diverse environments.

2.2. Tensorflow Lite

The proposed model is deployed on a device using TensorFlow Lite, which is a set of tools specifically designed for deploying TensorFlow models on mobile and embedded hardware devices. TensorFlow Lite serves as a lightweight version of TensorFlow, enabling the deployment of deep learning models on mobile phones or microcontroller development boards[17]. It comprises two main components: the TensorFlow Lite converter and the TensorFlow Lite interpreter.

The TensorFlow Lite converter converts TensorFlow models into a specialized, compact format suitable for embedded devices with limited memory. On the other hand, the TensorFlow Lite interpreter runs the converted model efficiently using high-performance operations compatible with the target device. In this paper, the proposed model is obtained using the Python API of the TensorFlow Lite converter.

However, it's important to note that optimizing the model using the TensorFlow Lite converter for real-time performance may result in a slight reduction in accuracy.

2.3. YOLOV5

YOLO is an advanced real-time object detector that has undergone several iterations from YOLOv1 to YOLOv4. The latest version, YOLOv5, has built upon this foundation and has demonstrated exceptional performance on widely used object detection datasets, such as Pascal VOC (visual object classes)[23] and Microsoft COCO (common objects in context)[24].

The YOLO algorithm, initially proposed by Redmon J.[25]. introduced the concept of combining bounding box and classification tasks into a regression problem, eliminating the need for candidate box extraction in the two-stage approach. Here's how the YOLO algorithm operates: The input image is divided into S x S grids, and each grid predicts the location of the target or the actual box that encompasses the target within the grid. A total of S x S x B bounding boxes are generated, each composed of the target's center point coordinates, width, height (x, y, w, h), and confidence in the box enclosing the target. The S x S grids estimate the probabilities of different target categories within each grid. The final predictions are obtained by filtering the bounding boxes using non-maximum suppression.

The YOLO algorithm has evolved rapidly in recent years, with the latest versions being YOLOv5 and YOLOv8. For face mask detection, the YOLOv5 algorithm has demonstrated high accuracy and fast processing times. To further enhance the detection method's precision, this study selected the s (small) version of YOLOv5 as the benchmark network model and made additional improvements.

YOLOv5 is considered the most sophisticated detection network available for the YOLO object detection algorithm. It introduced several innovations based on the YOLOv3 and v4 algorithms to improve detection speed. Notably, YOLOv5 replaced manually chosen anchor boxes with automatically determined anchor boxes, which accelerated the R-CNN algorithm. Additionally, K-means clustering was performed on the bounding box dimensions to obtain better prior values. YOLOv5 was released by Glenn Jocher in 2020 [26] and consists of an input layer, a backbone, a neck, and a prediction layer, as depicted in Figure 1[27].

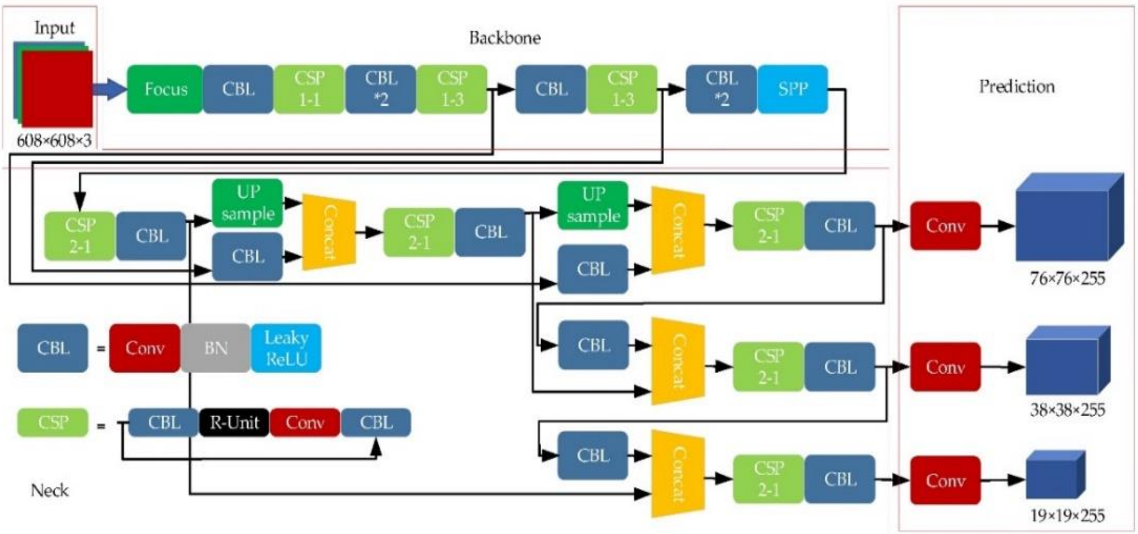

Figure 1. YOLOv5 network architecture

Detecting masks in real-world scenarios presents a significant challenge, as it involves identifying various types of masks, including those that are worn improperly or not worn at all. These masks can exhibit variations in shape, texture, and color, making it challenging for traditional machine learning models to extract meaningful features. However, our research has shown that YOLOv5 excels in learning long-range features, making it an ideal option for effectively detecting masks in diverse environments.

