Qubahan Techno Journal

Real Time Face-mask Detection with Arduino to Prevent COVID-19 Spreading

Ridwan B. Marqas
Department of Information Technology
Duhok
Duhok, Iraq
pgmr.red@gmail.com

***Abstract*— The rise of COVID-19 pandemic has had a lasting impact in many countries worldwide since 2019. Face-mask detection had been significant progress in the Image processing and deep learning fields studies. Many face detection models have been designed using different algorithms and techniques. The proposed approach in this paper developed to avoid mask-less people from entering to a desired places (i.e. Mall, University, Office, …etc.) by detecting face mask using deep learning, TensorFlow, Keras, and OpenCV and sending a signal to Arduino device that connected to the gate to be open. it detect a face in a real-time and identifies whether the person wear mask or not. The method attains accuracy up to 97.80%. The dataset provided in this paper, was collected from various sources.**

Keywords*— *face mask detection, deep learning, TensorFlow, Keras, OpenCV, Arduino.

I. Introduction

According to World Health Organization (WHO), since December 2019 more than 114 countries suffered from COVID-19 pandemic which has declared as a deadly diseases that has globally infected over 110 million people causing more than 2.43 million deaths in the worldwide as on Feb 18, 2021, In Iraq since the rise of COVID-19 more than 653000 effected case and more than 13000 deaths has been registered, this is Deu to the insufficient vaccinees to overcome this deadly disease till date [1]. Wearing a face-mask during this pandemic is a critical defensive in times when social distancing is hard to maintain. Therefore, many face mask detection and monitoring systems have been developed to provide effective supervision for hospitals, public transportations, airports, retail locations, and sports venues.

Over years, in the field of image processing, computer vision and pattern recognition, face detection is the very first step for various applications that depends facial analysis algorithms for identifies, recognize human faces and also to capturing facial motions in digital images, including the face recognition, face alignment, face verification, age recognition, face modelling, face authentication, access control, forensics, and human-computer interactions. face relighting, facial expression tracking, head pose tracking, facial expression recognition, gender recognition, and other face-detection based applications [2].

After the arise of Covid-19, the Face-Mask detection has widely considered problem in the image processing field. This technology is currently more appropriate because it is applied to detect faces and to identify people wearing masks in images, videos and also in real-time vision. By using deep learning and convolution neural network (CNN) techniques, it become possible to achieve high accuracy results in image classification and object detection applications.

Creating a system for detecting the face-mask will provide a way for controlling the people who enters any places. The proposed system in this paper uses deep learning, TensorFlow, Keras, and OpenCV which are used as an image classifier to detect face-mask and sends a signal to Arduino device that control the open and close of door.

II. Face Detection Overview

Face detection is a technique for recognizing or confirming an individual's identity by looking at their face. Face recognition software can identify people in pictures, videos, or in real time.

Over the past 60 years as shown in Fig-1, face detection methods widely used in various industries and have benefitted from the improvements in this technology and these include: law enforcement, border control, retail, mobile technology and banking and finance [3, 4, 5, 6].

- 1964: Bledso did a facial programming experiment. They imagine a semi-automnmatic input method, in which the operators enter twenty different measures, such as the size of the mouth and eyes

- 1977: 21 new markers were added to the Bledso 1964 system to improved it (i.e. , width of lips, eyes color, hair color).
- 1988: Artificial intelligence was used to improve previously used computational methods that exposed multiple flaws. Mathematics (“linear algebra”) are used to view symbols uniquely and to find a way to simplify and modify them independently of human markers.
- 1991: EIGENFACES which was the first successful techniques used in facial recognition technology, that depend on the statistical Principal component analysis (PCA) method, Was developed by Pentland and Turk of the Massachusetts Institute of Technology (MIT)
- 1998: Face recognition technology “FERET program” developed by the Defense Advanced Research Projects Agency (DARPA), created a database of 2400 images for 850 persons of deferent age and gender.
- 2005: The Face Recognition Grand Challenge (FRGC) was created to promote and improve face recognition technologies that would complement existing facial recognition initiatives.
- 2011: using deep learning and machine learning techniques that depend on artificial neural networks, enables The system to selects a point for the comparison: in large databases.
- 2014: Facebook's internal algorithm, Deepface, allows it to identify faces. According to the social network, the process comes close to matching the output of the human eye in approximately 97% of images.
- 2017: Apple launched a facial recognition technology in its updates, and its use has expanded to retail and banking.
- 2017: Selfie Pay is a facial recognition system for online transactions developed by Mastercard.
- In 2018, Chinese police used a smart monitoring system focused on live facial recognition to arrest a suspect of "economic crime" at a concert where his face was recognized in a crowd of 50,000 people after being identified in a national database.
- From 2019, People who want to purchase a new phone in China will now agree to have their faces scanned by the operator.

