

# Sistema identificador de cubrebocas en personas.



Jorge Irving Cristobal Pichardo  
Facultad de Ingeniería. Universidad Autónoma de Guerrero  
[jorgeirvingcristobalpichardo@gmail.com](mailto:jorgeirvingcristobalpichardo@gmail.com)

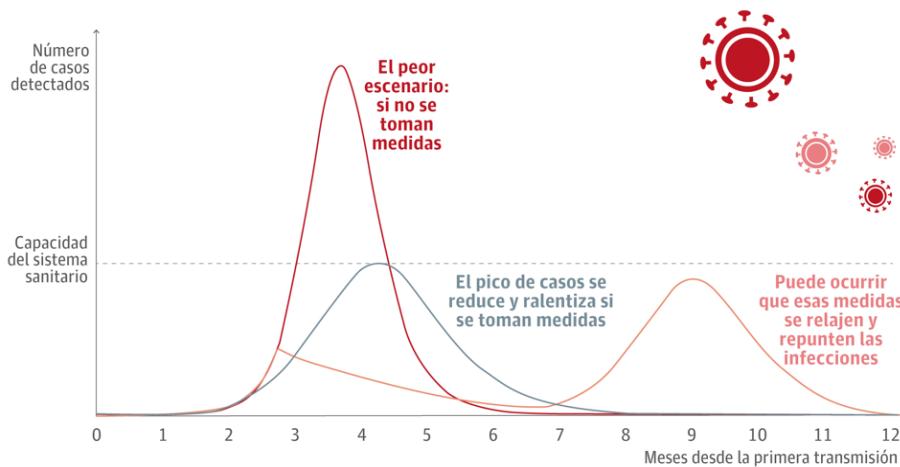
**Abstract.** Actualmente estamos viviendo una contingencia sanitaria mundial, a causa del SARS-CoV-2 [1], comunmente llamado Coronavirus o Covid-19, que ha provocado que la vida cotidiana se haya paralizado en proporciones nunca antes vistas, debido al confinamiento que se debe tomar para detener el avance de este virus.



**Palabras clave:** Redes Neuronales; Python; OpenCV; TensorFlow; Covid-19.

# 1 Introducción

El gobierno y los sistemas de salud tuvieron que tomar las medidas necesarias para evitar más contagios. Para esto se implementó, el uso obligatorio de cubrebocas en las calles [2] ya que los modelos matemáticos probabilísticos han hecho proyecciones de que sí se puede frenar la propagación como se muestra en la Figura 1 [3].

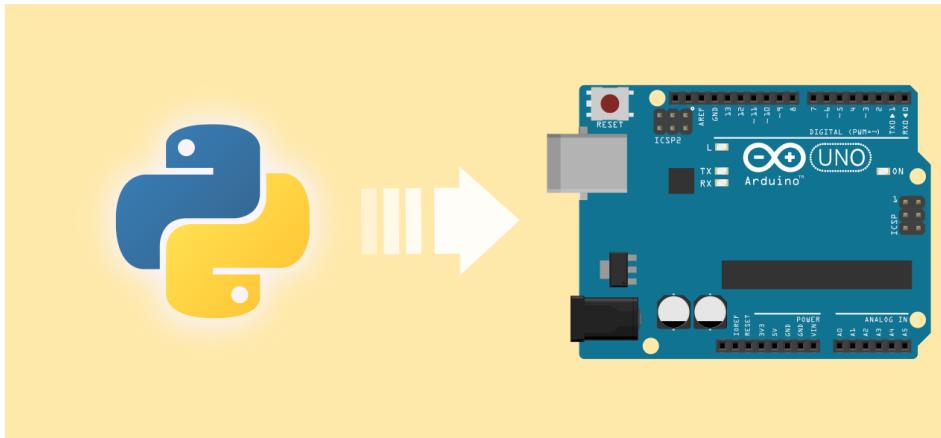


**Fig. 1.** Número de casos de Coronavirus dependiendo si se acatan o no las medidas.

Desafortunadamente no todas las personas pueden estar en confinamiento, ya que se necesitan estar activos los servicios con los que contamos en nuestros hogares.



Es por estos motivos que se ha propuesto el presente proyecto, para automatizar la revisión cubrebocas para tener acceso a lugares públicos. Retomado del proyecto del Doctor Adrián Rosebrock [4] sólo se hizo un afinamiento más al sistema y un acondicionamiento para poder conectarlo al microcontrolador Arduino [5].