2.4. Dataset Description

Researchers often create open-source datasets to fulfill their specific research needs. In light of the global COVID-19 pandemic and the mandatory requirement of wearing face masks in public settings, there has been a significant focus on applying deep learning techniques to detect faces with masks, faces without masks, and faces with improperly worn masks.

For this task, the primary dataset utilized is the face mask detection dataset sourced from Kaggle[28]. This dataset comprises 853 RGB images, featuring individuals wearing face masks correctly, individuals wearing masks improperly, and individuals without masks. A selection of sample images from the dataset is showcased in Figure 2.

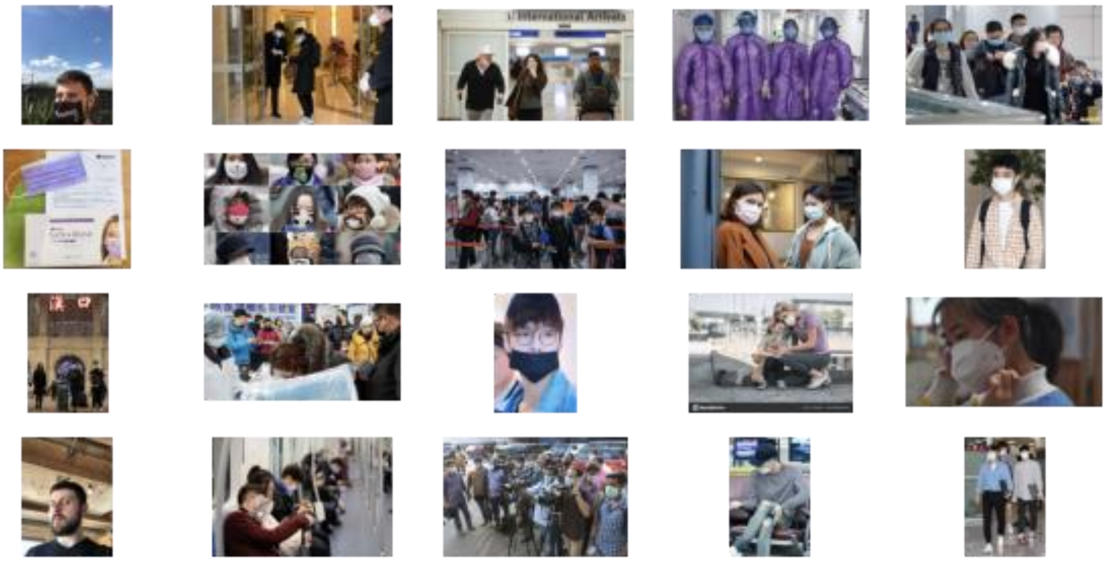

Figure 2. Sample images from the dataset

The datasets used in this study have undergone preprocessing to enhance their quality and diversity. Data augmentation techniques have been employed to increase the variation and quantity of images. Operations such as cropping, flipping, rotation, and alignment have been applied to the images. These augmentation processes aid in improving the representation of the images. Furthermore, techniques such as rescaling, segmentation, noise removal, and smoothing have been utilized to enhance the overall quality of the images.

Additionally, image adjustment techniques, such as improving image sharpness using the variance of Laplacian, have been applied. These preprocessing and data augmentation techniques have significantly improved the quality and variety of images in the face mask detection dataset. Consequently, the dataset is now more suitable for training deep learning models to accurately detect faces with masks, faces without masks, and faces with improperly worn masks. Sample images from the dataset after augmentation are presented in Figure 3.

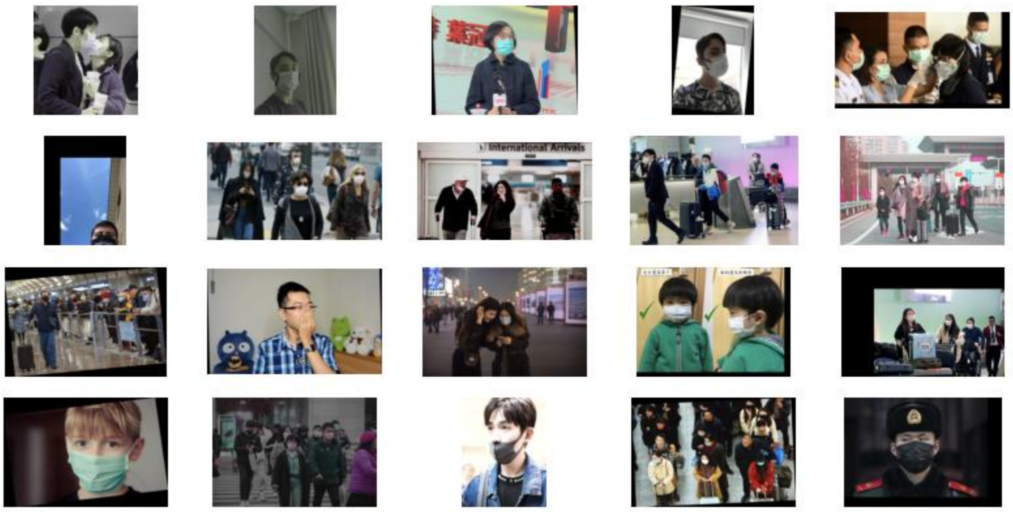

Figure 3. Sample images from the dataset after augmentation

The dataset used in this study consisted of a total of 17,860 images, as illustrated in Figure 4. The dataset was divided into three subsets: training, validation, and test sets. Approximately 80% of the data, which corresponds to 14,288 images, was allocated for training purposes. The validation set contained 10% of the data, equivalent to 1,786 images, while the remaining 10% of the data, also comprising 1,786 images, was reserved for testing and evaluating the algorithms.