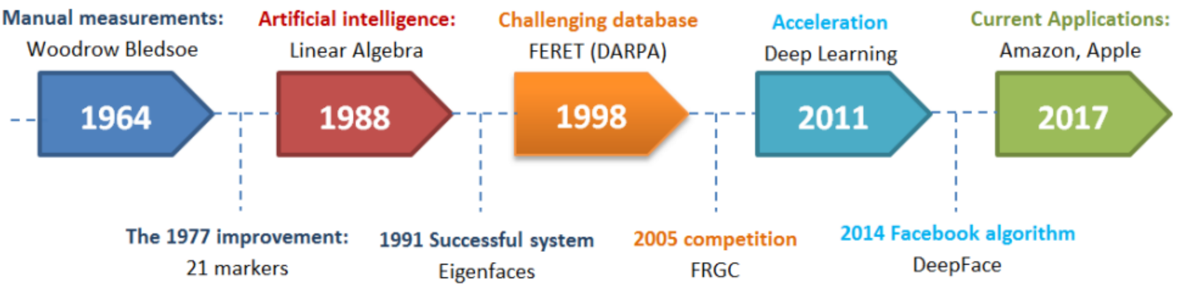
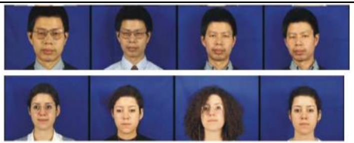
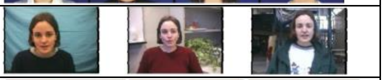


Figure 1. Primary stages in the history of face recognition.

Generally, all face detection system uses dataset of images for testing, evaluating and comparing the system accuracy in verifying or identifying individuals as shown in table 1 [4].

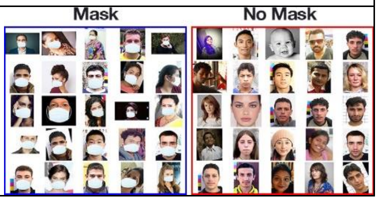
TABLE I. FACE DETECTION DATASETS

no	Dataset Name	Sample
1.	ORL Dataset	
2.	FERET Dataset	
3.	AR Dataset	

4.	XM2VTS Dataset	
5.	BANCA dataset	
6.	FRGC dataset	
7.	*LFW Database*	
8.	*CFP Dataset*	
9.	*IARPA Janus Benchmark-A*	
10.	*DMFD Database*	
11.	*VGGFACE Database*	
12.	*VGGFACE2 Database*	

The identification of face masks is a difficult task for the existing proposed face detector models. As in all normal face detections systems, face-mask detection system depends on evaluating and comparing the detected face with datasets for finding the face and then determining whether the person wear a mask on it or not. The data used in face mask detection model consists of categories: (1) without face mask, (2) with face mask, as shown in table 2.

TABLE II. FACE MASK DETECTION DATASETS

No.	Dataset Name	Sample
1.	Masked face detection dataset (MFDD)	
2.	Real-world masked face recognition dataset (RMFRD)	

3.	Simulated masked face recognition dataset (SMFRD)	

III. Deep Learning

Deep Learning (DL) is basically a subpart of Machine Learning (ML) model which involves algorithms that concerned with algorithms inspired by the structure and function of the brain and uses multilayer neural networks called artificial neural networks [7,8,9]. Basically, both DL and ML belongs to the higher field Artificial Intelligence (AI) as shown in Fig.2. DL depends on Neural Network layers that transforms the input in some way to produce output to implement its functioning. In deep learning, Image can be called as “matrix of pixel values”, therefore it can be an more easier to classify complex images matrix or images with similar forms of matrix or a very huge dataset of images with minimal changes in the matrix by depending on deep learning matrix classification. This may lead to clash in prediction scores and thereby affecting the accuracy and speed of classifier model [8,9].