## 2 Materiales y métodos

El proyecto del Dr. Rosebrock cuenta con 3 scripts programados en Python:

**train\_mask\_detector.py**  
**train\_mask\_detector.py**  
**detect\_mask\_video.py**

```
└── dataset
    ├── with_mask [690 entries]
    └── without_mask [686 entries]
└── examples
    ├── example_01.png
    ├── example_02.png
    └── example_03.png
└── face_detector
    ├── deploy.prototxt
    └── res10_300x300_ssd_iter_140000.caffemodel
└── detect_mask_image.py
└── detect_mask_video.py
└── mask_detector.model
└── plot.png
└── train_mask_detector.py

5 directories, 10 files
```

**Fig 2.** Directorio con el que cuenta el proyecto

## Mask



## No Mask



Fig 3. Imágenes de personas con y sin cubrebocas

## Librerías de Python...

TensorFlow, Keras, NumPy, Argparse, Imutils, Time, OpenCV, OS y PySerial.



## **Material de Arduino:**

1 Placa de Arduino Uno

1 cable USB para Arduino

Cables o Jumpers para las conexiones

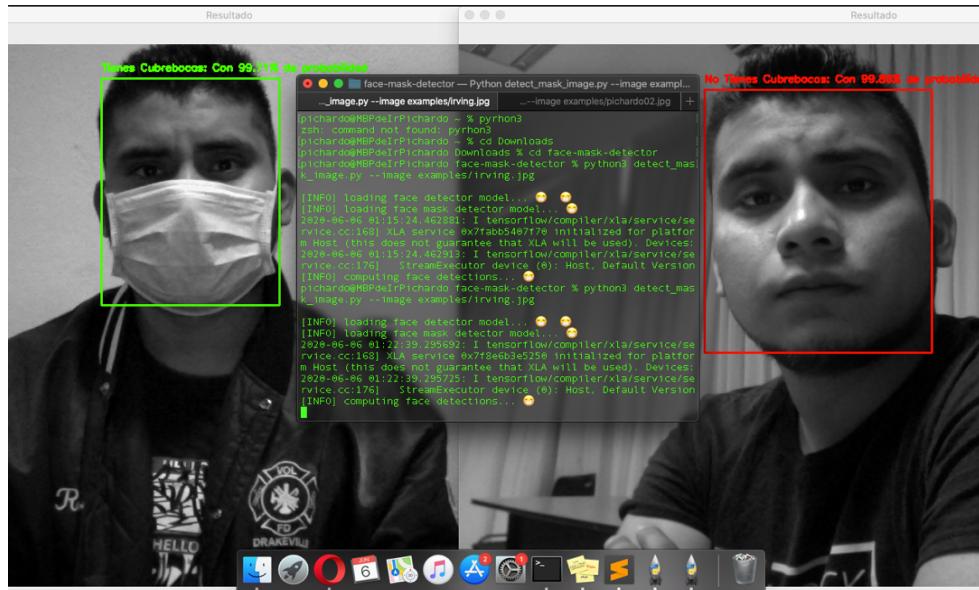
1 protoboard

Leds y resistencias (330, 220 o de 100 Ohms)

1 matriz de leds 8x8 para Arduino



Para detectar comprobar que el detector está funcionando de forma correcta ejecutamos el script *detect\_mask\_image.py* desde la Terminal del sistema: **python detect\_mask\_image.py --image examples/example\_01.png** cambiamos los nombres de las imágenes en la parte del comando para ver los ejemplos que se encuentran en la carpeta **/examples** y los resultados se pueden visualizar como la Figura 4.



**Fig. 4:** Muestra el resultado de ejecutar el archivo *detect\_mask\_image.py*

# 3 Propuesta



## 1.) Ejecución del archivo detect\_mask\_video.py

Este ultimo archivo que tendrá la carpeta servirá para poder accionar la cámara y poder detectar si la persona frente a ella cuenta con un cubrebocas o no y confirmar si está funcionando correctamente.



## 2.) Hacer pruebas de Python con Arduino

Para hacer nuestra conexión de Python con Arduino se realizan pruebas independientes para lograr el envío y recepción de información, y ubicar los puertos para la comunicación.