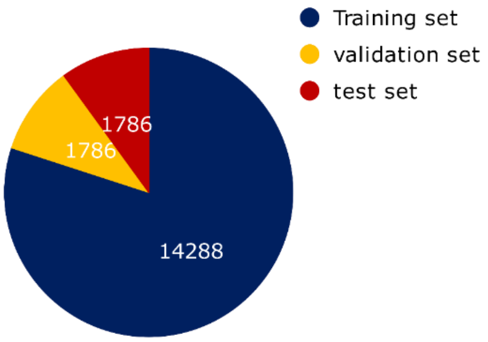

Figure 4. Distribution of the dataset into training, validation, and test sets

2.5. Hardware Components

2.5.1. Raspberry PI

The Raspberry Pi has gained popularity as an efficient and cost-effective solution for implementing AI models on a small scale, offering low power consumption high speed [17]. The latest iteration, the Raspberry Pi 4 Model B, is equipped with the Broadcom BCM2711, a Quad-core Cortex-A72 (ARM v8) 64-bit System on a Chip (SoC) running at a clock speed of 1.5 GHz. It is available in four variants with varying RAM sizes: 1 GB, 2 GB, 4 GB, and 8 GB. Additionally, it features general-purpose input-output (GPIO) pins, a camera serial interface (CSI) port, and two micro-HDMI terminals. These capabilities make the Raspberry Pi 4 an ideal platform for developing various detectors with different AI workloads.

The Raspberry Pi 4 operates at 5 V, ensuring energy efficiency in embedded devices. It utilizes a Secure Digital (SD) card, typically with a capacity of 64 GB, for storing the operating system and managing the reading and writing of large amounts of data. Figure 5 illustrates the connections of the Raspberry pi [29].

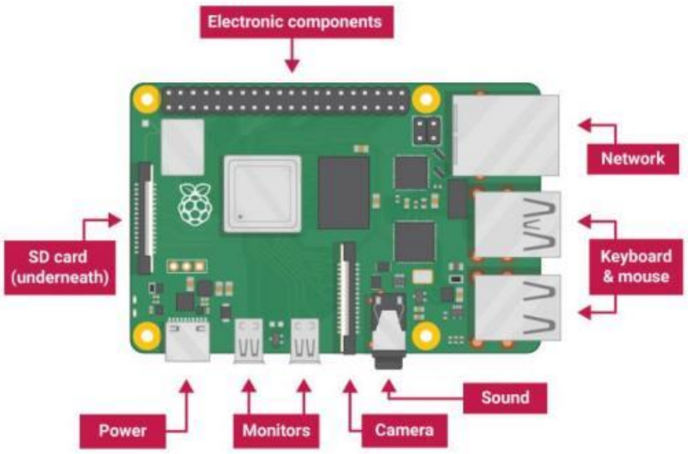

Figure 5. Shows Raspberry pi connections

2.5.2. Raspberry Pi Camera

For this project, the Raspberry Pi Camera module with an 8-megapixel sensor is employed. This module is capable of recording videos in various resolutions, including 1080p30, 720p60, and 640x480p90, and it offers a maximum resolution of 3270x2444 pixels. The camera module consists of a fixed lens and a Sony IMX219 image sensor, which is designed as an add-on board specifically for the Raspberry Pi. It is connected to the Raspberry Pi through one of its small ports located on the top of the board. The camera module utilizes the Camera Serial Interface (CSI) interface, which is purpose-built for connecting cameras to the Raspberry Pi.

2.5.3. Servo Motor

A servo motor is an electrical device utilized for controlling speed, torque, and position. It consists of an encoder that converts mechanical movement into digital signals, which are then interpreted by a motion controller [33]. The servo motor operates within a closed-loop system, where it utilizes position feedback to regulate its motion at various angles. In this project, the servo motor is adopted to simulate the sliding motion of the door. By accurately controlling the position of the servo motor, we can replicate the movement of a sliding door within the defined angle range of 0 to 180 degrees.

2.5.4. 7-inch Display

To provide real-time feedback on face mask detection and enable user interaction, a 7-inch screen can be employed. These screens typically offer high resolution and incorporate a capacitive touchscreen for easy user engagement. Connecting the screen to the Raspberry Pi is achieved through an HDMI cable, while power can be supplied either by a separate power source or a USB cable.

The screen serves multiple purposes, including displaying the live video feed captured by the Raspberry Pi camera and presenting the results of the face mask detection algorithm. When an individual is detected wearing a mask correctly, the screen can display a message indicating their adherence to mask-wearing guidelines. Conversely, if someone is not wearing a mask or is wearing it improperly, the screen can exhibit a warning message reminding them to wear a mask.