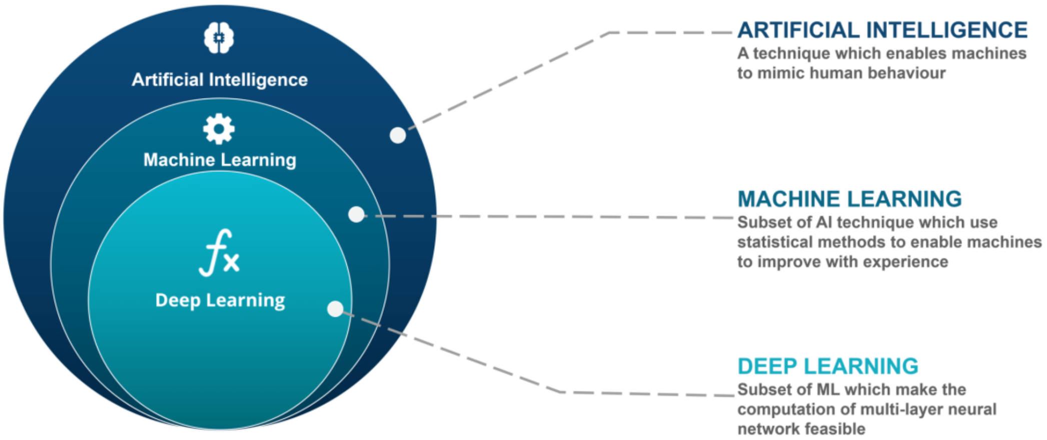

Artificial Intelligence
Machine Learning
f_x
Deep Learning
ARTIFICIAL INTELLIGENCE
A technique which enables machines to mimic human behaviour
MACHINE LEARNING
Subset of AI technique which use statistical methods to enable machines to improve with experience
DEEP LEARNING
Subset of ML which make the computation of multi-layer neural network feasible

Figure 2. AI subfields.

IV. Convolutional Neural Network

Convolutional Neural Network (CNN) is a Deep Learning algorithm that have the ability of assigning biases and weights to different objects in an Image and on basis of the same it can differentiate one image from another. It consists of processing different layers of Image Classification and it is designed with means of representing functioning of Neurons in Human Brain as explained in [10, 11, 12].

V. Deep Learning frameworks and libraries

Many popular ML frameworks and libraries already offer the possibility to use GPU accelerators to speed up learning process with supported interfaces (DLwiki 2018; Felice 2017; Kalogeiton et al. 2016). Some of them also allow the use optimised libraries such as CUDA (cuDNN) [10, 11], and OpenCL to improve the performance even further. The main feature of many-core accelerators is that their massively parallel architecture allows them to speed up computations that involve matrix-based operations [12]. The software development in the ML/DL direction community is highly dynamic and has various layers of abstraction as depicted in Fig. 3.

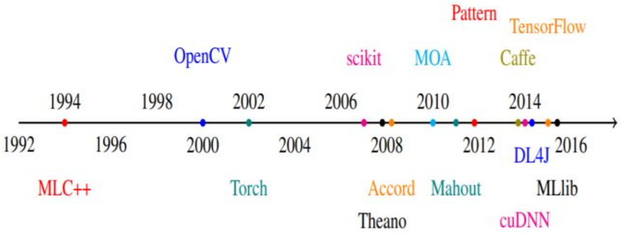


Figure 3, A timeline showing machine-learning libraries

A. *Tenserflow*

TensorFlow is an open source library with a large number of pre-designed models that are useful in Machine Learning and particularly in Deep Learning. The term TenserFlow consists of two parts “Tenser”, which considers as an Array of N-

Dimensional, and “Flow” considers graph of operations. TensorFlow is an open-source software library for numerical computation using data flow graphs. TensorFlow is designed for large-scale distributed training and inference. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them [13].

TensorFlow was created and is maintained by the Google Brain team within Google’s Machine Intelligence research organization for Machine Learning (ML) and Deep Learning (DL). It is currently released under the Apache 2.0 open-source license. The distributed TensorFlow architecture contains distributed master and worker services with kernel implementations. These include 200 standard operations, including mathematical, array manipulation, control flow, and state management operations written in C++. TensorFlow was designed for use both in research, development and production systems. It can run on single CPU systems, GPUs, mobile devices and large-scale distributed systems of hundreds of nodes.. TensorFlow programming interfaces include APIs for Python and C++ and developments for Java, GO, R, and Haskell are on the way. TensorFlow is also supported in Google and Amazon cloud environments [13].

B. OpenCV

Open Source Computer Vision Library (OpenCV) is an open source computer vision and machine learning software library, is utilized to differentiate and recognize faces, recognize objects, group movements in recordings, trace progressive modules, follow eye gesture, track camera actions, expel red eyes from pictures taken utilizing flash, find comparative pictures from an image database, perceive landscape and set up markers to overlay it with increased reality and so forth [14,15]. Now since all the training and classification is ready to be executed when it needed an eye for the designed system to capture real-time images (Face, Hand Gestures, Fashion) which can then be sent for classification and identification. OpenCV adds intelligence to Deep Learning models for visualization image processing [8].