## 3.) Modificación del código de detect\_mask\_video.py

Programar independientemente el Arduino para que haga su función de enviar alertas y recibir las premisas para cumplir dicha tarea que al detectar si una persona tiene cubrebocas o no.



## 4.) Ejecución del programa y muestra de resultados



## Representación del programa.

Lo que se pretende conseguir es que el sistema detecte a través de la cámara si una persona tiene o no un cubreboca, en caso de que sea positivo, manda una alerta al microcontrolador para que este accione una alarma (se puede interpretar cómo el abrir de una puerta, etc.).

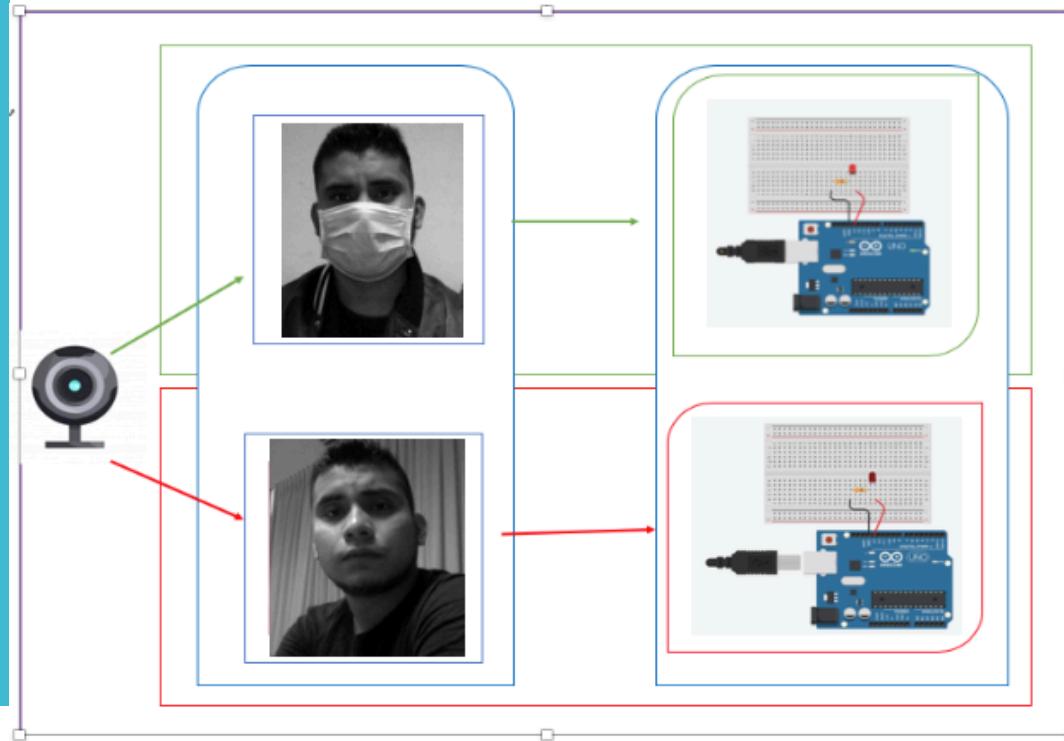


Fig. 5: Representación de cómo funcionaría el programa.

## Modificación del código *detect\_mask\_video.py*

```
125
126     # determinar la etiqueta de clase y el color que usaremos para dibujar
127     # el cuadro delimitador y el texto
128     label = "Tienes cubrebocas" if mask > withoutMask else "No Tienes cubrebocas"
129     color = (0, 255, 0) if label == "Tienes cubrebocas" else (0, 0, 255)
130
131     # incluye la probabilidad en la etiqueta
132     label = "{}: Con {:.2f}%".format(label, max(mask, withoutMask) * 100)+" de probabilidad"
133
134
135     import serial
136     arduino=serial.Serial(port='/dev/cu.usbmodem14201', baudrate=9600)
137     if mask==1:
138         arduino.write(mask)
139     else:
140         withoutMask==0
141         arduino.write(withoutMask)
142
143     # muestra la etiqueta y el rectángulo del cuadro delimitador en la salida
144     # marco
145     cv2.putText(frame, label, (startX, startY - 10),
146                 cv2.FONT_HERSHEY_SIMPLEX, 0.45, color, 2)
147     cv2.rectangle(frame, (startX, startY), (endX, endY), color, 2)
148
149     # muestra el marco de salida
150     cv2.imshow("Detector de Cubrebocas *2020*",frame)
151     key = cv2.waitKey(1) & 0xFF
152
```