2.6. System Overview

The integrated real-time face mask detection system, combined with a door lock, represents a state-of-the-art hardware solution that harnesses the capabilities of TinyML technology. The system operates by analyzing a video stream captured by a webcam that is attached to the door. Its primary objective is to ensure strict compliance with health protocols, especially during the occurrence of seasonal common colds, by effectively detecting face masks in real-time. This feature is particularly important in high-risk spaces where multiple individuals must wear masks to gain access.

Figure 6 provides a block diagram that outlines the real-time face mask detection system integrated with the door lock. The diagram offers a visual representation of the system's hardware components and their interconnections.

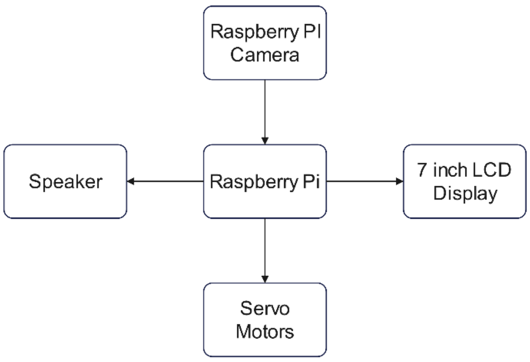

Figure 6. Block diagram of the system

The system is equipped with the YOLOv5 algorithm, enabling it to detect the presence of masks on all faces of users standing in front of the door's webcam. Once a group of users approaches the door, the system instantly activates and performs real-time face mask detection. If all individuals wear masks correctly, the system swiftly sends a command to unlock the door, granting seamless access to the group.

However, in the event that any individual is not wearing a mask or wearing it improperly, the system decisively denies access to all persons until everyone complies with the mask-wearing rules. The system provides informative voice prompts through a high-quality speaker and can also display instructions on an interactive screen to guide users on how to wear their masks properly.

Moreover, the voice prompts can be fully customized to provide crystal-clear instructions to the user. For instance, when all individuals wear masks correctly, the system can play a

voice prompt saying "Door opening" to signal that the group has been granted access. Conversely, suppose the system detects that any individual is not wearing a mask or wearing it improperly. In that case, the system can play a voice prompt saying "Door still closed, please wear your mask correctly" to prompt the users to wear their masks properly.

Figure 7 shows the system flowchart of the real-time face mask detection system integrated with a door lock. This flowchart outlines the step-by-step process that the system follows to detect face masks on users and make decisions on granting access.

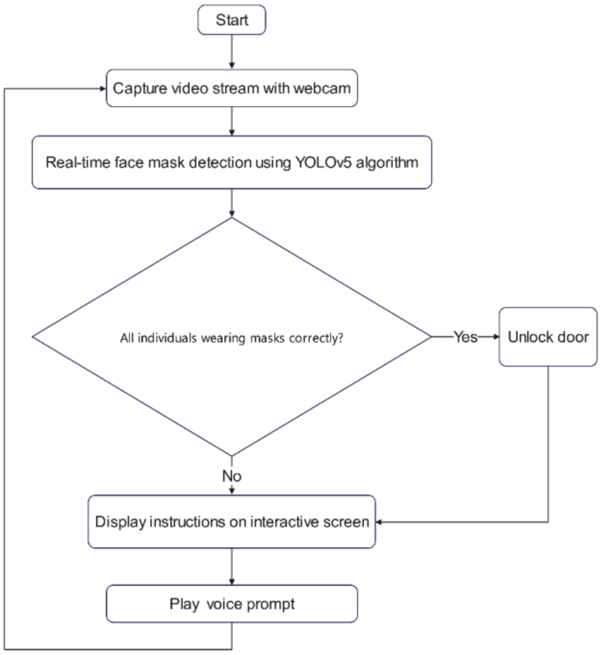

Figure 7. System flow chart

This system is ideal for deployment in various public places such as airports, shopping centers, and public transportation systems, where the use of masks is non-negotiable, and precisely in hospitals and sensitive areas where strict adherence to health protocols is critical. With its exceptional real-time detection capability, this system plays a pivotal role in curbing the spread of the virus and guarantees a safer environment for everyone, especially in densely populated places where compliance with health protocols is of utmost importance. The system's ability to ensure that everyone is wearing masks correctly is particularly valuable in hospitals and sensitive areas, where vulnerable patients and healthcare workers must be protected from potential infection.

3. RESULTS AND DISCUSSION

The 'face mask detection' dataset, consisting of images categorized into three classes: 'wearing a mask,' 'not wearing a mask,' and 'wearing a mask incorrectly,' was used to train the model in Google Colab, a free Integrated Development Environment (IDE). Google Colab provided access to Tesla K80 GPUs, enabling efficient training and inference processes for our deep learning model. Additionally, the model was deployed on a Raspberry Pi, specifically the Raspberry Pi 4 Model B 4GB ram version, due to its low-cost, low-power, and high-performance capabilities. The Raspberry Pi was equipped with a camera module to capture real-time video streams for face mask detection. These carefully chosen hardware components, combined with the TinyML approach, allowed us to achieve accurate and real-time detection of the three classes: individuals wearing a mask, individuals not wearing a mask, and individuals wearing a mask incorrectly. The utilization of the 'face mask detection' dataset, along with the properties of the Raspberry Pi, contributed significantly to the overall success of our system.