C. Keras

Keras gives fundamental reflections and building units for creation and transportation of ML arrangements with high iteration velocity. It takes full advantage of the scalability and cross-platform capabilities of TensorFlow. The core data structures of Keras are layers and models [16]. All the layers used in the CNN model are implemented using Keras. Along with the conversion of the class vector to the binary class matrix in data processing, it helps to compile the overall model[16].

VI. ARDUINO

Arduino is an open-source platform used for building electronics projects. Arduino consists of both a physical programmable circuit board (often referred to as a microcontroller) and a piece of software, or IDE (Integrated Development Environment) that runs on your computer, used to write and upload computer code to the physical board[17].

The Arduino platform has become quite popular with people just starting out with electronics, and for good reason. Unlike most previous programmable circuit boards, the Arduino does not need a separate piece of hardware (called a programmer) in order to load new code onto the board -- you can simply use a USB cable. Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn to program. Finally, Arduino provides a standard form factor that breaks out the functions of the micro-controller into a more accessible package [17].

The proposed method has been designed by Python that are connected by a USB to an Arduino uno micro controller. It require to have PySerial software, which is a Python API module uses for reading and writing serial data from Arduino or any other Microcontroller. It encapsulates the access for the serial port, and provides backends for Python running on Windows, OSX, Linux, BSD (possibly any POSIX compliant system) and IronPython. The module named “serial” automatically selects the appropriate backend [17,18,19]. PySerial have many features such as:

- Same class based interface on all supported platforms.
- Access to the port settings through Python properties.
- Support for different byte sizes, stop bits, parity and flow control with RTS/CTS and/or Xon/Xoff.
- Working with or without receive timeout.
- File like API with “read” and “write” (“readline” etc. also supported).
- The files in this package are 100% pure Python.
- The port is set up for binary transmission. No NULL byte stripping, CR-LF translation etc. (which are many times enabled for POSIX.) This makes this module universally useful.
- Compatible with io library

- RFC 2217 client (experimental), server provided in the examples.

VII. Methodology

The system is designed to detect the faces in a real-time video and to determine whether the person wears a face mask or not. Using the detected data, system can decide whether the concerned person can be allowed inside public places such as (market, or a hospital) or not allowed. This project can be used in the hospital, market, bus terminals, restaurants, and other public gatherings where the monitoring has to be done.

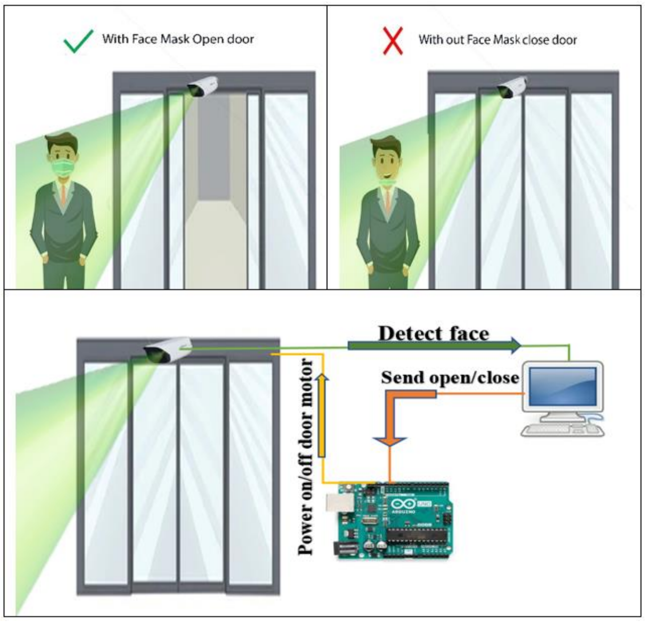


Figure 4, Real Time Face detection

This project basically consists of, a camera placed in front of door, the camera sends video frames to OpenCV running on a Windows PC. If OpenCV detects a face it will capture the image of the people entering public places, then the detected face image is sent to a the system to be compared (tested) whether the person wears a face mask or not using Tenserflow. After the data processing, TenserFlow will send the result to Arduino whether to open the door or not as shown in Fig.4.

A. *The system requirement*

The following software's and hardware's requires to run the system.

1) *Software requirement*

- Python: 3.4 and newer
- PySerial
- tensorflow>=1.15.2
- keras==2.3.1
- imutils==0.5.3
- numpy==1.18.2
- opencv-python==4.2.0.*
- matplotlib==3.2.1
- scipy==1.4.1

2) *hardware requirement*

- Camera: esp32-cam or any other webCam
- computer
- Arduino uno
- Relay

- Slide Door

B. System overview

the system consists software and hardware phases.

Software phase is responsible of training the dataset images and then applying the capturing face image from real-time video then identifying the mask, while the hardware phase open/close the slide-door according to software phase results as shown in Fig.5.