Fig. 6: Modificación de líneas de código

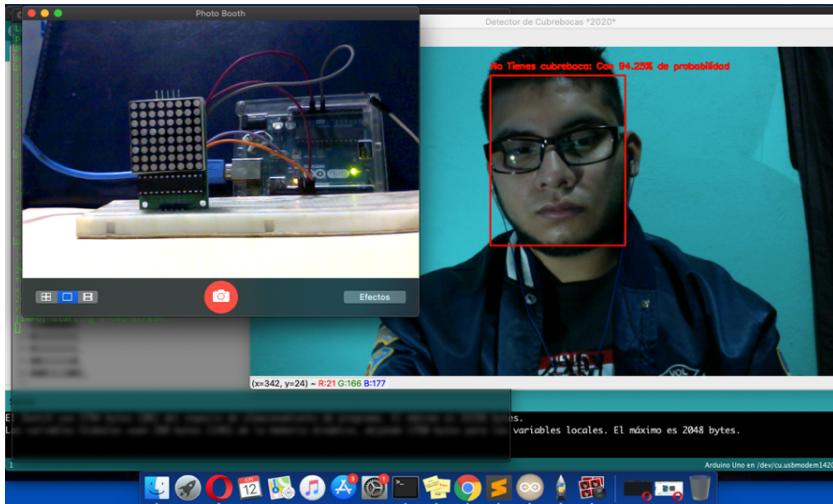
1.- Importamos aquí mismo la librería de PySerial

2.- Declaramos una variable llamada arduino y le pasamos los parametros del nombre del puerto y configuración de los datos a 9600 bps. [Nota importante: lo que muestra la linea 136 es exclusivo para sistemas Mac Os X, ya que en Windows sólo se colocaría: *arduino=serial.Serial('COM3', 9600)* por eso tuve unas complicaciones al conectar los puertos a Python porque desconocía las configuraciones típicas para el sistema].

3.- Se programa un if/else, por si se cumple con la premisa del cubre bocas se escriba sobre el Arduino y si no pues el sistema se queda estático o se apaga en caso de que la persona se quitó el cubrebocas.

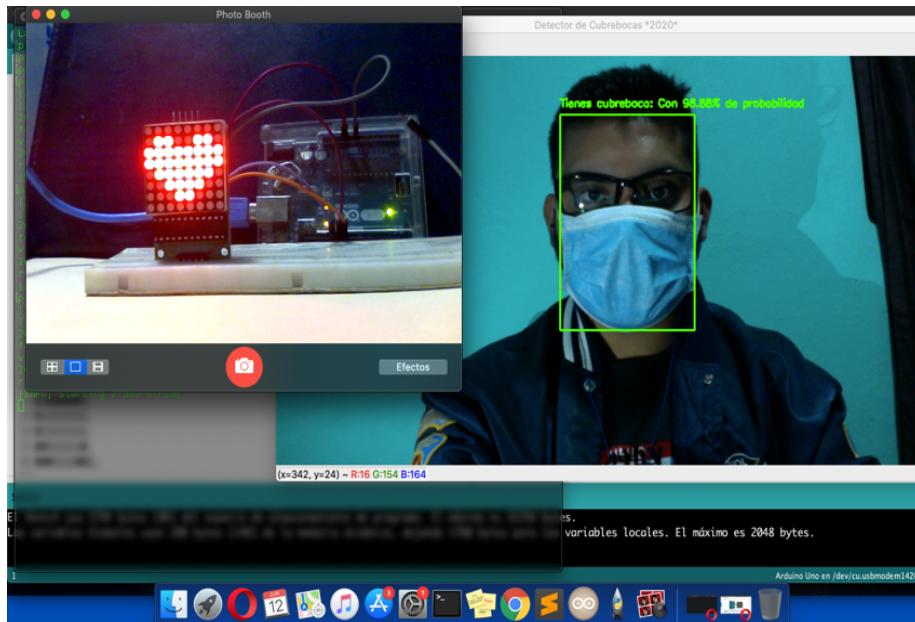
## 4 Results and discussion

Después de la sección anterior el programa está listo para ejecutarse, el Arduino ya tiene el código que va a ejecutar y entonces se ejecuta el archivo `detect_mask_video.py` con el comando: `python3 detect_mask_video.py` y muestra la imagen de la Figura 7.