3.1. Model Evaluation Indicators

Various evaluation metrics such as precision, recall, average precision (AP), and mean average precision (mAP) were used to evaluate the model. The precision measure can effectively and fairly assess the algorithm's ability to locate and detect its intended targets. The network's algorithmic performance for the task of classifying images with multiple defects was evaluated using mAP, which was deemed an appropriate measure for this purpose. The following (1) to (4) [17]give mathematical for evaluation and tracking metrics of the suggested model:

$$Precision = \frac{True\ Positives}{True\ Positives\ +\ False\ Positives} \tag{1}$$

$$Recall = \frac{True\ Positives}{True\ Positives\ +\ False\ Negatives} \quad (2)$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (3)$$

$$mAP = \frac{1}{Q} \times \sum_{i=1}^{Q} AP_i \quad (4)$$

The terms FN, TP, FP, and TN are used to represent false negative, true positive, false positive, and true negative respectively. TP values indicate images that have been correctly identified as positive by the model. Conversely, TN images are those that have been identified as positive but were incorrect due to an error in prediction. FP images are those that have been incorrectly identified as positive by the model. FN images are those that were incorrectly identified as negative even though, they were positive. Precision is a measure of the number of true positive predictions made by the model. Recall measures the ability of the classifier to identify all positive cases. The F1-score is a measure of the model's test accuracy. These performance metrics are considered highly precise measurements. The model is tested in stages to ensure accurate detections.

3.2. Training the YOLOv5

During this phase, objects in the preprocessing frames are given learnable weights and biases, after which the YOLOv5 algorithm is employed using its default parameters. You can refer to Table 1 for an illustration.

Table 1. Hyperparameters for YOLOv5

Hyperparameter	Value
weidths	YOLOv5s
depth_multiple	0.33
width_multiple	0.5
initial learning rate	0.01
final learning rate	0.01
SGD momentum	0.937
optimizer weight decay	0.0005
warmup initial momentum	0.8
initial bias learning rate	0.1
warmup epochs	3
total training iterations	100

3.3. Summary of Results

The object detection model has been trained to execute quickly and can be used in real-time, providing predictions within a fraction of a second. Table 2 displays the evaluation metrics for a dataset when using YOLOv5s with an image size of 640 and trained for 100 epochs.

Table 2. Performance metrics of the model

Metric	Value
Precision	0.99407
Recall	0.99263
mAP@0.5	0.99405
mAP@0.5:0.95	0.89273
F1 score	0.99335

To assess the neural network's sensitivity, we employed mAP as an effective metric, as illustrated in Figure 8. Moreover, Figure 9 showcases the Precision-Recall Curve (PR-curve) for the network size, evaluating the performance at various GIoU thresholds ranging from 0.5 to 0.95. Our evaluation results indicate that the model performed well overall, with a high mAP value of 0.99% achieved for the network size of 224 and a GIoU threshold of 0.5.

Additionally, to gain insights into the model's classification performance, we analyzed the confusion matrix, as shown in Figure 10. The confusion matrix allows us to examine the number of true positives, true negatives, false positives, and false negatives predictions made by the model. This analysis provides valuable information about the model's accuracy, precision, recall, and overall performance.

The combination of the evaluation metrics, including mAP, the Precision-Recall Curve, and the confusion matrix, provides a comprehensive understanding of the model's performance and its ability to accurately detect face masks.

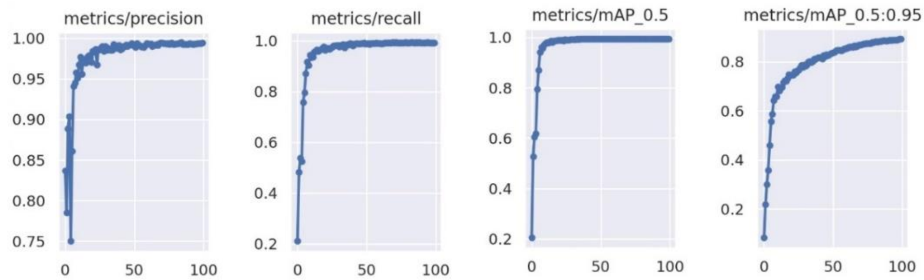

Figure 8. mAP of our model

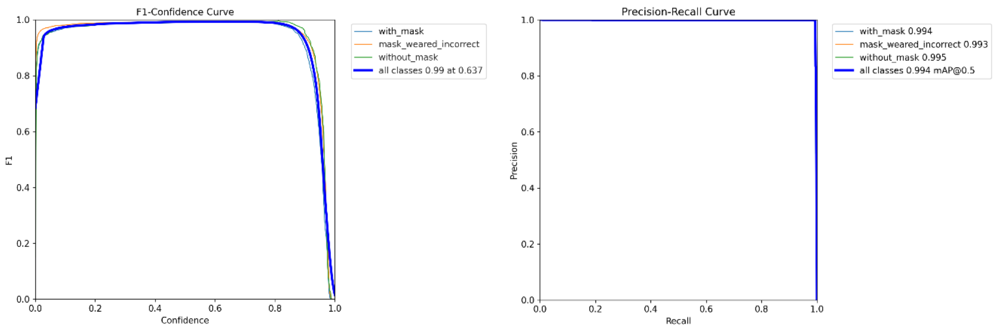

Figure 9. F1 score PR-curve of our model, with the precision of 0.994 and recall of 0.993. The curve demonstrates the trade-off between precision and recall at different decision thresholds