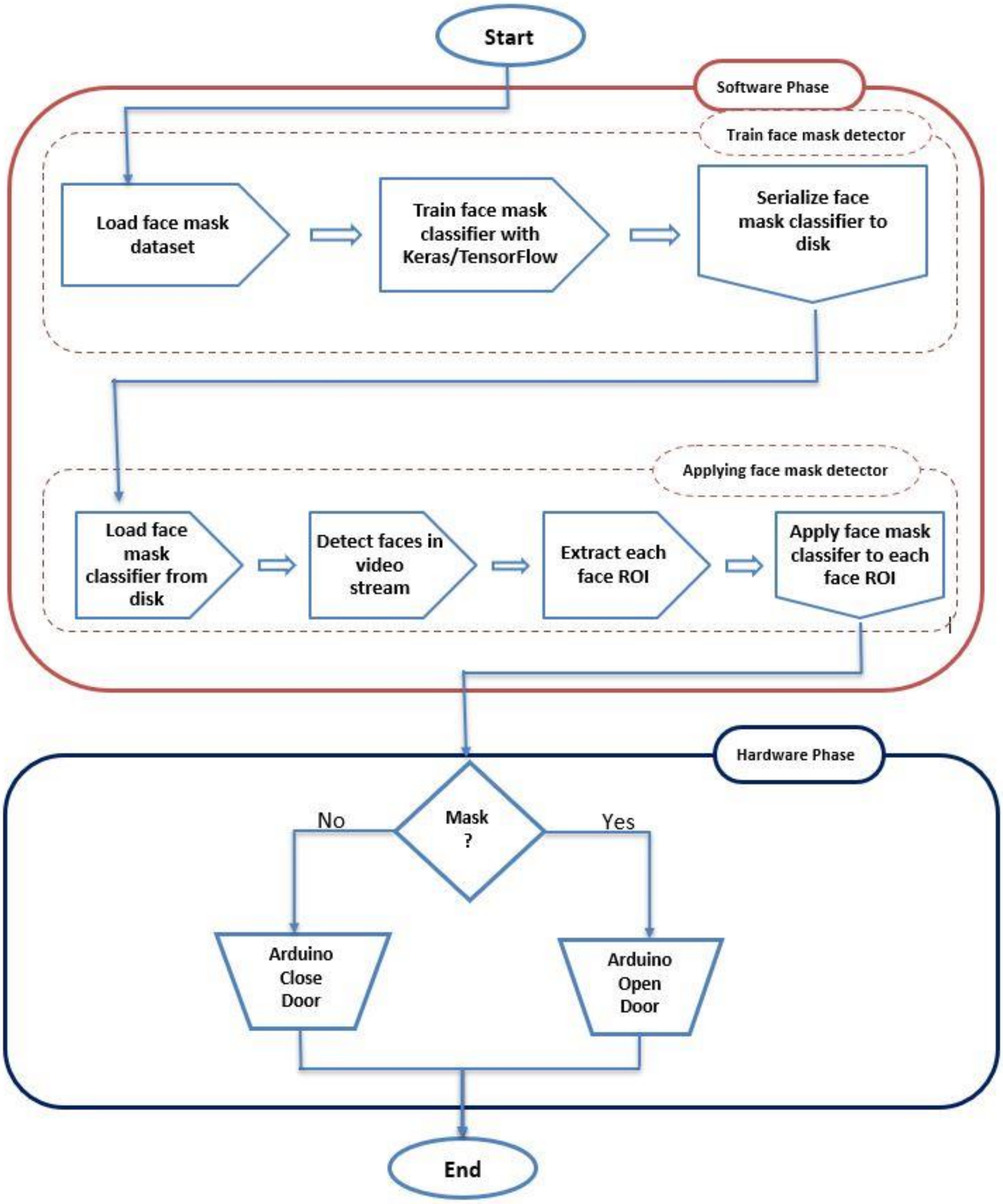

Start
Software Phase
Train face mask detector
Load face mask dataset
Train face mask classifier with Keras/TensorFlow
Serialize face mask classifier to disk
Applying face mask detector
Load face mask classifier from disk
Detect faces in video stream
Extract each face ROI
Apply face mask classifer to each face ROI
Hardware Phase
Mask ?
No
Yes
Arduino Close Door
Arduino Open Door
End

Figure 5, Real Time Face detection Flowchart

VIII. RESULTS

All the experimental trials have been conducted on a laptop equipped by an Intel i7-9750H processor (2.6 GHz), 32 GB of RAM with INVIDIA GTX1660ti with 6 GB of VRAM. The Spyder Notebook software equipped with Python 3.8 kernel was selected in this research for the development and implementation of the different experimental trails.