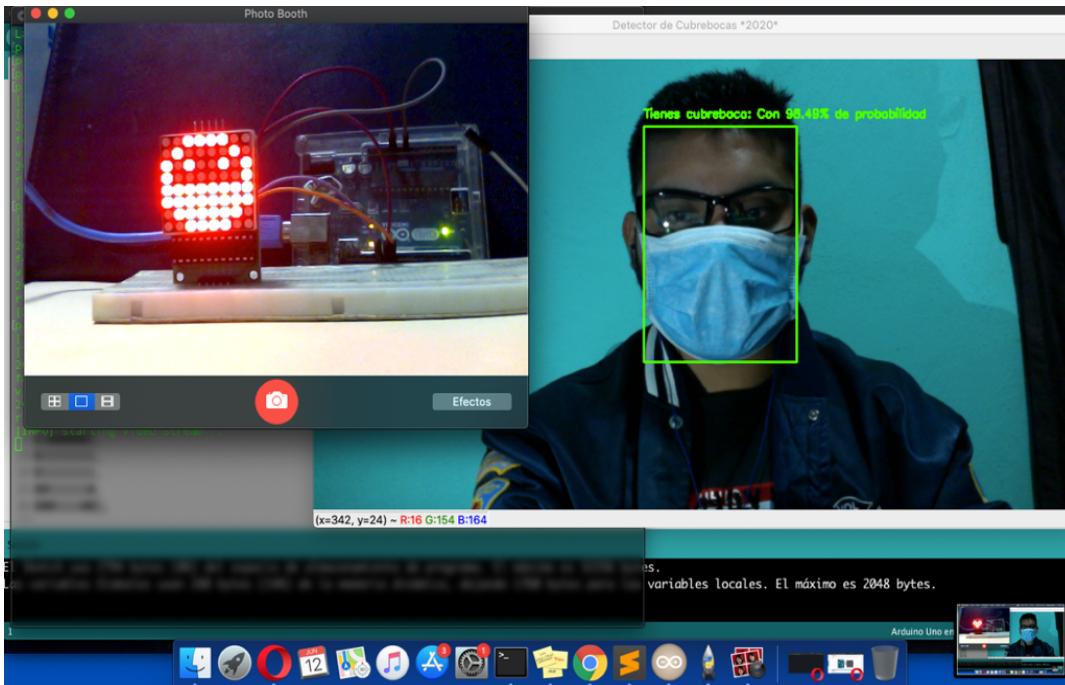
**Fig. 7:** Se muestra la ejecución y cómo el programa no responde a una persona sin cubrebocas.

Consecuentemente la persona debe tener cubrebocas, así como se muestra en la Figura 8.



**Fig. 8:** El microcontrolador responde a la premisa que cumple con el cubrebocas, entonces la matriz8x8 manda un mensaje o alerta.

¡Listo puedes ingresar de forma segura al espacio público!



**Fig. 9:** La matriz manda un emoji con cubrebocas después de 5 segundos transcurridos.

## 5 Conclusiones

Se llegó a la conclusión de que el proyecto es 100% viable para la implementación en la vida real, ya que los tiempos actuales así lo requieren.

El programa *detect\_mask\_video.py* sí manda la señal en tiempo real para poner en movimiento un accionador (puede ser un rele, servomotor, alertas, etc.)



## Referencias

1. <https://gacetamedica.com/investigacion/descifrando-los-origenes-del-sars-cov-2/> [Consultada el día 10 de junio de 2020]
2. <https://www.who.int/es/emergencies/diseases/novel-coronavirus-2019/advice-for-public/when-and-how-to-use-masks> [Consultada el día 7 de junio de 2020]
3. <https://www.elnortedecastilla.es/sociedad/salud/proyecciones-llevan-cierre-20200313215232-ntrc.html> [Consultada el día 5 de junio de 2020]
4. <https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/> [Consultada el día 2 de junio de 2020]
5. <https://arduino.cl/que-es-arduino/> [Consultada el día 8 de junio de 2020]