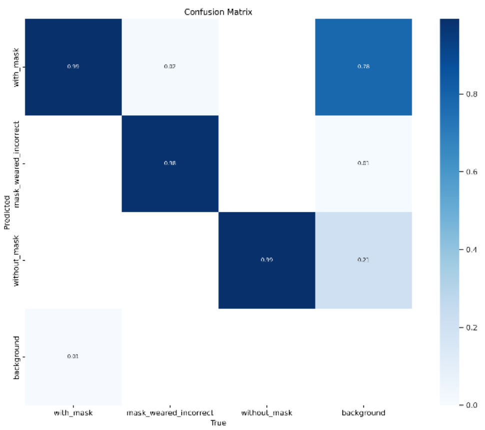

Figure 10. Confusion matrix for face mask detection.

Comparing our results with the findings of a related study[30], which proposed a face mask detection algorithm based on YOLOv5, we observed that their mean Average Precision (mAP) reached 0.709. In our experiments, after training the model for 100 epochs, we achieved a significantly higher mAP value of 0.99. It is worth noting that the differences in mAP values may arise due to variations in dataset composition and training settings. In our study, we employed augmentation techniques to enhance the diversity and robustness of our dataset, which could have contributed to the improved performance of our model. Furthermore, the specific hyperparameters used in training can also impact the model's accuracy and mAP values. Nonetheless, our results demonstrate the effectiveness of our approach in achieving a high level of accuracy in face mask detection tasks.

3.4. Inference

The inference time of YOLOv5 on Raspberry Pi is influenced by various factors, including the number of model weights, the input image size, and the number of objects in the image. Larger models and higher input resolutions can significantly increase the inference time, while reducing the input resolution can improve the speed at the expense of accuracy.

Notwithstanding these limitations, experimental results have demonstrated that YOLOv5 (tflite version) inference on a Raspberry Pi is practical and can achieve reasonable performance. For instance, using a Raspberry Pi 4 with 4 GB of RAM, YOLOv5 (tflite version) achieved an average inference speed of 2-4 FPS for an input frame size of 256, and 8-10 FPS for a size of 128. Additionally, when using YOLOv5 without converting it to the tflite version, the average inference speed for an input frame size of 128 was 2-4 FPS, with a lower resource utilization of approximately 100% of the processor.

Compared to related work [31], where the author also performed YOLOv5 inference on a Raspberry Pi without converting to TensorFlow Lite, a similar inference speed of approximately 3 FPS was achieved. These results highlight the practicality of YOLOv5 for real-time inference on resource-constrained devices like Raspberry Pi.

When considering the benefits and drawbacks of using TinyML for face mask detection on Raspberry Pi, several factors come into play. One of the key advantages is the ability to deploy lightweight deep learning models optimized for edge devices, allowing for real-time inference and reduced reliance on cloud-based processing. Additionally, TinyML frameworks such as TensorFlow Lite provide tools and optimizations specifically designed for efficient execution on embedded systems.

However, it is important to note that the use of TinyML may come with certain trade-offs. Model size reduction techniques and lower precision quantization can lead to a slight decrease in accuracy compared to their full-size counterparts. Additionally, the limited computational resources of devices like Raspberry Pi may impose restrictions on model complexity and input resolution, affecting inference speed and performance.

What sets the usage of TinyML, such as TensorFlow Lite, apart from traditional approaches is its focus on efficient inference on edge devices, enabling real-time processing without relying on cloud connectivity. By leveraging the power of TinyML, face mask detection can be performed directly on the Raspberry Pi, ensuring privacy, low latency, and reduced network dependencies. This offers a significant advantage in scenarios where real-time response and edge computing capabilities are critical, such as in crowded public spaces or remote locations.

3.5. Implementation

Once the training and testing procedures are complete, the model is validated and implemented on the intended embedded device. The Figure 11 depicts the complete arrangement of hardware components and their interconnections.