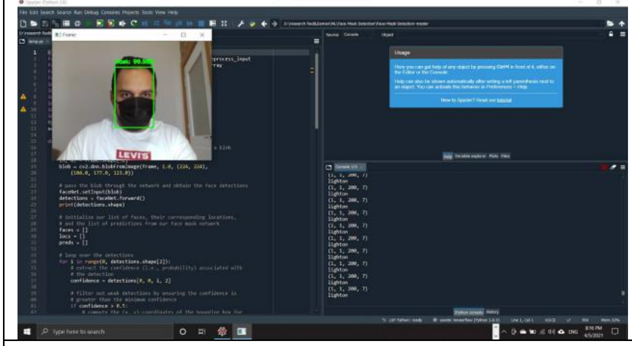
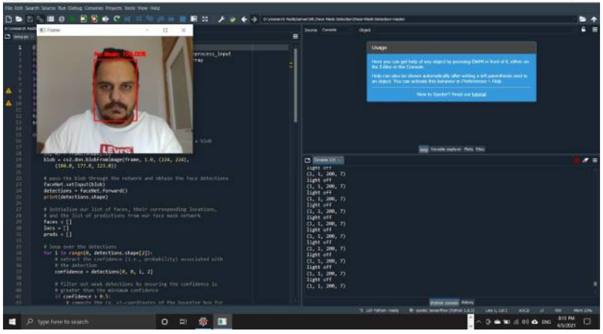
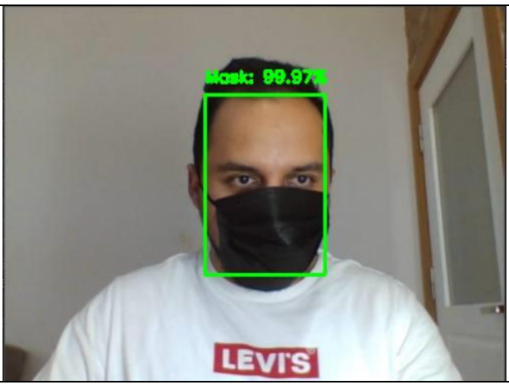
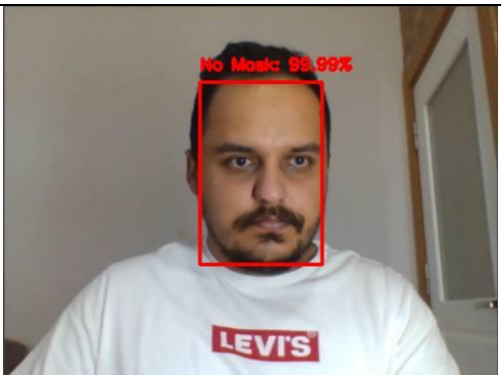
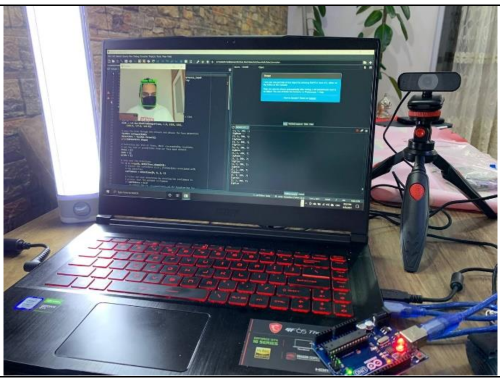
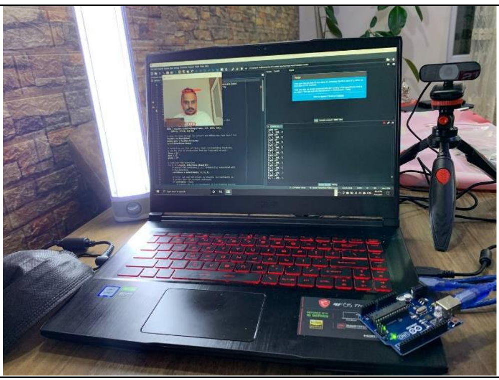
The exprimental results in Table-3, shows that the system works with high accuary that reach 99% in detecting Face mask.

The Real time face detection system firstly has been tested in two different ways:

(1) by testing it only in Arduino led on/off.