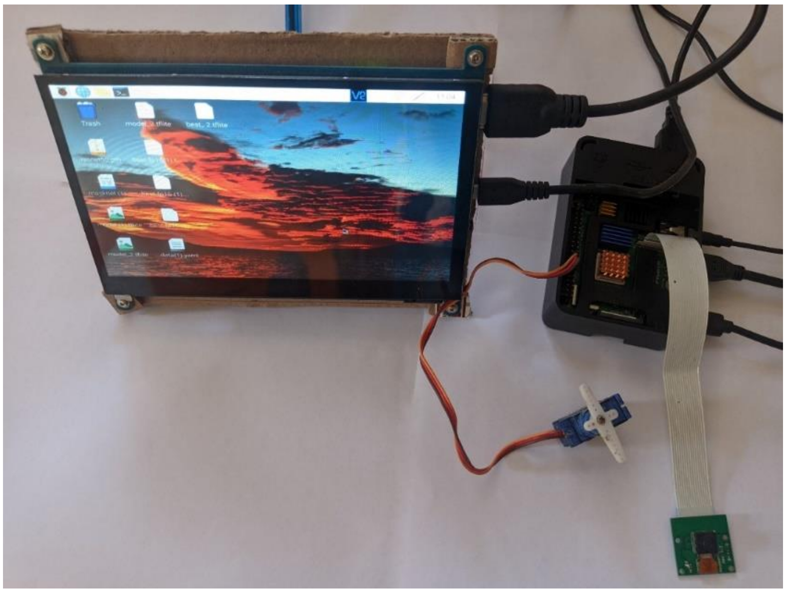

Figure 11. The complete arrangement of hardware components and their interconnections

In order to evaluate the effectiveness of our approach for a face mask detection in unlocking doors, we tested the YOLOv5 Lite model on a video stream. The Figure 12 below shows the different results obtained from the model when detecting face masks in the video stream. To make it convenient for users to operate the system, a graphical user interface (GUI) is designed to provide an easily accessible interface for human-machine interaction.

a. Inference results for input frame size of 128

b. Inference results for input frame size of 256

Figure 12. Comparison of the model's inference results for two different input frame sizes. The subfigures show the output predictions of the model for each frame size, with different colors representing different classes.

4. CONCLUSION

In conclusion, our research project has successfully demonstrated the benefits of using the TinyML approach to deploy machine learning on the Raspberry Pi for improving the application of door lock systems with face mask detection in public spaces and high-risk areas prone to the rapid spread of coronaviruses, including COVID-19 and seasonal common cold viruses. By utilizing the power of TinyML, we were able to develop a highly efficient and accurate machine learning model based on the YOLOv5 architecture.

The integration of TinyML technology with the Raspberry Pi allows for real-time face mask detection, enabling enhanced security measures to be implemented seamlessly. Our research showcases the practical implications of this technology, as it can significantly contribute to public health and safety during pandemics or outbreaks.

With an impressive mean Average Precision (mAP) score of 0.99, our machine learning model can accurately identify whether individuals are wearing masks correctly, incorrectly, or not at all, providing valuable information for implementing appropriate measures in public spaces. The capability to run the model efficiently on the Raspberry Pi, achieving an inference speed of 10 frames per second for 128-frame size input, further highlights the practicality and effectiveness of our approach.

By leveraging the benefits of TinyML and deploying it on accessible hardware like the Raspberry Pi, our research offers a cost-effective and scalable solution for face mask detection in various settings. This technology can enhance the security measures in public spaces, mitigating the risk of virus transmission and contributing to overall public health and safety.

We believe that our findings provide a solid foundation for future advancements in using TinyML technology for a wide range of applications, particularly those aimed at addressing health-related challenges in public spaces. It is our hope that our research inspires further exploration and adoption of TinyML-based solutions to improve security measures during pandemics or outbreaks, ultimately safeguarding communities from the spread of infectious diseases.

REFERENCES

[1] "Coronavirus disease (COVID-19) prevention and treatment methods and effective parameters: A systematic literature review," *Sustainable Cities and Society*, vol. 64, p. 102568, Jan. 2021, doi: 10.1016/j.scs.2020.102568.

[2] A. Bassi, B. M. Henry, L. Pighi, L. Leone, and G. Lippi, "Evaluation of indoor hospital acclimatization of body temperature before COVID-19 fever screening," *Journal of Hospital Infection*, vol. 112, pp. 127–128, Jun. 2021, doi: 10.1016/j.jhin.2021.02.020.

[3] D. K. Jain, Z. Zhang, and K. Huang, "Multi angle optimal pattern-based deep learning for automatic facial expression recognition," *Pattern Recognition Letters*, vol. 139, pp. 157–165, Nov. 2020, doi: 10.1016/j.patrec.2017.06.025.

[4] N. Zeng, H. Li, and Y. Peng, "A new deep belief network-based multi-task learning for diagnosis of Alzheimer's disease," *Neural Comput & Applic*, Jun. 2021, doi: 10.1007/s00521-021-06149-6.

[5] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection." arXiv, Apr. 22, 2020. doi: 10.48550/arXiv.2004.10934.