(2)by testing it by connecting it on a real slide door

TABLE III. EXPREMENTAL RESULTS

With Face Mask	**Without Face Mask**
	
(a)	
 	
(b)	
	
(c)	

```
blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
    (104.0, 177.0, 123.0))

# pass the blob through the network and obtain the face detections
faceNet.setInput(blob)
detections = faceNet.forward()
print(detections.shape)

# initialize our list of faces, their corresponding locations,
# and the list of predictions from our face mask network
faces = []
locs = []
preds = []

# loop over the detections
for i in range(0, detections.shape[2]):
    # extract the confidence (i.e., probability) associated with
    # the detection
    confidence = detections[0, 0, i, 2]

    # filter out weak detections by ensuring the confidence is
    # greater than the minimum confidence
    if confidence > 0.5:
        # compute the (x, y)-coordinates of the bounding box for
```

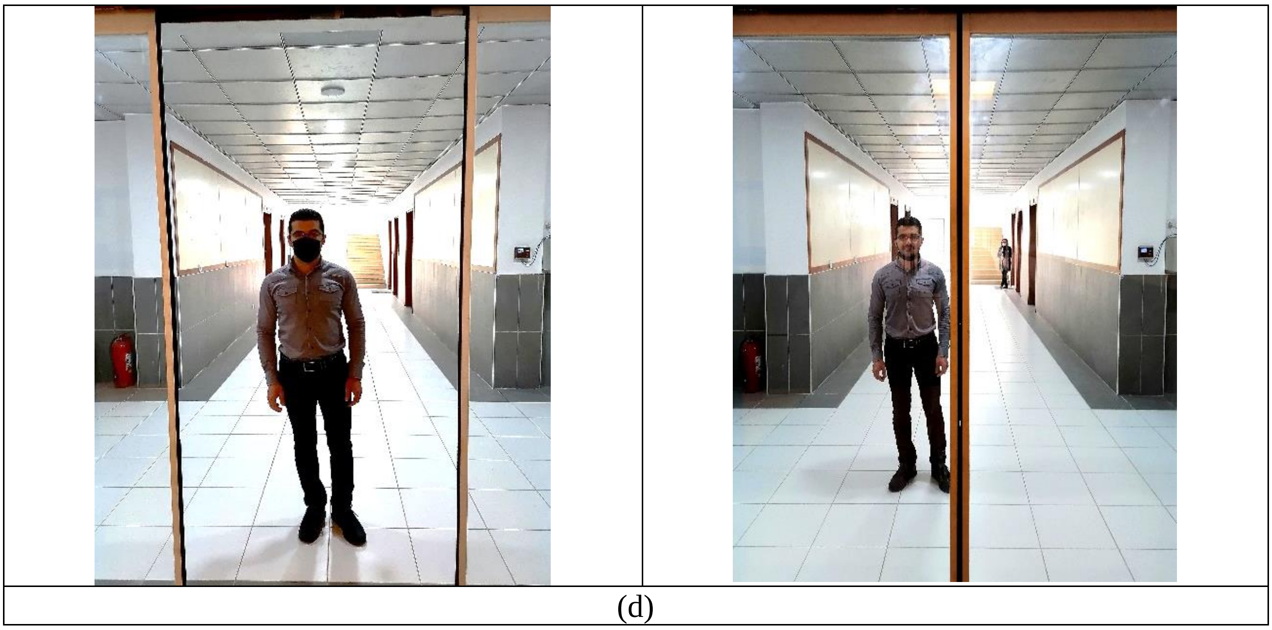
```
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
(1, 1, 200, 7)
lighton
```

```
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
light off
(1, 1, 200, 7)
```

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in Preferences > Help.

New to Spyder? Read our tutorial



(d)

IX. Conclusion

Modern artificial intelligence systems and machine learning algorithms have revolutionized approaches to scientific and technological challenges in a variety of fields. Nowadays Deep Learning (DL) and Machine Learning (ML) techniques has been a vary useful tools in solving problems.

In the proposed real-time-face mask detection model Deep Learning libraries has been used to detect face and determine whether the person wear a mask or not. from the experimental result table, is seems that the real time face-detection system has a high accuracy in detecting mask, this help to control the spreading of COVID-19 in public places by preventing the people to enter it without wearing a face mask.

References

[1] E. Dong, H. Du and L. Gardner, "An interactive web-based dashboard to track COVID-19 in real time", The Lancet Infectious Diseases, vol. 20, no. 5, pp. 533-534, 2020. Available: 10.1016/s1473-3099(20)30120-1 [Accessed 6 April 2021].

[2] P.S. Othman, R.R. Ihsan, R.B. Marqas, S.M. Almufti, "Image Processing Techniques for Identifying Impostor Documents Through Digital Forensic Examination". Image Process. Tech. 2020, 62, 1781–1794.

[3] P. Othman, R. Marqas, D. Abdulqader and S. Almufti, "Effect of Mean Filter on Face Image from Video Frames", 2020 8th International Symposium on Digital Forensics and Security (ISDFS), 2020. Available: 10.1109/isdfs49300.2020.9116277 [Accessed 3 April 2021].

[4] I. Adjabi, A. Ouahabi, A. Benzaoui and A. Taleb-Ahmed, "Past, Present, and Future of Face Recognition: A Review", Electronics, vol. 9, no. 8, p. 1188, 2020. Available: 10.3390/electronics9081188.

[5] R. Ihsan, S. Almufti and R. Marqas, "A Median Filter With Evaluating of Temporal Ultrasound Image for Impulse Noise Removal for Kidney Diagnosis", Journal of Applied Science and Technology Trends, vol. 1, no. 2, pp. 71-77, 2020. Available: 10.38094/jastt1217 [Accessed 3 April 2021].

[6] R. Marqas, "Implementation and Experiments on Fingerprint Based Authentication System (FBAS) Using Delaunay Triangulation and Voronoi Diagram" (2017). [online] Hdl.handle.net. Available at: http://hdl.handle.net/11129/4191 .

[7] R. Rajab Asaad and R. Masoud Abdulhakim, "The Concept of Data Mining and Knowledge Extraction Techniques", QAJ, vol. 1, no. 2, pp. 17–20, Mar. 2021.

[8] R. Rajab Asaad, "Review on Deep Learning and Neural Network Implementation for Emotions Recognition", Qubahan Academic Journal, vol. 1, no. 1, 2021. Available: 10.48161/qaj.v1n1a25. Available: https://doi.org/10.48161/qaj.v1n2a43.

[9] S. M. Almufti, "Historical survey on metaheuristics algorithms", International Journal of Scientific World, vol. 7, no. 1, p. 1, 2019. Available: 10.14419/ijsw.v7i1.29497 [Accessed 6 April 2021].

[10] R. Asaad and R. Ali, "Back Propagation Neural Network(BPNN) and Sigmoid Activation Function in Multi-Layer Networks", Academic Journal of Nawroz University, vol. 8, no. 4, p. 216, 2019. Available: 10.25007/ajnu.v8n4a464 [Accessed 6 April 2021].

[11] R. Asaad, "Güler and Linaro et al Model in an Investigation of the Neuronal Dynamics using noise Comparative Study", Academic Journal of Nawroz University, vol. 8, no. 3, p. 10, 2019. Available: 10.25007/ajnu.v8n3a360 [Accessed 6 April 2021].

[12] R. Asaad. "An Investigation of the Neuronal Dynamics Under Noisy Rate Functions". Thesis (M.S.), Eastern Mediterranean University, Institute of Graduate Studies and Research, Dept. of Computer Engineering, Famagusta. 2014: North Cyprus. http://hdl.handle.net/11129/1619

[13] G. Nguyen et al., "Machine Learning and Deep Learning frameworks and libraries for large-scale data mining: a survey", Artificial Intelligence Review, vol. 52, no. 1, pp. 77-124, 2019. Available: 10.1007/s10462-018-09679-z [Accessed 4 April 2021].

[14] D. Someshwar, D. Bhanushali, V. Chaudhari and S. Nadkarni, "Implementation of Virtual Assistant with Sign Language using Deep Learning and TensorFlow", 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), 2020. Available: 10.1109/icirca48905.2020.9183179 [Accessed 4 April 2021].

[15] H. Lai and H. Lai, "Real-Time Dynamic Hand Gesture Recognition", 2014 International Symposium on Computer, Consumer and Control, 2014. Available: 10.1109/is3c.2014.177 [Accessed 4 April 2021].

[16] A. Das, M. Wasif Ansari and R. Basak, "Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV", 2020 IEEE 17th India Council International Conference (INDICON), 2020. Available: 10.1109/indicon49873.2020.9342585 [Accessed 4 April 2021].

[17] B. Mohammed and H. Ahmad, "Advanced car-parking security platform using Arduino along with automatic license and number recognition", Academic Journal of Nawroz University, vol. 10, no. 1, p. 1, 2021. Available: 10.25007/ajnu.v10n1a996 [Accessed 3 April 2021].

[18] P. Kazarinoff, "Using Python to control an Arduino", Python for Undergraduate Engineers, 2021. [Online]. Available: https://pythonforundergradengineers.com/python-arduino-LED.html. [Accessed: 05- Apr- 2021].

[19] C. Liechti,"pySerial — pySerial 3.0 documentation", Pythonhosted.org, 2021. [Online]. Available: https://pythonhosted.org/pyserial/pyserial.html#overview. [Accessed: 05- Apr- 2021].