[6] J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement.” arXiv, Apr. 08, 2018. doi: 10.48550/arXiv.1804.02767.
[7] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” in *Advances in Neural Information Processing Systems*, Curran Associates, Inc., 2015. Accessed: Apr. 09, 2023. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/14bfa6bb14875e45bba028a21ed38046-Abstract.html
[8] H. Law and J. Deng, “CornerNet: Detecting Objects as Paired Keypoints,” presented at the Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 734–750. Accessed: Apr. 09, 2023. [Online]. Available: https://openaccess.thecvf.com/content_ECCV_2018/html/Hei_Law_CornerNet_Detecting_Objects_ECCV_2018_paper.html
[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, “Focal Loss for Dense Object Detection,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.
[10] Z. Tian, C. Shen, H. Chen, and T. He, “FCOS: Fully Convolutional One-Stage Object Detection,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 9626–9635. doi: 10.1109/ICCV.2019.00972.
[11] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, “BlazeFace: Sub-millisecond Neural Face Detection on Mobile GPUs.” arXiv, Jul. 14, 2019. doi: 10.48550/arXiv.1907.05047.
[12] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, “Fighting against COVID-19: A novel deep learning model based on YOLO-v2 with ResNet-50 for medical face mask detection,” *Sustainable Cities and Society*, vol. 65, p. 102600, Feb. 2021, doi: 10.1016/j.scs.2020.102600.
[13] U. Mahbub, S. Sarkar, and R. Chellappa, “Partial face detection in the mobile domain,” *Image and Vision Computing*, vol. 82, pp. 1–17, Feb. 2019, doi: 10.1016/j.imavis.2018.12.003.
[14] G. Zheng and Y. Xu, “Efficient face detection and tracking in video sequences based on deep learning,” *Information Sciences*, vol. 568, pp. 265–285, Aug. 2021, doi: 10.1016/j.ins.2021.03.027.
[15] 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, May 2021, doi: 10.1007/s11042-021-10711-8.
[16] A. Alzu’bi, F. Albalas, T. AL-Hadhrami, L. B. Younis, and A. Bashayreh, “Masked Face Recognition Using Deep Learning: A Review,” *Electronics*, vol. 10, no. 21, Art. no. 21, Jan. 2021, doi: 10.3390/electronics10212666.
[17] A. A. Abed, A. Al-Ibadi, and I. A. Abed, “Real-time multiple face mask and fever detection using YOLOv3 and TensorFlow lite platforms,” *Bulletin EEI*, vol. 12, no. 2, pp. 922–929, Apr. 2023, doi: 10.11591/eei.v12i2.4227.
[18] B. Qin and D. Li, “Identifying Facemask-Wearing Condition Using Image Super-Resolution with Classification Network to Prevent COVID-19,” *Sensors*, vol. 20, no. 18, Art. no. 18, Jan. 2020, doi: 10.3390/s20185236.
[19] T. Nurindini, M. N. Swacaesar, R. W. Astika, H. Purwanto, R. A. Wijayanti, and M. Taufik, “Design of Smart Door Lock System Using Face Recognition and Mask Detection Based on Viola-Jones Algorithm with Android Integration,” vol. 13, 2023.
[20] X. Su, M. Gao, J. Ren, Y. Li, M. Dong, and X. Liu, “Face mask detection and classification via deep transfer learning,” *Multimed Tools Appl*, vol. 81, no. 3, pp. 4475–4494, Jan. 2022, doi: 10.1007/s11042-021-11772-5.
[21] A. K. Gaddam and B. K. Menugonda, “Face Mask Detection and Door Unlocking System Using Deep Learning,” *IJRASET*, vol. 10, no. 12, pp. 651–655, Dec. 2022, doi: 10.22214/ijraset.2022.47965.
[22] M. Loey, G. Manogaran, M. H. N. Taha, and N. E. M. Khalifa, “A hybrid deep transfer learning model with machine learning methods for face mask detection in the era of the COVID-19 pandemic,” *Measurement*, vol. 167, p. 108288, Jan. 2021, doi: 10.1016/j.measurement.2020.108288.
[23] “The Pascal Visual Object Classes Challenge: A Retrospective | SpringerLink.” https://link.springer.com/article/10.1007/s11263-014-0733-5 (accessed Apr. 02, 2023).
[24] T.-Y. Lin *et al.*, “Microsoft COCO: Common Objects in Context.” arXiv, Feb. 20, 2015. doi: 10.48550/arXiv.1405.0312.
[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection.” arXiv, May 09, 2016. doi: 10.48550/arXiv.1506.02640.
[26] G. Jocher *et al.*, “ultralytics/YOLOv5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation.” Zenodo, Nov. 22, 2022. doi: 10.5281/zenodo.7347926.
[27] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, “Real-Time Vehicle Detection Based on Improved YOLO v5,” *Sustainability*, vol. 14, no. 19, Art. no. 19, Jan. 2022, doi: 10.3390/su141912274.
[28] “Face Mask Detection.” https://www.kaggle.com/datasets/andrewmvd/face-mask-detection (accessed Mar. 08, 2023).
[29] M. Altayeb and A. Al-Ghraibah, “Voice controlled Camera Assisted Pick and Place Robot Using Raspberry Pi,” *Indonesian Journal of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 51–56, 2022, doi: 10.52549/ijeei.v10i1.3636.
[30] J. Ieamsaard, S. N. Charoensook, and S. Yammen, “Deep Learning-based Face Mask Detection Using YOLOV5,” in *2021 9th International Electrical Engineering Congress (iEECON)*, Mar. 2021, pp. 428–431. doi: 10.1109/iEECON51072.2021.9440346.
[31] I. ben Abdel Ouahab, L. Elaachak, M. Bouhorma, and Y. A. Alluhaidan, “Real-time Facemask Detector using Deep Learning and Raspberry Pi,” in *2021 International Conference on Digital Age & Technological Advances for Sustainable Development (ICDATA)*, Jun. 2021, pp. 23–30. doi: 10.1109/ICDATA52997.2021.00014.